Contents

- User Guide
- Contents
- Creating an account
  - Activating your account
  - Changing your password
  - Updating your profile picture
- First steps
  - Account Configuration
- Reviewing results
  - Dashboard & Inventory Views
  - Insights from queries
  - Reports
  - Alerts
- Installing an explorer
  - Installation
  - System requirements
  - Web screenshots
  - Configuration
  - Network communication
  - Removing an explorer
  - Log management
  - Restart an explorer
  - Certificate Authorities (CAs)
  - Manual mode
  - Storage locations
  - Container installations
  - Automated installations
- Managing explorers
  - Viewing all explorers
  - Screenshot capabilities
  - Searching for explorers
  - Explorer actions
  - Bulk management operations
  - Viewing explorer details
  - Upgrading npcap
- Installing on a Raspberry Pi
  - What you’ll need
  - Connecting to your Raspberry Pi via SSH
  - Next steps
- Automated MSI deployments
  - Binary downloads
• Installing self-hosted Rumble
  • Requirements
  • Offline mode
  • Installation steps
  • Offline installation
  • Installation with your own PostgreSQL database
• Rumble updates
• Managing users
• CLI service management
• CLI update management
• CLI update with offline mode
• CLI user management
• CLI organization management
• Advanced configuration
• Permissions
• Rumble binary verification
  • Binary downloads
• Managing your team
  • Single sign-on (SSO)
  • Global roles
  • Account settings
• Bulk importing users
  • Creating the CSV file for importing users
  • Importing users into Rumble
  • Verifying users have registered
• Setting up Azure AD SSO
  • Requirements
  • Step 1: Add and configure Rumble as an Azure app
  • Step 2: Download the SSO configuration metadata
  • Step 3: Set up Azure AD SSO in Rumble
  • Step 4: Add users to the Rumble app in Azure
• Setting up Okta SSO
  • Requirements
  • Step 1: Add and configure Rumble as an Okta app
  • Step 2: Set up SSO in Rumble
  • Step 3: Add users to the Rumble App in Okta
• Organizations and sites
  • Organizations
  • Sites
• Discovering assets
  • Site
  • Explorer
  • Discovery scope
  • Scan name
  • Scan speed
• Schedule
  • Scheduling grace period
  • Advanced scan options
• Managing tasks
• Using the inventory
  • Understanding assets
  • Loading assets
  • Connecting to other systems
  • Viewing services
  • Viewing screenshots
  • Viewing wireless networks
• Understanding assets
  • Asset fields
• Exporting asset data
• Data retention
  • Stale asset expiration
  • Offline asset expiration
  • Scan data expiration
• Credentials
• Exporting HP iLO data
  • How to export HP iLO CSV data in Rumble Enterprise
  • HP iLO CSV export data
• Enriching scans with EC2
  • Find explorers with EC2 enrichment capabilities
  • Add permissions to describe instances
  • Attributes Rumble gets from the EC2 API
• Viewing coverage reports
  • RFC1918 coverage report
• Search query syntax
  • Boolean operators
  • Wildcard and fuzzy searches
  • Time and date values
  • Empty values
  • Asset and service inventory searches
• Search queries
  • Asset search keywords
  • Service search keywords
  • Wireless search keywords
  • Analysis report search keywords
  • Query library search keywords
  • Explorer search keywords
  • Task search keywords
  • Site search keywords
  • Organization search keywords
  • Credential search keywords
• Your team search keywords
• Event search keywords
• Automating queries
  • Turn on automatic search queries
• Managing alerts
• Using the rules engine
  • Key concepts
  • Create a rule
• Creating alert templates
  • Template building basics
  • Objects and fields reference
  • Example: Alert when scan completes
  • Data type accepted by each channel
  • Managing templates
• Managing licenses
  • How do I view my license?
  • When does my Professional or Enterprise subscription expire?
  • How do I renew my Professional subscription?
  • How do I renew my Enterprise plan?
  • How do I convert to the Starter Edition?
  • How do I find my invoices?
  • How do I change or cancel my subscription?
• Using the scanner
  • Starter Edition limits
  • Performance & scope
  • Automatic web screenshots
  • Rumble scanner commands and options
  • Scan outputs
  • Raw Scan Data
• Site Comparison Report
  • Generate a Site Comparison Report
  • View how assets change over time
  • View how exposure differs between networks
  • Analyze the results in the Site Comparison Report
  • Search the Site Comparison Report
• ServiceNow CMDB
  • Before you begin
  • Set up an ETL import
  • Things to know about Rumble export data
  • Useful links
• Splunk Search
  • Get the Rumble add-on for Splunk
  • Asset sync modes
• CrowdStrike Falcon
  • Getting started
• Requirements
• Step 1: Configure CrowdStrike to allow API access to Rumble
• Step 2: Add the CrowdStrike credentials to Rumble
• Step 3: Set up and activate the CrowdStrike connection to sync data
• Step 4: View CrowdStrike assets
• Azure Virtual Machines
  • Getting started
  • Requirements
  • Step 1: Configure Azure to allow API access through Rumble.
  • Step 2: Add the Azure credential to Rumble.
  • Step 3: Set up and activate the Azure VM connection to sync data
  • Step 4: View Azure assets
• Amazon Web Services
  • Getting started
  • Requirements
  • Step 1: Configure AWS to allow API access through Rumble
  • Step 2: Add the AWS credential to Rumble
  • Step 3: Set up and activate the AWS EC2 connection to sync data
  • Step 4: View AWS assets
• Miradore MDM
  • Getting started
  • Requirements
  • Step 1: Create a Miradore API key.
  • Step 2: Add the Miradore API key to Rumble.
  • Step 3: Set up and activate the Miradore MDM connection to sync data
  • Step 4: View Miradore assets
• Censys Search & Data
  • Censys Search API
  • Censys Universal Internet Dataset
• Leveraging the API
• Data formats
  • Formats
  • Scan data
  • Asset data
  • Change reports
• Frequently Asked Questions
  • Why are there so many identical assets in my inventory?
  • How do I run Rumble without crashing my router?
  • How do I scan VMware virtual machines without crashing the host?
  • Why didn’t the Rumble Explorer capture screenshots?
  • What protocols does Rumble scan for?
  • What ports does Rumble scan?
  • Can I safely scan my IoT or OT environments?
• Rumble release notes
  • Latest release notes
• Older platform release notes
  • Platform release notes
• Older explorer release notes
  • Explorer release notes (agents)
• Older scanner release notes
  • Scanner release notes
Creating an account

To get started, you'll need to sign up for a Rumble account. Rumble offers three editions: starter, professional, and enterprise. Choose the edition that works for you.

- **Enterprise account**: This is suited for large organizations or service providers and those who want to take advantage of Rumble integrations.

  Get a quote for an enterprise account

- **Professional account**: This is great for mid-size and larger companies, as well as consultants, who need to scan more than 256 assets. You'll need a business email to sign up.

  Sign up for a professional account

- **Starter account**: This is great for personal or small business use, if you have less than 256 assets. You can use a business or personal email to sign up.

  Sign up for a starter account

Activating your account

After you sign up for an account, we'll email you a link to activate your account. If you don't see an email from us, check your spam folder.

Open the link in the email to go to the **Activation** page. Follow the instructions on the page to activate your account. You'll need to provide your name, set up a password, specify your location, and accept our privacy policy and terms of service.

After activating your account, you'll be taken directly to the Rumble Console. Your new account has administrative access, so you will be able to manage sites, organizations, users, and explorers.

If you have any trouble creating your account, please contact support.

Changing your password

To change your password, go to your account settings. You'll need to provide your current password before you can enter a new one.

All passwords must contain:

- At least 8 characters
- At least 1 uppercase character
• At least 1 lowercase character
• At least 1 number

**Updating your profile picture**

User profile images are managed through Gravatar and associated with your email address. If you don't have an account, [sign up for one](http://www.gravatar.com).
First steps

Once your account is set up, you'll need to take a few first steps to onboard users, organize your data, and get scans started.

Account Configuration

- **Create users**: The person who signs up for Rumble service will be given superuser permissions. See the “Managing Your Team” page to learn how to create additional users, set up single sign-on and multi-factor authentication options, and discover the user roles available.

- **Set up organizations and sites**: Rumble scans and explorers can be grouped into organizations that contain sites. Sites can have overlapping address space. Create any organizations and sites that you need.

- **Deploy explorers**: Explorers are the active scanners that will be used to discover your network. First switch into the organization where you would like to store your results and deploy an explorer. Make sure to review the system requirements and dependencies for web screenshots as you plan where to install the explorer.

- **Create scan tasks**: After deploying your explorers, create your first scan task. A discovery scan finds, identifies, and builds an inventory of all the connected devices and assets on your internal network. The explorer and discovery scope are the most important aspects of configuration, followed by SNMP community information since switch topology can't be generated without it.
Reviewing results

Dashboard & Inventory Views

Once scans complete the dashboard will be populated with results. The dashboard provides trend data and insights that will help you assess how your inventory is changing over time. You can select a time period and site for the trend data using the selectors at the top right of the dashboard page.

The main asset trends graph shows the number of assets in each of the four main states – live, offline, scanned and unscanned. Beneath the graph are additional asset breakdowns, each of which shows a top 10 of an asset category – asset type, operating system, hardware and tags.

The service trends graph shows how many total services were found in your asset inventory, along with breakdowns for ARP, ICMP, TCP and UDP. Below the service trends graph are breakdowns of the top 10 TCP ports, UDP ports, protocols and products detected.

Clicking “View more” at the bottom of each table shows a more detailed inventory by category.

Insights from queries

Queries and reports can help you gain valuable insights, but you may wonder where to get started. We recommend trying the pre-built queries in the Query Library first. Some of these queries are a result of Rumble’s Rapid Response to emerging threats and are described on our blog.

Rumble’s query language allows you to search and filter your asset inventory based on asset fields and value pairs. See the Overview and Syntax references. Once you are familiar with the query language you can write your own queries.

With a Rumble Professional or Enterprise license, you can set queries to run automatically by opening the query and setting the “Automatically track query results on the dashboard”. The query will run when scans complete, and you will be notified of any resulting insights on the dashboard page.

Sample Queries

**Asset inventory**

- Equipment that is likely 8+ years old: alive:t mac_age:>8years
- Assets with end-of-life OS: os_eol:<now
• Virtual machines: has:virtual
• Devices acting as a router: router:true
• Devices that may be bridging: has_public:t and has_private:t

Service inventory

• Protocol on a non-standard port example: protocol:ssh not port:22
• Publicly addressed assets running RDP or VNC: has_public:t and (protocol:rdp or protocol:vnc)
• Authenticated web services that are not encrypted: (_asset.protocol:http AND not _asset.protocol:tls) AND (html.inputs:"password:" OR last.html.inputs:"password:" OR has:http.head.wwwAuthenticate OR has:last.http.head.wwwAuthenticate )
• Older TLS versions in use: alive:t AND protocol:"=tls" AND (tls.versionName:"=TLS 1.0" OR tls.versionName:"=TLS 1.1")

Some other sample queries are described in our blog entries:

• Finding duplicate SSH host keys
• Identifying rogue remote access solutions
• Finding device serial numbers

Reports

After viewing the dashboard and inventory, your next stop should be the Rumble Reports page.

Switch topology

This report uses SNMP information to map how the switches on your network are connected. Each switch displays its IP address, name, and the number of assets connected to it. If Rumble detected MAC addresses that were not found as part of the scan scope, you will see a number of unmapped assets indicated below the switch.

You can click on a switch to see a pop-up with the number of identified and unmapped assets. From there, you can click to view the unmapped assets, and be taken to a table of unmapped MACs by switch port.

Double-clicking on a switch will expand that part of the diagram and show the individual assets connected to the switch.
**Subnet utilization**

The subnet utilization report lists the subnets scanned on your network, and what percentage of each is in use. For example, if you have scanned 10.0.1.0/24 and found 25 assets, the report will show that 10% of the available IP addresses in the subnet are in use.

**Network bridges**

The network bridges report is a way to find devices that bridge multiple network segments. It can be useful to locate unintentional bridging between your internal networks and the Internet.

The report shows your internal networks in green, and external networks in red. It then shows you the multihomed assets which bridge an internal network to an external one.

**RFC 1918 coverage**

The RFC 1918 coverage report is a way to view how much of the private internal network address space has been scanned for assets. It can help you discover rogue assets, unscanned subnets, and secondary interfaces on scanned devices. More information is in the section on coverage reports.

**Unmapped MACs**

The unmapped MACs report lists MAC addresses Rumble found evidence for, but which weren’t encountered as addresses of assets during the network scan. The MAC addresses are grouped by the switch that reported them, along with information about the vendor, manufacture date and switch port of the possible asset, to help identify them.

**Outliers**

The outliers reports allow you to obtain a summary of how often different values occur in specific attributes of assets and services. The values are sorted from most frequent to least frequent.

For example, the HTTP servers outliers report will list all of the HTTP servers encountered by Rumble, starting with the most common.

As well as the one-click outliers reports, you can produce an outliers report for any asset or service attribute.
• Switch topology to identify how your assets are connected and find “unmapped” MAC addresses (in red) that were not included in your scan scope (a summary of which is in the Unmapped MACs report)
• Bridging to visualize what hosts may have both public and private connections
• RFC 1918 coverage that can identify potential blindspots on your network like missing (unscanned) subnets, rogue devices, and “hinted” IPs that are secondary interfaces on unscanned network ranges.
• See the “View all” button at the top right for a list of other reports to investigate outliers

**Domain membership report**

The domain membership report lists the Active Directory domains encountered by Rumble, and lists how many assets are in each.

**Analysis reports**

Analysis reports are reports which run as tasks, rather than being generated on-the-fly. The first analysis report is **Compare Sites**, which generates searchable reports of the differences between two sites.

**Alerts**

As well as manually generated reports and queries, Rumble also supports [automatic alerts](#) to designated channels for post-scan inventory queries, asset changes, explorer and scan issues, security operations, or API events.

Available channels are internal notifications in the Rumble web console, email, or webhooks that can enable integration with services such as Slack or Mattermost. Alerts use the same query language as the sample queries above, so this is a good way to automate proactive notification for critical events.
Installing an explorer

Rumble requires the use of at least one Explorer within your environment to enable network discovery. The explorer should be installed on a system with reliable connectivity to the network you want to discover. For internal networks, Rumble works best when installed on a system with a wired (vs wireless) connection.

For external network discovery, nearly any cloud provider with a reliable connection should do. If the Rumble Explorer is installed in a container or virtualized system, ensure that it has direct access to the network (host networking in Docker, bridged networking in VMware, etc).

Installation

To install the Rumble Explorer, log in to the Rumble Console and switch to the Organization that should be associated with the explorer. Explorer downloads are then available by selecting Deploy in the left navigator and choosing the Deploy explorers tab.

**Note:** The explorer download link is specific to your active Organization and using the wrong link can result a new explorer being associated with the wrong organization.

Download the correct binary for your system from the explorer download page. For most systems, select the 64-bit (x86_64) architecture. For macOS, you will need to select 64-bit Intel (x86_64) or ARM (Apple M1), depending on your hardware. For embedded devices, such as the Raspberry Pi 3+, choose the ARM7 architecture. Windows binaries are signed with a valid Authenticode signature, which should be validated before the executable is launched.

The explorer installation process requires administrative privileges. On Windows, a UAC prompt may be displayed. On Linux and macOS the downloaded binary should be made executable (chmod u+x rumble-explorer.bin) and then executed with root privileges (sudo or from root shell). In either case, the explorer should install itself as a system service and start immediately, displaying a new entry in the explorers page.

System requirements

Windows

- Windows Server 2012 R2+ or Windows 10 Build 1604+
- Processor running at 2.0 GHz or faster
- At least 16GiB of memory (8GiB for small environments)
- At least 1GB of free storage space
Windows Server 2008, Windows Server 2012, Windows 7, and Windows 8 may be able to run the explorer in a pinch, but are not officially supported.

**Linux**

- Kernel version 2.6.23 or later
- Processor running at 2.0 GHz or faster
- At least 16GiB of memory (4GiB for small environments)
- At least 1GB of free storage space

Linux ARM devices with limited processing power and memory, such as the Raspberry Pi, can run the Rumble Explorer, but may have trouble scanning larger networks.

**MacOS**

- macOS 10.11 (El Capitan) or newer
- Processor running at 2.0 GHz or faster
- At least 16GiB of memory (8GiB for small environments)
- At least 1GB of free storage space

macOS systems running Catalina (10.15) or newer need to use the `curl` download method to avoid issues with the new Notary requirements.

**BSD variants**

- Processor running at 2.0 GHz or faster
- At least 16GiB of memory (4GiB for small environments)
- At least 1GB of free storage space

Requires root access to a system running a recent version of the operating system. FreeBSD 11.2 or newer, recent versions of NetBSD/DragonFly/OpenBSD.

**Web screenshots**

*Google Chrome* or *Chromium* should be installed on the Explorer system to enable *web screenshots*. Please note that “snap”-based Chromium installs (Ubuntu 20.04 and newer) don’t appear to work properly in headless mode and the official Chrome packages should be used instead.

**Configuration**

The explorer can be configured by setting variables in a `.env` file located in the same directory as the executable. On Windows this file should be created in `C:\Program`
Files\Rumble\env, while other platforms should use /opt/rumble/bin/.env. The format of this file is VAR=VAL with one variable per line.

**Configuration file locations**

**Windows:** C:\Program Files\Rumble\env

**Other Platforms:** /opt/rumble/bin/.env

**Network communication**

The explorer connects to the console.rumble.run host on TCP port 443 using TLS and two static IPv4 addresses (13.248.161.247, 76.223.34.198). This connection is used for explorer registration, job scheduling, status messages, and submission of completed scan jobs. For completely offline environments, the Rumble Scanner can be used to create scan data files that can be uploaded later via the Inventory Import action. The host console.rumble.run is used for automatic updates of the explorer executable.

Please note that certain web proxies that perform TLS inspection do not handle Websocket communication properly and TLS inspection will need to be disabled for the Rumble Explorer to successfully connect. The most popular product with this problem is the Sophos (previously Cyberoam) security appliance. Websense users may need to add a bypass rule for console.rumble.run.

Proxy support is handled automatically in most cases. On the Windows platform, proxy information is read from the registry keys (used by Chrome, Edge, and IE).

The proxy can be configured by setting the HTTPS_PROXY environment variable. The value of the HTTPS_PROXY environment variable should be a hostname and port (proxy:8080) or just a hostname (proxy). Environment variables are read from your configuration file. Please view the **Configuration** section to see how to set environment variables.

**Removing an explorer**

The easiest way to remove an explorer is to use the Explorers page Manage menu and choose the Remove explorer option. This will remove the service and terminate the current explorer process. If you would like to remove the explorer without using the Rumble Console, there are a couple options.

On the Windows platform, each explorer will be listed in Programs and Features (as the Rumble Agent), and can be uninstalled like any other application.
On all platforms, including Windows, the explorer can uninstall itself if run with the uninstall argument from a root or Administrator shell:

**Windows**

c:\Program Files\Rumble\rumble-explorer-[oid].exe uninstall

**Other Platforms**

/opt/rumble/rumble-explorer-[oid] uninstall

**Log management**

The explorer logs to a file and to standard output by default. On Windows the default log file location is the installation directory (C:\Program Files\Rumble) while other platforms log to the files /var/log/rumble.log and /var/log/rumble.err. The default configuration limits log files to 100Mb, creates three backups, and expires logs after 90 days. These defaults can be be changed by setting the following values in the .env file:

- The RUMBLE_AGENT_LOG_MAX_SIZE setting controls the maximum log size in megabytes. The default is 100.
- The RUMBLE_AGENT_LOG_MAX_BACKUPS setting controls the number of backup files created by log rotation. The default is 3.
- The RUMBLE_AGENT_LOG_MAX_AGE setting controls the maximum age in days, this applies to all files, including backups. The default is 90.
- The RUMBLE_AGENT_LOG_COMPRESS setting determines whether to gzip compress the backups. The default is false.
- The RUMBLE_AGENT_LOG_STDOUT setting determines whether to write logs to standard output (and syslog for systemd/upstart). The default is true.

**The explorer must be restarted for these settings to take effect.**

**Restart an explorer**

The quickest way is to force an update from the cloud console, otherwise you can find the service name and restart it by hand.

On Linux systems using systemd, first obtain the name of the explorer (rumble-agent) service:

```
$ systemctl | grep rumble-agent
```

Then restart the service using this name:

```
$ systemctl restart rumble-agent-[uuid-value]
```
A kill -9 of the explorer pid should cause a restart as well.

**Certificate Authorities (CAs)**

The Rumble Explorer uses the system-installed certificate authorities to validate TLS connections in addition to an internal CA certificate bundle (derived from Debian 10). By default, both the system certificate roots, and the bundled roots are considered for all secure TLS connections. This behavior can be controlled via environment variables (set in the .env file or at the system level):

- The RUMBLE_TLS_IGNORE_SYSTEM_ROOTCA setting can be set to `true` to ignore the system CA roots.
- The RUMBLE_TLS_IGNORE_EMBEDDED_ROOTCA setting can be set to `true` to ignore the bundled CA roots.
- The RUMBLE_TLS_ADDITIONAL_ROOTCA setting can be set to a file path containing additional CA roots in PEM format.

**Manual mode**

If a supported system service manager, such as systemd or upstart, is not detected, the Rumble Explorer will switch to manual mode, running in the foreground, and replacing and re-executing its own binary as new updates become available. For temporary explorer installations or to run the explorer in a container environment, the argument “manual” can be specified:

```
$ sudo ./rumble-explorer.bin manual
```

**Storage locations**

The Rumble Explorer installs into %PROGRAMFILES%\Rumble on Windows and /opt/rumble on all other platforms. Temporary files are stored in the default operating system locations. These locations can be overridden using the .env file. Note that the explorer service needs to be restarted (or force updated) for these changes to take effect.

On Windows, the temporary file location is chosen from the first non-empty environment value of TMP, TEMP, or USERPROFILE, falling back to the Windows directory. To override this location, set an entry in .env like the following:

```
TMP=D:\Storage\Rumble
```

On all other platforms, the temporary file location is chosen based on the value of TMPDIR, falling back to /tmp otherwise. To override this location, set an entry in .env like the following:
TMPDIR=/home/storage/rumble

Any scans that fail to upload are stored in the Rumble Explorer installation directory and can be imported into the platform manually or using the Rumble Scanner's --import and --upload options.

**Container installations**

The Rumble Explorer can run in standard container environments, but may require additional configuration. To run as a standalone executable, the explorer can be run with the argument manual. For non-persistent containers an explorer identifier needs to be persisted through an environment variable. This can be done by setting the variable RUMBLE_AGENT_HOST_ID to a 32-character hexadecimal string. This identifier is used to uniquely identify the explorer within an organization.

To generate a suitable identifier, the openssl tool may be used:

```
$ openssl rand -hex 16
01b0283809b24511929d0b062bd36109
```

Here is a sample Containerfile you can edit and use:

```
## Sample Containerfile for running the Rumble Explorer in a container, with screenshot support.
FROM debian:stable-slim
WORKDIR /opt/rumble
RUN apt update && \
    apt install -y chromium # add wireless-tools if you want WiFi scanning
## Set AGENT_URL to be the download URL for your Linux Rumble Explorer. To find your URL, go to https://console.rumble.run/deploy/download/explorers and click on the first URL box to copy it to the clipboard.
ENV AGENT_URL=https://console.rumble.run/download/explorer/E94D4DD347C0408F8F6B6-
```
This ID is used to track the explorer even if the container is rebuilt. Set it to a unique 32 character hex ID. You can generate one via:

```
$ openssl rand -hex 16
```

```
ENV RUMBLE_AGENT_HOST_ID=112233445566778899aabbccddeeef
```

If you need to set environment variables to change the explorer behavior, you can do so via the ENV directive. Example:

```
ENV RUMBLE_AGENT_LOG_DEBUG=true
```

```
ADD ${AGENT_URL} rumble-explorer.bin
RUN chmod +x rumble-explorer.bin
```

For full functionality the Rumble scanner needs to send and receive raw packets, which requires elevated privileges.

```
USER root
```

The argument `manual` tells Rumble not to look for SystemD or upstart.

```
ENTRYPOINT [ "/opt/rumble/rumble-explorer.bin", "manual"]
```

This containerfile works with podman as well as Docker. Note that because of the requirement for root privileges, you should start the container as root.

**Automated installations**

The explorer will automatically install when executed if root or administrative privileges are available.

On Linux and BSD systems, automatic installation depends on the presence of a supported init service like systemd or upstart. If no supported init service is found, the explorer will instead run in manual mode, automatically overwriting and re-executing itself with each update. To automatically deploy an explorer on systems without a supported init service, the explorer should be executed in the background and with the nohup wrapper.

On Windows systems, the explorer will automatically install when run interactively or when the updater parameter is passed to the binary. For environments where MSIs are required, the Explorer MSI wrapper can be used to deploy an explorer from the Rumble Console or a local mirror.
Managing explorers

The Rumble Explorer is a lightweight scan engine that enables network and asset discovery. You should have at least one explorer deployed per broadcast domain. After deployment, you can manage your explorers from the Deploy page in your Rumble web console.

Viewing all explorers

For each explorer, you can see:

- The explorer status (whether it is communicating with Rumble)
- The OS it is running on
- Its name
- Any site it is associated with
- Its IP addresses
- The software version it is running
- Whether the version of npcap installed is up-to-date, if the OS is Windows (see upgrading npcap below)
- The CPU architecture of the host machine
- Any tags associated with the explorer
- The status of its last scan
- Its capabilities, like Chrome support

Screenshot capabilities

To capture screenshots, Chrome must be installed. You can check if an explorer has screenshot capabilities by looking for the Chrome icon in the Capabilities column.

Here's what each icon means:

- Green icon - The explorer has access to a Google Chrome binary and can take screenshots.
- Red icon - No suitable Chrome binary was found.

Searching for explorers

You can use the search bar to find explorers. The query syntax is similar to other search bars in Rumble, with keywords to filter by specific fields:

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Search by</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>arch:</td>
<td>CPU architecture</td>
<td>arch:amd64</td>
</tr>
</tbody>
</table>
name: scanner.local
address: 10.0.1.200
capability: screenshot or capability: aws
tag: dev
npcap_version: 1.31

Explorer actions

Each explorer has a set of action buttons that allow you to:

- **Reinstall an explorer** - Performs a reinstall or upgrade of the explorer. The current explorer will download the latest explorer code from Rumble, and then run the install process.
- **Configure an explorer** - You can associate the explorer with a specific site, and add tags to it. You can also set the maximum number of concurrent scans allowed – for Professional and Enterprise licenses, a single explorer can be configured to run multiple scan tasks at once.
- **Reassign an explorer** - You can reassign an explorer to a different organization within your account or even to a different Rumble client account entirely.
- **Remove an explorer** - If the explorer is running, the explorer will be asked to uninstall itself from the host machine. If the explorer is not running, you can still tell Rumble to forget about it. This is useful if you have decommissioned the machine the explorer was running on or uninstalled the explorer manually. If the explorer runs again after Rumble has been told to forget it, it will be readded to the registered explorers list.

Bulk management operations

Bulk operations allow you to perform a set of actions to multiple explorers at one time. Bulk actions are available from the Manage All Explorers menu.

You can bulk:

- **Update all online explorers** - Tells all explorers—that are up and communicating with Rumble—to upgrade their software.
- **Forget all offline explorers** - Clears all explorers currently offline, and makes Rumble forget them. No data will be lost. If any of the explorers are reactivated, they will be added back to the active list.
- **Uninstall all online explorers** - Tells all online explorers to uninstall themselves from their host systems.
- **Automatically assign sites** - Runs through all of the explorers that are not currently assigned to a specific site. It checks their IP address against the CIDR IP ranges of the registered subnets of all sites in the current organization. If the
explorer’s IP address only matches a single site, the explorer is assigned to that site.

Viewing explorer details

Clicking on an explorer’s name takes you to a page showing the diagnostic information for that explorer, including its software version, available memory, and network interfaces.

At the bottom of the page is a diagnostics text area. Clicking the Update Diagnostics button will fetch an updated list of all sub-processes active within the explorer. This is useful to send to Rumble support if you are having problems with a particular explorer.

Upgrading npcap

On Windows, Rumble uses a licensed third-party library called npcap for access to raw network traffic. Other software installed on the explorer’s host machine may also use npcap, and sometimes will have installed obsolete versions of the software. This can cause reliability problems.

Rumble will alert you to obsolete versions of npcap by displaying a warning icon in the list of explorers.

However, Rumble cannot yet reliably upgrade npcap for you. Rumble can’t automatically upgrade npcap/winpcap, as it tends to be shared between applications, and forcing an upgrade from the Rumble side can break other services (EDRs, Wireshark, etc).

To upgrade npcap manually:

1. Stop any running Rumble services. This can be done using the Windows Services app. You’ll need to look for “Rumble Network Discovery Explorer”.
2. Stop any other running software which uses npcap.
3. Uninstall Winpcap and any npcap installations via the Windows Control Panel.
4. Reboot the computer.

Rumble will restart automatically, and install the latest npcap.
Installing on a Raspberry Pi

The Rumble explorer enables discovery scanning. In most cases, you can deploy an explorer on an existing system that has connectivity to the network you want to discover. However, there may be times when the traditional deployment model may not work for you. Some locations, like retail stores or customer sites, may not have staff or hardware available to install the explorer, making remote deployment a bit tricky.

In these types of scenarios, you can install a Rumble explorer on a Raspberry Pi and send the device to the location for them to plug into their network.

What you’ll need

- Raspberry Pi 4 Model B (4GB or 8GB), Raspberry Pi Compute Module 4, or Raspberry Pi 400
- At least 1GB free of storage space on your Raspberry Pi’s MicroSD card after installing the operating system

Connecting to your Raspberry Pi via SSH

In this section, we’re going to show you how to SSH to your Raspberry Pi and install the explorer from your terminal.

Step 1. Enable SSH on your Raspberry Pi

Before you can connect to your Raspberry Pi, make sure to enable SSH on the device.

You can add a file called ssh.txt at the root of the SD card and reboot the Raspberry Pi. The contents of the file can be empty. On reboot, the Pi looks for the SSH file and enables SSH if it finds the file.

Step 2. Update the password for your Raspberry Pi

If you enable SSH on your Raspberry Pi, you must update your credentials.

Log in as the pi user and use the passwd command to change the default password. Entering the passwd command will prompt you for your current password to authenticate before you can change it.

Step 3. Copy the explorer instructions

- Go to your console. Verify you are in the right organization. Rumble keys your explorer’s download link to the organization you are currently viewing to associate
them together.

- Go to the **Explorer Deployment page** and select **Linux Distributions**, and then choose Linux ARM 32-bit V7. Note that Raspbian uses a 32-bit kernel by default, even on 64-bit Raspberry Pi hardware.

- Click the instructions at the bottom of the Linux installation page to place the commands into your clipboard.

**Step 4. Install the explorer**

Paste the installation command into the terminal after connecting to the Raspberry Pi via SSH. The URL in the command links the installation to your active organization and will be different from the example below.

```
me@mac ~ % ssh pi@hostIP
pi@host's password:
pi@raspberrypi:~ $ sudo bash
root@raspberrypi:/home/pi# cd Downloads
root@raspberrypi:/home/pi# curl -o rumble-explorer.bin https://console.rumble.run/download/ex
```

The explorer automatically installs itself in /opt/rumble and sets up a systemd service with the name `rumble-agent-`. The service automatically starts on boot.

A successful installation shows the following screen:

```
INFO installing rumble-agent-8775adf-d3-0c8-4f48-b2f8-6a787457762f to /opt/rumble/bin/rumble-agent-8775adf-d3-0c8-4f48-b2f8-6a787457762f
INFO writing executable to /opt/rumble/bin/rumble-agent-8775adf-d3-0c8-4f48-b2f8-6a787457762f
INFO installing service rumble-agent-8775adf-d3-0c8-4f48-b2f8-6a787457762f
INFO starting service rumble-agent-8775adf-d3-0c8-4f48-b2f8-6a787457762f
INFO installation complete
```

**Next steps**

Your Raspberry Pi is now set up to use as a Rumble Explorer box and ready to be sent wherever you need.
You can always view and manage your registered explorers from your console.
Automated MSI deployments

Rumble uses dynamically generated binaries for the Rumble Explorer downloads and this doesn't always play well with MSI-based installation methods.

To work around this issue, we have provided a shim MSI package that can be used with automated installers. This package has a valid Authenticode signature and can also be verified using the Rumble Verifier.

To use this package, deploy it with the URL parameter specified as the organization-specific download URL from the Rumble Console explorers section.

C:\> msiexec /i rumble-explorer-installer-amd64.msi URL=https://console.rumble.run/download/e

The MSI shim will verify that the URL contains a valid Rumble Explorer binary and install it normally.

**Warning:** Note that installing with the /a parameter will not work and /i must be used instead.

### Binary downloads

<table>
<thead>
<tr>
<th>Build</th>
<th>SHA256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumble Explorer Installer MSI x86 64-bit</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Explorer Installer MSI x86 32-bit</td>
<td>sha-256</td>
</tr>
</tbody>
</table>
Installing self-hosted Rumble

Note: Self-hosting is only available with Rumble Enterprise

The self-hosted version Rumble allows you to run the entire platform on-premises or within your own cloud environment. This platform is functionally identical to the hosted service, provides a fully-offline mode, and does not send any inventory data back to Rumble.

Requirements

Before you get started, make sure your system meets the following requirements.

Hardware requirements

- 4 CPU Cores at 2 GHz or faster
- 100 GB of local disk storage
- 32 GB of RAM (more for large sites)

Software requirements

- PostgreSQL 12 or newer (Our installer can install this for you)

Supported operating systems

- Ubuntu 20.04 x86_64
- Ubuntu 18.04 x86_64
- Red Hat Enterprise Linux 8.x x86_64
- Red Hat Enterprise Linux 7.x x86_64
- CentOS Linux 8.x x86_64
- CentOS Linux 7.x x86_64
- Oracle Linux 8.4+ x86_64 with UEK 5.4 or kernel 4.18+
- Oracle Linux 7.9+ x86_64 with UEK 5.4 or kernel 3.10+
- Debian Linux 11.x x86_64
- Debian Linux 10.x x86_64
- Debian Linux 9.x x86_64

Connectivity

The self-hosted Rumble platform requires the following connectivity for online updates:

- Outbound access to console.rumble.run (13.248.161.247, 76.223.34.198) on TCP port 443 for HTTP over TLS.
The system running the Rumble platform should have a static IP address. If the IP is changed at a later date, the Explorers may need to be redeployed to receive the updated address.

**Offline mode**

The self-hosted version of Rumble has the ability to run in offline mode. In this mode, the Rumble update service is not used and offline updates must be applied manually. Enable this mode if you're in an isolated network or you don't want your self-hosted Rumble console to make any connections to the internet. In addition to disabling online updates, offline mode also disables certain DNS probes that could reflect responses to the internet during a scan.

**Installing with offline mode**

- Run the installer with the --offline flag in combination with --distro-packages-only or --postgres-rpm-directory

More information can be found under Offline installation.

**Enabling offline mode for existing installs**

- Open /etc/rumble/config with an editor of your choice.
- Look for OFFLINE= and change it to OFFLINE=true.
- Restart the Rumble service rumblectl restart.

**Updating with offline mode**

More information can be found under CLI update with offline mode.

**Installation steps**

For offline installs please see Offline installation.

For installs that use your own database credentials see Installation with your own PostgreSQL database.

Here’s what the installation process will do:

- Set up PostgreSQL and create a passworded user.
- Generate TLS certificates for your IP address located in /etc/rumble/certs.
- Generate a configuration file at /etc/rumble/config and set some defaults.
- Create a systemd service for the Rumble platform.
• Create all the necessary cron jobs required for the Rumble platform.

**Step 1: Download and run the installer**

• Go to [https://console.rumble.run/deploy/download/platform.](https://console.rumble.run/deploy/download/platform)
• Copy the command directly from the download page and run it in your terminal. This will download the installer and mark it as executable. The download path for the installer is uniquely keyed.

**Step 2: Initialize the admin user**

After you've installed the Rumble platform, you will have access to the Rumble CLI rumblectl.

To initialize an admin user, run:

rumblectl initial [email address]

**Step 3: Log in to your self-hosted console**

If everything is set up correctly, you can log in to your console at [https://YourInternalIPAddress](https://YourInternalIPAddress).

Note that you may need to enable HTTPS to pass through the Linux system firewall. Example commands:

**Ubuntu Linux:**
sudo ufw allow https/tcp

**RHEL/CentOS/Oracle:**
sudo firewall-cmd --add-service=https

To make a firewall-cmd change permanent across reboots, run the command a second time with the --permanent flag added.

**Offline installation**

The self-hosted Rumble platform comes with a few options for your installation. You can utilize these options by adding flags to the install command. The current flags available are --offline, --distro-packages-only, and --postgres-rpm-directory

--offline                                 Install the self-hosted platform in a completely isolated environment.
--distro-packages-only                    Install without using third-party repositories. (Recommended)
--postgres-rpm-directory [directory]     Install using supplied PostgreSQL RPM files. (Recommended)
PostgreSQL RPMs required for --postgres-rpm-directory

RHEL/CentOS 8 RPMs can be found here: https://download.postgresql.org/pub/repos/yum/13/redhat/rhel-8-x86_64/

RHEL/CentOS 7 RPMs can be found here: https://yum.postgresql.org/13/redhat/rhel-7-x86_64/repoview/

There are four RPMs we require for installing PostgreSQL 13

- postgresql13
- postgresql13-server
- postgresql13-contrib
- postgresql13-libs

Example install commands for offline mode

The first step is to download the Rumble platform

RHEL/CentOS 7 or 8

1. Download the required RPMs above and store them in a directory.
2. Run this install command.

./rumble-platform-[VERSION]-linux-amd64.bin install --offline --postgres-rpm-directory [RPM_DIR]

Ubuntu 18.04+, Debian 10+, or RHEL/CentOS 8

1. Run this install command.

./rumble-platform-[VERSION]-linux-amd64.bin install --offline --distro-packages-only

Installation with your own PostgreSQL database

Rumble defaults to installing and configuring a PostgreSQL user and database for you. If you would like to provide your own details this option will allow you to override that behavior.

Requirements

- PostgreSQL 13 or newer
- Password authentication must be enabled
- Two extensions are required: pg_trgm and uuid-ossp

PostgreSQL example to enable extensions and add a database/user
CREATE EXTENSION IF NOT EXISTS pg_trgm; CREATE EXTENSION IF NOT EXISTS "uuid-ossp";
CREATE DATABASE rumble;
CREATE USER rumble WITH PASSWORD 'YOURPASSWORD';
GRANT ALL PRIVILEGES ON DATABASE rumble TO rumble;

Steps to install the self-hosted Rumble platform with your own database credentials

1. Run this install command.
   ```
   ./rumble-platform-[VERSION]-linux-amd64.bin install --manual-database
   ```

2. Edit your Rumble configuration and add your database details in
   ```
   /etc/rumble/config.
   ```
The line you need to edit is:
   ```
   DATABASE_URL=postgres://rumble:{DB_PASSWORD}@127.0.0.1:5432/rumble?sslmode=disable
   ```
Change this to match your credentials. You need to set the user, password, host, port, and database name. Here is the format:
   ```
   DATABASE_URL=postgres://{DB_USERNAME}:{DB_PASSWORD}@{DB_HOST}:{DB_PORT}/{DB_NAME}?{DB_OPTIONS}
   ```

4. Verify the self-hosted Rumble platform can connect to your database with this command.
   ```
   sudo rumblectl database verify
   ```
5. Once your database is configured and verified you can restart the self-hosted Rumble platform service
   ```
   sudo systemctl restart rumble-console
   ```

Rumble updates

For offline updates please see [CLI update with offline mode](#).

The self-hosted Rumble platform must be updated prior to first use. The `rumblectl` command can be used to download the update and then restart the service after the update is complete.

You can update the platform and scanners at the same time or separately with the [CLI update management commands](#).

Managing users

You can manage users inside your self-hosted Rumble platform console at

https://YourInternalIPAddress/team or via the [Rumble CLI](#)

Some things you can manage:

- Adding, deleting, and listing users
- Resetting passwords
- Changing default roles
- Viewing details
- Changing organization roles

**CLI service management**

**Start the Rumble service**

Starts the Rumble platform service.

`rumblectl start`

**Stop the Rumble service**

Stops the Rumble platform service.

`rumblectl stop`

**Restart the Rumble service**

Restarts the Rumble platform service.

`rumblectl restart`

**Install the Rumble platform**

Install the Rumble platform service and all required dependencies such as PostgreSQL. Creates a SystemD service, generates cron jobs, and generates a configuration file in `/etc/rumble`.

`rumblectl install`

**Uninstall the Rumble platform**

Stop and remove the Rumble platform service from SystemD and removes the generated cron jobs. This does not remove your PostgreSQL database, and it retains your data.

`rumblectl uninstall`

**Purge the Rumble platform**

Stop and remove the Rumble platform service from SystemD and removes the generated cron jobs. This will delete your Rumble database and remove all the Rumble directories `/etc/rumble` and `/opt/rumble`.

`rumblectl purge`
You can uninstall and purge everything except the database and your PostgreSQL settings with this flag:

```
rumblectl uninstall --purge --ignore-database
```

**Run the Rumble platform manually**

Starts the Rumble platform manually. Logs will be written to standard output.

```
rumblectl server
```

**Verify your database is reachable**

Attempts to connect to your database using your self-hosted Rumble platform configuration. It will either succeed or display an error.

```
rumblectl database verify
```

**CLI update management**

**Update the Rumble platform and scanners**

Updates the Rumble platform service and Rumble scanners. You can use the optional parameter `force` to force the update even if the current installation is the latest version.

```
rumblectl update [--force]
```

**Update the Rumble platform**

Updates just the Rumble platform service. You can use the optional parameter `force` to force the update even if the current install is the latest version.

```
rumblectl update-platform [--force]
```

**Update the Rumble scanners**

Updates just the Rumble scanners. You can use the optional parameter `force` to force the update even if the current installation is the latest version.

```
rumblectl update-scanner [--force]
```

**CLI update with offline mode**

The self-hosted Rumble platform must be updated prior to first use.
Update the Rumble platform and scanners with an offline update

- Go to https://console.rumble.run/deploy/download/platform.
- Copy the command directly from the download page and run it in your terminal, or you can use one of the following commands to update using the zip archive you downloaded.

  rumblectl update rumble-platform-update-[VERSION].zip
  rumblectl update --offline --zip-file-path rumble-platform-update-[VERSION].zip

You will need to change the version to match the zip archive you downloaded.

**CLI user management**

Create the initial administrator account

Creates the initial admin user for a new installation. You must provide an email address.

  rumblectl initial <email>

List user accounts

Lists all the users along with their email address, full name, and current roles.

  rumblectl user list

Add a user account

Creates a new user account under the initial administrator user. You must provide an email address.

  rumblectl user add <email>

Delete a user account

Deletes a user account. You must provide an email address. This cannot be undone.

  rumblectl user delete <email>

Get user details

Gets the details for a user account. You must provide an email address. Provides information such as full name, date created, last login IP, last login
time, last activity, default organization role, and their current roles. You must provide an email address.

rumblectl user details <email>

**Set a user role**

Sets a user's role to the role provided. Email and role must be provided. The organization is optional. If the organization isn't provided this sets their default role.

rumblectl user set-role <email> [organization name or organization ID]:<role>

**Reset a user password**

This will generate and apply a new password for the specified user. The password will be printed to the terminal. You must provide an email address.

rumblectl user reset <email>

**CLI organization management**

**List all organizations**

Lists all the organizations by their name and ID.

rumblectl organization list

**Advanced configuration**

The file at /etc/rumble/config can be modified to support a wide variety of configurations.

After making changes, apply them by running rumblectl restart.

**Email server**

Rumble uses a SMTP server for user account invitations and notifications. The default configuration assumes that a SMTP server is available on localhost that does not require authentication:

SMTP_SERVER=127.0.0.1:25
SMTP_AUTH_METHOD=none

Rumble will automatically use STARTTLS with plaintext SMTP servers and validate the certificate. In internal environments where the SMTP server is not using a valid TLS certificate, verification can be disabled by setting:
SMTP_TLS_NOVERIFY=true

Transport-layer TLS (instead of STARTTLS) can be configured with:

SMTP_TLS=true

If authentication is required, the following three settings should be configured:

SMTP_AUTH_METHOD=plain
SMTP_AUTH_USER=YourUsername
SMTP_AUTH_PASS=YourPassword

Emails are sent from noreply@rumble.run by default, but this can be changed by setting the FROM_EMAIL option:

FROM_EMAIL=rumble@yourcompany.int

Hostname and port

The RUMBLE_CONSOLE variable is used for creating inbound links, configuring deployed explorers, and generating the default self-signed TLS certificate. This setting is how both users and deployed explorers connect to the platform. Changing this setting may require regeneration of the TLS certificate and redeployment of explorers.

RUMBLE_CONSOLE=https://{IP ADDRESS OR HOSTNAME}:443

Rumble can be configured to run on a different port with the CONSOLE_PORT setting. This port defines where the console listens, but users and explorers still connect to the RUMBLE_CONSOLE value. In most cases this should match the port specified by the RUMBLE_CONSOLE.

CONSOLE_PORT=443

TLS configuration

Rumble will generate a self-signed TLS certificate and serve all web requests using HTTP over TLS. The standard configuration uses a self-signed certificate stored in the filesystem:

TLS=true
TLS_CERT=/etc/rumble/certs/cert.pem
TLS_KEY=/etc/rumble/certs/key.pem

The certificate and key file are PEM encoded and can be replaced with any valid certificate. Please ensure that any new certificate lists the value of RUMBLE_CONSOLE in the list of Subject Alternative Names.
If a TLS-terminating reverse proxy is used (AWS ELB, nginx, etc), TLS can be disabled at the application level:

TLS=false

Please note that while the web interface can be accessed over plain HTTP in this scenario, explorers will refuse to connect to a plain HTTP port, and features like WebAuthn MFA will not work unless the site is accessed through TLS.

Specific TLS versions and ciphers can be configured.

TLS versions are chosen by minimum and maximum:

TLS_VERSION_MIN=1.2
TLS_VERSION_MAX=1.3

TLS ciphers may be chosen by profile name:

- **default**: A set of strong ciphers, great for most configurations
- **nist80052**: A set of strong ciphers, approved in **NIST 800-52r2**.
- **nist80052-aes256**: A set of strong ciphers, approved in **NIST 800-52r2**, restricted to AES-256 variants

Please note that TLS 1.3 ciphers work differently and if a specific set of ciphers is required, both TLS_VERSION_MIN and TLS_VERSION_MAX should be set to 1.2. For example, to restrict Rumble to **only** NIST 800-52r2 approved ciphers using AES-256, the following configuration should be used:

TLS_VERSION_MIN=1.2
TLS_VERSION_MAX=1.2
TLS_CIPHERS=nist80052-aes256

TLS ciphers may also be chosen using comma-separated list of **cipher constants**.

### Database configuration

Rumble uses a PostgreSQL database for all platform data, except for raw scan files, change reports, and images processed from scans. By default, Rumble will configure a local PostgreSQL server on the same system, with a random password, and without TLS encryption:

DATABASE_URL=postgres://rumble:{DB_PASSWORD}@127.0.0.1:5432/rumble?sslmode=disable

If separate database is preferred, any PostgreSQL server running 12.x or newer should work. TLS (sslmode=require) should be enabled when a non-local database server is configured.
The default database pool (connection count) can be modified for high throughput environments:

DATABASE_POOL_COUNT=50

**Proxy configuration**

Rumble makes outbound connections to receive platform updates (in online mode), to connect to third-party APIs, and to delivery webhooks for notifications. If a proxy server is required, it can be configured with this setting:

HTTPS_PROXY=host:port

**Storage configuration**

Rumble uses local file storage to store raw scan data, change reports, and images retrieved from assets. This storage directory must be owned by the rumble user and be mounted below the /opt/rumble path.

RUMBLE_STORAGE_MODE=local
RUMBLE_STORAGE_PATH=/opt/rumble/storage

Files within the storage directory are split up into two groups, assets and scans. The names of these can be changed by setting:

ASSET_BUCKET=assets
SCAN_BUCKET=scans

To use AWS S3 for file storage instead, the following configuration can be set:

RUMBLE_STORAGE_MODE=s3
ASSET_BUCKET=company-rumble-assets
SCAN_BUCKET=company-rumble-scans

If S3 is used, AWS must also be configured.

**Secret configuration**

Rumble uses three randomly generated secret tokens to secure the platform. These keys are hexadecimal strings generated by 16 bytes of random. Compatible values can be generated by OpenSSL:

$ openssl rand -hex 16

The authentication key used for local storage HMAC operations. This key can be rotated as long as the service is restarted afterwards:

RUMBLE_STORAGE_KEY_SECRET={SECRET_32_HEX}
The session secret key is used to sign and validate browser session cookies. This key can be rotated, but doing so will invalidate all existing web logins:

```
SESSION_SECRET={SECRET_32_HEX}
```

The DB key is used for encryption of sensitive fields (user password bcrypt hashes). This key cannot be rotated, as password authentication will no longer work. If this key is changed, users must reset their password from the command-line or web interface using email before they can sign back in:

```
DB_KEY={SECRET_32_HEX}
```

**AWS configuration**

The AWS region is required:

```
AWS_REGION=us-east-1
```

The Access Key ID and Secret must be valid and correlated to a user with read-write access to the S3 buckets and read-only access to Secrets Manager.

```
AWS_ACCESS_KEY_ID=AKIA....
AWS_SECRET_ACCESS_KEY=SECRET....
```

AWS Secrets Manager can be used to retrieve almost any configuration setting at startup. The Secrets Manager entries should match the key names of the configuration file. The secret name can be defined with:

```
AWS_SECRETS_MANAGER_KEY=rumble/production
```

The location of the explorer and scanner binaries can be changed with these settings. Note that these should still live under /opt/ rumble or the service will not be able to load them:

```
RUMBLE_RELEASE_DIR=/opt/rumble/agent/
RUMBLE_SCANNER_RELEASE_DIR=/opt/rumble/scanner/
```

**Content Security Policy**

In the case of a non-standard S3 configuration (or S3-like deployment, such as minio), the Content Security Policy headers need to be configured to allow external image loads.

The CSP IMAGES setting can be used to specify one or more (comma-delimited) external image sources:

```
CSP_IMAGES=https://*.custom-storage-backend.lan
```
In additional to CSP_IMAGES, the following additional CSP settings are available:

CSP_SCRIPTS=https://*.myscripts.lan
CSP_FONTS=https://*.myfonts.lan
CSP_STYLES=https://*.mystyles.lan

**Permissions**

The self-hosted platform requires **root** access to install and manage from the command-line.

The platform service (rumble-console) runs as **root** and spawns a worker subprocess that runs as the **rumble** user account inside of a chroot environment (/opt/rumble). All substantive work happens within this isolated subprocess.

The following filesystem locations are used by the self-hosted platform:

### /etc/rumble

<table>
<thead>
<tr>
<th>Path</th>
<th>Owner</th>
<th>Permission</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/rumble</td>
<td>root</td>
<td>0700</td>
<td>Configuration files and certificates</td>
</tr>
<tr>
<td>/etc/rumble/config</td>
<td>root</td>
<td>0600</td>
<td>A plain-text configuration file</td>
</tr>
<tr>
<td>/etc/rumble/certs</td>
<td>root</td>
<td>0700</td>
<td>A directory containing the TLS certificate and key</td>
</tr>
<tr>
<td>/etc/rumble/certs/cert.pm</td>
<td>root</td>
<td>0600</td>
<td>The TLS certificate in PEM format</td>
</tr>
<tr>
<td>/etc/rumble/certs/key.pm</td>
<td>root</td>
<td>0600</td>
<td>The TLS certificate private key in PEM format</td>
</tr>
</tbody>
</table>

### /opt/rumble

<table>
<thead>
<tr>
<th>Path</th>
<th>Owner</th>
<th>Permission</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>/opt/rumble/tmp</td>
<td>rumble</td>
<td>0755</td>
<td>A temporary directory</td>
</tr>
<tr>
<td>/opt/rumble/storage</td>
<td>rumble</td>
<td>0700</td>
<td>Contains asset and scan artifacts</td>
</tr>
<tr>
<td>/opt/rumble/console</td>
<td>root</td>
<td>0755</td>
<td>Contains the platform executable</td>
</tr>
<tr>
<td>/opt/rumble/console/rumble-console.bin</td>
<td>root</td>
<td>0755</td>
<td>The platform executable</td>
</tr>
<tr>
<td>/opt/rumble/agent</td>
<td>root</td>
<td>0755</td>
<td>Contains the Explorer binaries</td>
</tr>
<tr>
<td>/opt/rumble/agent/rumble-agent-*</td>
<td>root</td>
<td>0755</td>
<td>The Explorer binaries</td>
</tr>
<tr>
<td>/opt/rumble/scanner</td>
<td>root</td>
<td>0755</td>
<td>Contains the Scanner binaries</td>
</tr>
<tr>
<td>/opt/rumble/agent/rumble-scanner-*</td>
<td>root</td>
<td>0755</td>
<td>The Scanner binaries</td>
</tr>
<tr>
<td>/opt/rumble/proc</td>
<td>root</td>
<td>0755</td>
<td>Contains copies of system /proc files</td>
</tr>
<tr>
<td>/opt/rumble/proc/cpuinfo</td>
<td>root</td>
<td>0644</td>
<td>A copy of /proc/cpuinfo</td>
</tr>
<tr>
<td>Directory</td>
<td>Owner</td>
<td>Permissions</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>---------</td>
<td>-------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>/opt/rumble/proc/meminfo</td>
<td>root</td>
<td>0644</td>
<td>A copy of /proc/meminfo</td>
</tr>
<tr>
<td>/opt/rumble/proc/version</td>
<td>root</td>
<td>0644</td>
<td>A copy of /proc/version</td>
</tr>
<tr>
<td>/opt/rumble/etc</td>
<td>root</td>
<td>0755</td>
<td>Contains copies of system files</td>
</tr>
<tr>
<td>/opt/rumble/etc/resolv.conf</td>
<td>root</td>
<td>0644</td>
<td>A copy of /etc/resolv.conf</td>
</tr>
<tr>
<td>/opt/rumble/etc/ca-certificates.crt</td>
<td>root</td>
<td>0644</td>
<td>A copy of the system root CA store</td>
</tr>
<tr>
<td>/opt/rumble/etc/rumble</td>
<td>rumble</td>
<td>0700</td>
<td>Contains instance identifiers</td>
</tr>
<tr>
<td>/opt/rumble/etc/rumble/cruncher.id</td>
<td>rumble</td>
<td>0700</td>
<td>A unique ID to identify the cruncher instance</td>
</tr>
<tr>
<td>/opt/rumble/etc/rumble/hub.id</td>
<td>rumble</td>
<td>0700</td>
<td>A unique ID to identify the hub instance</td>
</tr>
<tr>
<td>/opt/rumble/config</td>
<td>root</td>
<td>0700</td>
<td>Unused today</td>
</tr>
</tbody>
</table>
Rumble binary verification

Rumble uses dynamically generated binaries for the Rumble Scanner and Rumble Explorer downloads. Although Windows binaries have a valid Authenticode signature, all binaries also contain a secondary, internal signature. Dynamic binaries make it easy to deploy explorers that connect back to the right organization, but present a challenge for independent integrity validation. To enable verification of the internal signature, we offer the Rumble Verifier. This verification tool can confirm whether a given binary contains a valid internal signature, in addition to any existing Authenticode signatures.

To get started, download the latest version of the verifier from the bottom of this page along with the PGP signature file for the selected architecture.

The Rumble Verifier is always signed by PGP Key ID AE96EC3E8E1F27C6.

To validate the signature of the Rumble Verifier for Windows 64-bit, you will need a GPG client and to run the following commands:

C:\> curl https://keybase.io/hdm/pgp_keys.asc | gpg --import
C:\> gpg --verify rumble-verifier-2.0.2-windows-amd64.exe.asc

Successful validation will show a valid signature by key ID CEC20C193A94F31CE670C668AE96EC3E8E1F27C6.

```
gpg: Signature made Wed 05 Jun 2019 06:39:03 PM EDT
gpg: using RSA key CEC20C193A94F31CE670C668AE96EC3E8E1F27C6
```

The warning below is expected and does not indicate a problem with the signature:

```
gpg: WARNING: This key is not certified with a trusted signature!
gpg: There is no indication that the signature belongs to the owner.
```

Once the Rumble Verifier itself has been validated, it can be used to check the signature of any Rumble binary:

```
C:\> rumble-verifier-2.0.2-windows-amd64.exe rumble-explorer-0.5.30-windows-amd64.exe
rumble-explorer-1.0.0-windows-amd64.exe: VALID SIGNATURE
```

A failed validation will show the error Invalid or missing signature and the verifier will set exit status to 1.

**Binary downloads**

**Windows**

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
</table>

43
### Linux

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumble Verifier x86 64-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier x86 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
</tbody>
</table>

#### Additional Linux builds

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumble Verifier ARM v5 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM v6 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM v7 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM 64-bit (aarch64)</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier PPC 64-bit Little Endian</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier MIPS 32-bit Big Endian</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier MIPS 32-bit Little Endian</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier MIPS 64-bit Big Endian</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier MIPS 64-bit Little Endian</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier S390X</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
</tbody>
</table>

### MacOS

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumble Verifier x86 64-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM 64-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
</tbody>
</table>

### BSD Variants

#### FreeBSD

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
</table>

### Rumble Verifier

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumble Verifier x86 64-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier x86 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM v6 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM v7 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
</tbody>
</table>

### NetBSD

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumble Verifier x86 64-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier x86 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM v5 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM v6 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
<tr>
<td>Rumble Verifier ARM v7 32-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
</tbody>
</table>

### Dragonfly

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumble Verifier 64-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
</tbody>
</table>

### OpenBSD

<table>
<thead>
<tr>
<th>Build</th>
<th>PGP sig</th>
<th>SHA hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rumble Verifier 64-bit</td>
<td>pgp signature</td>
<td>sha-256</td>
</tr>
</tbody>
</table>
Managing your team

Rumble supports multiple concurrent users with a variety of roles. To add a team member, access the Your Team page, and use the Add Team Member button to send an invitation.

The team view has three tabs. The first tab, Team members, shows users who have access to all organizations. The second, Restricted users, lists users who only have access to a single organization. The third tab lists users who have access to the current organization, as selected from the organization selector at the top of the screen.

Single sign-on (SSO)

If you use a SAML2-compatible single sign on (SSO) implementation, the SSO Settings page can be used to configure an Identity Provider (IdP) and allow permitted users to login to the Rumble console.

Rumble’s SSO implementation is designed to work with common SAML providers with minimal configuration requirements, but it has a few requirements:

- Your SAML IdP should provide something that looks like an email address in the NameID parameter. It doesn’t need to be a valid email address, but it should be a unique value that has the same syntax as an email address (user@example.com).
- If the NameID does not look like an email address, Rumble will check the fields email, user.email, emailaddress and email address for a suitable ID.
- Your users need to authenticate to a single domain such as example.com, not to multiple domains or a domain with many subdomains.
- Rumble will check for the user’s full name in the fields name, gecos, user.name and displayname. If no full name field is found, Rumble will proceed to check for a first name in first_name, firstname, given_name, user.firstname, givenname or first name; and for a last name in last_name, lastname, family_name, user.lastname, surname, sn, or last name. These attributes are case insensitive.

Global roles

Rumble allows roles to be defined per-user at both the global and organization level. The standard roles are admin, user, viewer, and billing. There is also a superuser role available to manage global settings.
**Superusers**

The first user created within the Rumble console is considered a superuser. This role allows management of global settings like subscriptions and SSO parameters, and is shown as an access level of “everything”.

If you are a superuser, you can promote someone else to be a superuser. To do this, check the row listing them, and click the *Promote to superuser* button.

If you are using SSO authentication, you should configure at least one superuser with a strong password and MFA that can used as a backup if SSO settings need to be changed in the future.

We strongly recommend having more than one superuser, particularly if you are using MFA. That way if an MFA token is lost or a superuser leaves your organization, another superuser can fix the problem.

**Administrators**

Administrators can modify any aspect of an organization and have the unique ability to permanently delete bulk data, create additional organizations, and reset settings for other users.

**Users**

Users have full access to an organization and can update sites, modify assets, schedule scans, and generally use most functionality. Users are not permitted to reset other user’s security credentials, bulk delete data, or delete an organization.

**Annotators**

Annotators have the same permissions as a viewer, except they have the ability to add tags to assets. Annotators do not have any other write-access within an organization, so they are unable to modify or remove existing tags. Modifications to existing tags must be made by a Rumble user or administrator.

**Viewers**

Viewers have read-only access to an organization. This includes all inventory data, all reports, and all task configurations. Viewers are not allowed to interact with tasks, modify settings, or update assets. Viewers may not download the command-line Rumble Scanner and install Rumble Explorers, and they do not have access to view API tokens or export tokens.
Billing

Billing users are unable to see any asset data, but can manage the licensing, billing, and entity settings for the account.

No Access

Accounts with no access, which is set in the global role, are limited to those organizations where they have been granted access. If no organizations are allowed, the user is limited to managing their own account settings.

These accounts can only see other users that share their authorized organizations. The no access global role can be used to create a single-organization user, such as a customer or third-party that needs access to the inventory for a specific organization. For consulting use cases, a single-organization user is a way to provide clients with visibility into their environment at no additional cost.

Account settings

The Account page is available to superusers. It contains settings which apply to all users and organizations within the account.

Multi-factor authentication (MFA)

Rumble supports multi-factor authentication, also known as two-factor authentication or 2FA. Physical hardware keys such as Google TitanKey and Yubico YubiKey are supported via the WebAuthn standard.

You can configure MFA policies for your account via the Account settings page. If multi-factor authentication is required, users who do not have an MFA token set up will be required to set one up when they next log in. You can choose between requiring this for all users, or only requiring it for non-SSO users. The latter option is useful if your SSO server enforces MFA use.

Once a user registers one or more MFA tokens, they will be required to use one of the tokens every time they log in.

Note that changing the account settings to not require MFA will not alter the MFA status of existing accounts. Existing accounts will keep any existing MFA tokens they have registered, and will still be required to use one to log in. To disable MFA for a user, the user must clear the MFA token registration. To do this, they can go to their user settings page and click the red “Unlink“ text next to the token ID in the bottom right.
Disabling support access

If you check the box labeled *Disable support access to your account*, Rumble support staff will not be allowed to switch to your account.

If you choose to disable support access, this may make it harder for Rumble support to answer any questions you have. In some cases we may need you to turn support access back on so that we can help you.

Idle times and login duration

You can set a maximum idle session time in minutes. If set, users whose web browsers don’t access Rumble for the specified time period will be considered idle, and logged out.

You can also specify a maximum login duration. If set, users will be forced to log in again regularly, at least once every specified period.

Account API keys

The Account API is a REST API which allows account-level operations such as adding and removing organizations and sites, adding users, and accessing the system event log. The *Generate API Key* button on the Account page can be used to generate a token which will allow access to the Account API.

License information

The *License* tab shows information about your Rumble software license, including how many assets you are licensed for, how many assets you have across all organizations, and when your license renews.

Entity information

The *Entity* tab allows you to update information about the legal entity Rumble is licensed to. You should ensure that this information is kept up-to-date if your company changes name or location, as we use the information to calculate taxes and ensure compliance with appropriate regulations.
Bulk importing users

Instead of manually adding users one at a time, Rumble administrators can add multiple users via bulk import. To bulk import users, you will need to create a CSV (comma separated values) file that contains the user information, such as their first name, last name, email, role, and organizational access.

Bulk imports will only add new users; it will not update existing users. If the file contains a user that already exists in the system, the import will not complete. You’ll need to remove all duplicate users from your CSV file and import the file again.

Here are the high-level steps for bulk-importing users:

1. Create a CSV file with your user information.
2. Import the CSV file into Rumble.
3. Verify users have registered their accounts.

Creating the CSV file for importing users

To create the CSV file, you can use the provided template or create a file from scratch. You’ll need to follow the guidelines and include the fields outlined below.

- The first row of the CSV file is the header row and it must be included in the file. It contains the required fields needed for importing users, and those fields must follow a specific order (first_name, last_name, email, all_orgs_role, org_roles).
- Rumble requires the first_name, last_name, and email fields. The organization-level fields, all_orgs_role and org_roles, are optional.
- To define the org_roles field, use the following format: orgName=role. You can use pipes to separate multiple definitions.
- Excluding the all_orgs_role field from the CSV will result in users being assigned a no-access role in all organizations.
- Excluding the orgs_role field from the CSV will result in users being assigned the role defined in the all_orgs_role field. If both fields are excluded from the file, all users will be assigned the no-access role. They will need to contact the Rumble administrator to modify their account access.

User fields for the CSV file

The following table lists the fields you can include in your CSV file:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>first_name</td>
<td>The user’s first name.</td>
<td>sarah</td>
</tr>
<tr>
<td>last_name</td>
<td>The user’s last name.</td>
<td>smith</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------</td>
<td>-------</td>
</tr>
<tr>
<td>email</td>
<td>The user’s email address.</td>
<td><a href="mailto:ssmith@company.org">ssmith@company.org</a></td>
</tr>
<tr>
<td>all_orgs_role</td>
<td>The role assigned to the user for all organizations. This is optional.</td>
<td>user</td>
</tr>
<tr>
<td>org_roles</td>
<td>The role assigned to the user for a specific site. Use pipes to separate the organizations. This field is optional.</td>
<td>org1=user</td>
</tr>
</tbody>
</table>

Your resulting CSV file should look like the following example:

```
first_name, last_name, email, all_orgs_role, orgs_role
Sarah, Smith, ssmith@company.org, user, org1=annotator|org2=annotator
```

**Importing users into Rumble**

1. Go to your Team Import page.
2. Choose the CSV file that contains your user information and upload it to Rumble.
3. Import the file.

Rumble will alert you if there are issues with your file. For example, if the file contains an existing user, the import will not complete. You’ll need to remove all duplicate users from your file and import it again.

**Verifying users have registered**

After you add new users via import, each user receives an email invitation to join your team. The email contains a link to register their email address, which prompts them to enter their name and password for their account. After they create an account, they’ll be able to log in to the Rumble console.

You can visit the Teams page to find the status and last login for each user. Users who have completed the registration process will show an Activated status.
Setting up Azure AD SSO

Superusers can configure single sign-on to the Rumble Console using an external identity provider (IdP), which enables authentication and user access control to the Rumble Console from your single sign-on (SSO) solution. By default, Rumble has SSO functionality available, but it's not a requirement to log in to the console. You can make it a requirement or disable it completely.

Here are the high-level steps to set up SSO using Azure AD to authenticate and manage user access to Rumble:

1. Add and configure Rumble as an Azure AD app.
2. Download the SSO configuration metadata in XML format.
3. Set up SSO in Rumble.
4. Add users to your Rumble app in Azure AD.

Requirements

Before you can set up SSO for Azure AD:

- Verify that you have administrator privileges for Azure AD.
- Verify that you are a superuser in Rumble. Look for the yellow star in your account status.

Step 1: Add and configure Rumble as an Azure app

The first thing you need to do is add Rumble as a non-gallery application to your Azure AD setup and to configure the settings for Rumble as an Azure AD application.

1. In Azure, go to Enterprise Applications > New Application > Create your own application.
2. Under the What are you looking to do with your application? section, choose the Non-gallery application option.
3. Name your application something like Rumble, and then add it.
4. Go to Azure Active Directory > Enterprise applications and open the newly created Rumble application.
5. Select the Single sign-on tab, and then choose SAML as the sign-on method.
6. For the fields on the Configure App Settings page, go to https://console.rumble.run/team/sso/sp and copy the necessary service provider details:
   - Entity ID
   - Single sign-on URL
   - SSO callback (ACS) URL
7. Enter the values into the relevant fields in the Azure AD portal.
Step 2: Download the SSO configuration metadata

While editing your application settings, you can get the download link to obtain the SSO configuration metadata in XML. You’ll need this information to set up SSO in Rumble.

1. On the Configure App Settings page, find the SAML Signing Certificate section.
2. Locate the XML download link under the Federation Metadata URL.
3. Download the file. You’ll need the contents of this file for the next step.

Step 3: Set up Azure AD SSO in Rumble

Now that you have the SSO configuration metadata in XML, you can configure Azure AD SSO settings in Rumble.

1. Go to https://console.rumble.run/team/sso/idp to access the SSO IdP provider settings page in Rumble.
2. Choose one of the following modes to enable SSO:
   - **Allowed** - Enables SSO, but users still have the option to login without SSO.
   - **Required** - Requires users to log in with SSO. Only superusers can log in without SSO.
3. Enter the domain name that is associated with SSO authentication. This is likely your company domain (companyabc.com).
4. Choose a default role for SSO users. This is the role all new users will be assigned when their account is created.
5. Copy the XML you downloaded from Azure and paste it into the Metadata XML field on the Rumble SSO IdP page.
6. Apply your SSO settings. The remaining IdP fields will auto-configure for you.
   - The issuer URL will look something like https://sts.windows.net/00000000-0000-0000-0000-000000000000/ where the UUID at the end is your unique Microsoft Active Directory (tenant) ID, listed under **App registrations > Overview > Endpoints**.
   - The login URL will be something like https://login.microsoftonline.com/00000000-0000-0000-0000-000000000000/saml2 with the zero UUID replaced with your unique tenant ID.
   - The certificate will be Microsoft’s PEM encoded certificate, which will be extracted automatically from the XML.
   - On the Microsoft side, the redirection URL for Rumble should be https://console.rumble.run/auth/<domain>/saml20/process, where <domain> is replaced with the domain specified in the Rumble SSO settings.
Step 4: Add users to the Rumble app in Azure

Now that you've completed the set up, you can go to the Rumble app in Azure portal to add users and assign their access. Any users you add to the Rumble app will be viewable from the Team members page in Rumble, once they have logged into Rumble.
Setting up Okta SSO

Superusers can configure single sign-on to the Rumble Console using an external SAML identity provider (IdP), such as Okta, which enables authentication and user access control to the Rumble Console without typing in credentials.

Here are the high-level steps to set up single sign-on (SSO) using Okta to authenticate and manage user access to Rumble:

- Add Rumble as an application in Okta.
- Set up SSO in Rumble.
- Add users to the Rumble app in Okta.

Requirements

Before you can set up Okta SAML:

- Verify that you have administrator privileges for Okta.
- Verify that you are a superuser in Rumble. Look for the yellow star in your account status.

Step 1: Add and configure Rumble as an Okta app

1. Go to Okta > Applications > Create App Integration. When the Create a new app integration window appears, select SAML 2.0 for your sign-in method.

   - OIDC - OpenID Connect
     Token-based OAuth 2.0 authentication for Single Sign-On (SSO) through API endpoints. Recommended if you intend to build a custom app integration with the Okta Sign-In Widget.

   - SAML 2.0
     XML-based open standard for SSO. Use if the Identity Provider for your application only supports SAML.

   - SWA - Secure Web Authentication
     Okta-specific SSO method. Use if your application doesn’t support OIDC or SAML.

   - API Services
     Interact with Okta APIs using the scoped OAuth 2.0 access tokens for machine-to-machine authentication.

2. For the general settings, you’ll need to provide a name for the app. Call the app Rumble. You can also add a logo to make it easier for users to identify the Rumble
3. For the SAML settings, you'll need to go to the service provider information page in Rumble to find the relevant URLs.
   - **Single sign on URL** - In Rumble, this is the assertion consumer service (ACS) URL.
   - **Audience URI or SIP Entity ID** - In Rumble, the entity ID, or SAML audience, will be `https://console.rumble.run`.
     - **NOTE**: make sure to include the leading `https://` when entering this field in Okta.
4. For the remaining settings, like the attribute statements, visit the Okta documentation to learn how to configure them.
5. When you finish configuring the SAML settings, Okta will prompt you for some feedback on how you will be using the app. You can opt to provide feedback or skip to complete the set up.
6. After Okta creates the app, you will need to view the SAML 2.0 instructions to complete the set up. Go to the the Sign On tab for the Rumble app and view the SAML 2.0 instructions. You'll need these details for the next step.
Step 2: Set up SSO in Rumble

1. Go to the **SSO setup page** in Rumble. Choose one of the following modes to enable SSO:
   - **Allowed** - Enables SSO, but users still have the option to login without SSO.
   - **Required** - Requires users to log in with SSO. Only superusers can log in without SSO.
2. Enter the domain name that is associated with SSO authentication. This is likely your company domain (companyabc.com).
3. Choose a default role for SSO users. This is the role all new users will be assigned when their account is created.
4. Copy the fields from Okta into Rumble.
   - **Issuer URL** - In this field, enter the Identity Provider Issuer URL from Okta. This will look something like http://www.okta.com/<ID>
   - **Login URL** - In this field, enter the Identity Provider Single Sign-On URL from Okta. This will look something like http://<okta-instance>/app/<app-name>/<ID>/sso/saml.
   - **Certificate** - Copy and paste the entire contents of the X.509 certificate from Okta.
5. Apply your SSO settings.

Step 3: Add users to the Rumble App in Okta

Now that you've completed the set up, you can go to the Rumble app in Okta to add and manage user access. After you've completed this step, your users will be able to go to your SSO sign-in URL to access Rumble.
Organizations and sites

Rumble uses the concept of organizations and sites to manage information within your account.

Organizations

An organization represents a distinct entity; this can be your business, a specific department within your business, or one of your customers. All actions, tasks, explorers, scans, and other objects managed by Rumble are tied to specific organizations and isolated from each other.

Your active organization can be switched by using the dropdown select field on the top right of the Rumble Console.

Creating organizations

Organizations can be created, modified, and deactivated by going to the organizations section within the console. Click the Organizations button under Global Settings on the left-hand sidebar.

Projects

A project is a special type of organization designed for temporary use. They behave like organizations, and can have sites defined within them. The important difference is that projects automatically become read only after 30 days, and are automatically deleted after 90 days. While organizations allow you to customize expiration times, the expiration times for projects are fixed.

Your Rumble license allows you to have five times as many assets in projects as in regular organizations.
If you decide that you want to keep a project indefinitely, it can be converted into an organization.

Note that organizations support custom data retention settings that you can edit to your requirements.

**Sites**

Every organization has at least one site, but may have multiple sites. A site represents a distinct network segment, usually defined by addressing or accessibility. Sites in Rumble do not necessarily correspond to physical sites or locations.

All analysis actions within Rumble occur at the site level. This allows for multiple sites to use the same RFC1918 space, something common in retail, while still being possible to differentiate their assets within the Inventory.

For flat networks, where every IP address can reach any other address on the network, a single site is usually enough, and avoids the possibility of accidentally creating duplicate assets by scanning the same devices from multiple sites. Sites are recommended for complicated, sprawling, and highly-segmented environments.

Sites can be tied to specific explorers, which can help limit traffic between low-bandwidth segments. The site configuration allows a default scan scope to be defined, along with an optional list of excluded scan scopes. These fields can be used to set the scan scope for scans of the site.

By default, your account includes a single organization, which itself contains a single site, named Primary. If the only site in an organization is deleted, a replacement will be created automatically. Similarly, if the last organization is removed, a replacement will be created. You can rename organizations and sites at any time.
Discovering assets

A discovery scan finds, identifies, and builds an inventory of all the connected devices and assets on your internal network. Running a discovery scan routinely will help you keep track of and know exactly what is on your network.

Discovery scans are configured by site, explorer, and scope. In order to run a scan against a specific site, an explorer must be activated and either assigned to that site or configured for all sites.

When creating a new scan, you have multiple parameters you can set, ranging from scheduling a date to more advanced options. To launch a discovery scan, browse to the Inventory page and click the New Scan button in the upper right.

Site

Rumble organizes information into organizations and sites. Organizations are distinct entities that are useful for keeping data separate and contain a collection of sites. Sites are used to model segmented networks, particularly independent networks which use the same private IP address ranges.

For example, you might have multiple physical locations with their own local networks, all using the 10.0.0.0/8 private IP range. By defining them as sites, you can set up an explorer for each, and the networks and assets will be treated as completely independent even if similar systems are seen at the same IP addresses in each.

Since scan analysis occurs at the site level, the boundaries you define for a site set the default scope for scans for that site.

Explorer

Select the explorer to run the scan from, chosen from the set of registered explorers for the site. The explorer you choose must be able to directly communicate with the networks and addresses you define for the discovery scope.

The chosen explorer should ideally be able to reach all addresses in the scope directly, without a firewall in the way. Stateful firewalls and VPN gateways may interfere with the discovery process.
Discovery scope

The discover scope defines what IP ranges will be scanned. The scope uses the Site settings when specified as “defaults”, but may be changed on a per scan basis as well. The scope should include at least one IP address or hostname. IP address ranges can be specified in most standard formats:

- 10.0.0.1
- 10.0.0.0/24
- 10.0.0.0/255.255.255.0
- 10.0.0.1-10.0.0.255

Hostnames specified in the scope will be resolved at runtime by the assigned explorer. If the hostname returns multiple IP addresses, all addresses in the response will be scanned. Hostnames can also have masks applied, indicating that the mask should expand to each resolved address of the hostname. For example, if example.com resolves to both 1.2.3.4 and 5.6.7.8, the input of example.com/24 would become 1.2.3.0/24 and 5.6.7.0/24.

Scan name

You can assign a name to your Scan task to make it easier to keep track of.

Scan speed

Specify the maximum packet rate for the overall discovery process, in network packets per second. 500 is conservative, 3000 works for most LANs including WiFi, 10000 or more may be helpful for large sites with fast connectivity.

Schedule

You can set a date and frequency for your scan task. Dates and times take into account your browser’s advertised timezone.

Scans scheduled to start in the past will be launched immediately and then repeated at the specified time based at the frequency selected.

Scheduling grace period

Specify the number of hours to wait for an available explorer before giving up on this scan. A zero or negative value will result in the scan retrying indefinitely until an explorer becomes available.
Advanced scan options

The Advanced tab can be used to display and modify additional scan settings, such as network exclusions, scan speed, the ports covered by the TCP scan, and which probes are enabled. The default settings should work for most organizations but may need to be tweaked for slow networks or unreliable links.

Maximum host rate

As well as setting an overall scan rate in packets per second, you can also control the maximum rate at which packets are sent to any single host IP address. This is useful when you have devices which are easily overloaded by network traffic. The default should be safe for most systems.

Max group size

When Rumble scans your network, it spreads the scan load across many IP addresses at once. The max group size determines how many IP addresses can be actively being scanned at once – allowing for the fact that hosts may take some time to respond to probes. The max group size needs to be at least as large as the overall scan speed, or else it would limit the speed of the scan to below the set value. If you provide a value that's lower than the overall scan speed, it will be increased automatically at scan time.

The max group size is mostly useful when dealing with stateful network devices that can only track a limited number of connections at once, as a way to restrict how many active TCP sessions will result from a Rumble scan.

Max TTL

The IPv4 standards define the Time To Live or TTL as a time in seconds. However, the standards also specify that every device processing a packet must decrease the TTL by at least one, no matter how quickly it is processed. The TTL value therefore serves two purposes: it determines the maximum time a connection attempt can take to be processed, and also the maximum number of network hops it can pass through. The TTL value can be used to limit a Rumble scan from traversing distant networks.

TCP ports

The Included TCP ports and Excluded TCP ports fields can be used to override the default scan ports. The string “defaults” will lookup the current default port list at scan time. The current port list is:
Prescan modes for large IP spaces

Sometimes, the scope of your IP space is unknown, subnet usage is unknown, and the total number of assets is unknown. These unknowns can make it challenging to optimize your discovery scans for efficiency and speed. And when your IP space is large, like a /16 space with a few thousand IPs in use, a full discovery scan can take more time to complete, since it looks at more than 500 TCP ports and 15 UDP ports on every address. In these types of cases, you may want to tune your scan settings toprefilter ranges and IP addresses before a full scan.

Rumble has two prescan modes that you can use to run a faster scan: subnet sampling and host ping.

Subnet sampling

To speed up scans of large subnets you can use the “**Only scan subnets with active hosts**” advanced scan option. If this option is on, a prescan runs against the target space to identify the subnets with an active host. This mode leverages heuristics Rumble has collected to identify addresses that are more likely to be responsive across subnets. This process allows Rumble to quickly scan larger spaces by identifying the subnets that are in use, before starting full probes. All subnets that are identified as having active hosts are then fully scanned – unless you enable host pings.

There are two tweakable parameters for subnet sampling. The sample rate determines what percentage of addresses in each subnet are prescanned to determine if the subnet should be scanned. The subnet size determines how many IP addresses are in each subnet. By default, the subnet size is 256 addresses, corresponding to a /24 subnet, and 3% of the addresses in each subnet are prescanned.

Host ping

After you have some insights on the subnets that are in use, you may want to limit the full scan to only addresses that respond to the most common ping methods, such as ICMP and some TCP and UDP ports. If you choose the "**Limit scans to pingable hosts**" advanced scan option, only hosts that respond to pings request will be fully scanned. You can use this option in conjunction with subnet pings to accelerate your scan time.
Managing tasks

Discovery scans and other background actions are managed using the Tasks page of the Rumble Console. Standard tasks such as Scan and Import will be shown along with their current progress and summarized results in the Active and Completed task sections.

Tabs along the top of the tasks page allow you to view a subset of tasks – you can filter by tab to see active, scheduled, recurring, completed or failed tasks. The Search tab allows you to browse and search your complete task history.

Scheduled tasks are one-off actions that will be started in the future, while Recurring tasks are actions that re-occur on a regular basis. Recurring scan tasks generate a new standard task on each iteration of their schedule. Both Scheduled and Recurring scans will only launch if their associated explorer is online and no other scan tasks are running.

The Tasks page allows Scheduled and Recurring tasks to be removed and Active tasks to be stopped. Please note that stopping an active Scan task may take a few moments, as the status of the task is not updated until the explorer confirms that the scan was terminated.

Clicking on the Task type or the summary in the information column will present details of that task. For Scan and Import tasks, these details include the parameters used, summary of the Site changes made as a result, and a link to the change report JSON.
Using the inventory

The Inventory page is the heart of Rumble Network Discovery and the key to understanding what is on your network. The Inventory displays all assets within the Organization and can be sorted, filtered, and exported to obtain specific views of the environment.

Understanding assets

An asset within Rumble is defined as a unique network entity. Assets may have multiple IP addresses and MAC addresses and these addresses may change as the environment is updated. Rumble tracks assets based on several heuristics, including MAC address, IP address, hostnames, and fingerprint results for the operating system and running services.

In most cases, Rumble can accurately follow assets over time in environments using DHCP, even across remote subnets. For external networks, scans that are initiated with fully qualified hostnames will consolidate assets based on the hostname, which allows for consistent asset tracking for cloud-based external systems with dynamic IP addresses.

Within an organization, assets are isolated by site, and each site can have address space that overlaps with other sites. Sorting the Inventory view based on the site column can help in these scenarios, as can filtering the Inventory based on a specific site name.

The search field allows the Inventory to be filtered based on the specified criteria. Please see the Search Query Syntax documentation for specific details.

In addition to viewing assets, the Inventory page provides data export functionality, along with the ability to select assets, and specify the comments field. The Rescan action can be used to selectively rescan specific systems from the inventory, while the Remove Assets and Purge Assets can be used to permanently remove data from the inventory view.

The Reports button provides quick access to key reports from the Rumble reports page.

Loading assets

Data is loaded into the inventory using the Scan and Import buttons. The results are analyzed and merged, updating asset information as necessary.
The *Scan* button has two options: Standard Scan and Full RFC 1918 Discovery. The latter is an easy way to set up a fast scan of all private range IP addresses. You can then use the coverage reports to check for assets in unexpected private address ranges.

The *Import* button has two options. Importing Rumble Scan Data allows you to import data that was scanned by the standalone Rumble scanner. This means you can scan networks that have no connectivity to the Internet, and still view the results in the Rumble console. It’s also useful for reprocessing old scan data so that you can use the site compare feature to see how assets have changed over time.

The bulk asset update feature allows you to modify assets by exporting a CSV using the Export button, making changes to the data in a spreadsheet program, and then importing the result back into Rumble.

**Connecting to other systems**

The *Connect* button lets you connect Rumble to other systems. These include:

- Sources of cloud and VM inventory information
  - Amazon Web Services EC2
  - Microsoft Azure
  - VMware
- Endpoint detection and response systems
  - CrowdStrike Falcon
- Sources of Internet scan data
  - Censys Search API
- Mobile Device Management (MDM) systems
  - Miradore MDM

**Viewing services**

The inventory page has a second tab labeled *Services*. This flips the table of data from an asset-focused view to a service-focused view. For each asset, you will see one row for each service Rumble detected.

Like the asset tab, the services tab has a full search interface. You can filter services by protocol, port, and many other criteria, using the Rumble search language.
**Viewing screenshots**

If the Rumble Explorer has access to Google Chrome, it will attempt to take screenshots of web pages it finds while scanning your network. (This feature can be disabled in the scan options when setting up the scan.)

You can view the screenshots for all of your assets via the *Screenshots* tab, and click through to the asset records for full details.

**Viewing wireless networks**

If the machine running the Rumble Explorer has a working WiFi adapter, the explorer will attempt to scan for nearby wireless networks. The *Wireless* tab will show the results of the scan.
Understanding assets

Rumble treats assets as unique network entities from the perspective of the system running the explorer. An asset may have multiple IP addresses, MAC addresses, and hostnames and it may move around the network as these attributes are updated. Rumble tries hard to follow assets by correlating new scan data with the existing inventory, using multiple attributes.

An asset is always associated with a single site. If the same system happens to be covered by multiple sites, these will be treated as different assets, and will only be correlated against assets within their respective site. This separation by site allows the same network to be scanned from multiple perspectives and compared in a single view within the organization.

After each scan, all assets within the corresponding site are updated. If a system is identified that doesn't match an existing asset, a new asset will be created. If an asset is part of the site and it is not found during a scan, it will be marked as offline. If an asset is not correlated, due to substantial changes to the fingerprint (for example, a new network adapter was installed and the firewall was enabled), the previous asset will be marked as offline, and a new asset will be created to track the new configuration. This can lead to some level of duplication within a site, but these duplicates are usually marked as offline, and can be safely ignored or removed from the inventory by hand.

Asset fields

The following asset fields are available.

Addresses

Rumble will report at least one and often multiple IP addresses for a given asset. These addresses can encompass multiple network interfaces but will only be displayed as a primary address if that address was within the scan scope used to detect it.

Secondary addresses

Rumble may report one or more secondary addresses, based on network response probes. These are IP addresses that were detected on the asset but were not within the scan scope. Secondary address detection is critical when trying to identify systems that bridge networks that should be isolated.
Names

Rumble may report one or more hostnames. These names can be obtained from the initial DNS lookup (when hostnames are provided in the scan scope), from DNS PTR lookups during the scan, and by extracting names advertised within network probe responses.

Operating System (OS)

Rumble attempts to fingerprint, and failing that, guess at the operating system running on each asset. If limited information is available, this field may be empty.

Type

Rumble attempts to determine the general device type through analysis of fingerprints and running services.

Hardware

Rumble attempts to determine the physical (or virtual) hardware if enough information is present.

MAC addresses

Rumble may be able to enumerate one or more MAC addresses from the asset. MAC addresses are pulled from ARP if available, but also several network services that can return MAC address information across routed segments.

Services

Rumble tries to detect approximately 100 TCP services by default, along with several useful UDP services. These services are in addition to ARP and ICMP. The services field contains a list of the most recently recorded services for the asset.

Round Trip Time (RTT)

Rumble records the amount of time certain probes take in order to get a rough sense of the latency between the explorer and the asset.
Detected by

Rumble records which probe was used to identify an asset. For assets that are on remote subnets and have firewalls in place, this field indicates what service was used to obtain a response.

Alive status

Rumble tracks whether a given asset was found during the most recent scan where its site was in scope. If the asset was not found, it will be marked as offline until a following scan detects it again.

First seen

Rumble tracks the initial timestamp when an asset was first identified.

Last seen

Rumble tracks the last timestamp when an asset responded to a probe during a scan.
Exporting asset data

The **Inventory** view provides a few ways to (export asset data). The Export menu offers Export All options in both CSV and JSON format, and when a search query has been provided, options to export just the search results as both CSV and JSON.

The CSV format can be opened with tools like Microsoft Excel and easily imported into other applications but does not contain the full details of certain fields, such as Services. The JSON format contains a complete export but may take additional processing to use with other tools. If you have a specific export/import scenario in mind, please contact support and let us know. The Export API leverages both the CSV and JSON formats and supports arbitrary search queries in the same syntax as the Inventory.
Data retention

Rumble allows the data retention periods to be configured at the Organization level. The Organization settings page provides three ways to control how Rumble manages your asset and scan data. Data expiration is processed as a nightly batch job based on the current settings for each organization in your account.

<table>
<thead>
<tr>
<th>Stale asset expiration</th>
<th>Offline asset expiration</th>
<th>Scan data expiration</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>90</td>
<td>365</td>
</tr>
</tbody>
</table>

A non-zero value sets the number of days before stale assets are removed from the inventory. This applies to online and offline assets not seen within the configured number of days.

A non-zero value sets the number of days before offline assets are removed from the inventory. This applies to offline assets not seen within the configured number of days.

A non-zero value sets the number of days before completed scans are removed from the platform. This applies to the scan metadata as well as the raw data file change reports.

Stale asset expiration

This setting can be used to remove assets that have not been seen in the specified number of days. This correlates with the last_seen:<Xdays search query, where X is defined as the number of days since this asset was found to be responsive. This setting applies to both online and offline assets. The default setting of zero disables this feature. This feature is useful if your organization has a maximum data retention policy for cloud services.

Offline asset expiration

This setting can be used to remove offline assets that have not been seen in the specified number of days. This correlates with the last_seen:<Xdays AND alive:false search query, where X is defined as the number of days since this asset was last found to be responsive. This setting is useful when short DHCP leases or guest wireless networks result in many offline assets appearing in the inventory. The default setting for new organizations is 90 days, but this can be set to zero to disable automatic asset removal.

Scan data expiration

This setting controls how long Rumble will retain the raw data and metadata for Scan and Import tasks. If your organization has a maximum data retention policy for cloud services, this setting can be used to force the removal of any data older than a given number of days. The default setting of 365 preserves one year of historical scan data.
Credentials

The Credentials page provides a single place to store any secure credentials needed by Rumble. That includes:

- SNMPv3 credentials
- Access secrets for cloud services like AWS and Azure
- API keys for services such as Censys and Miradore

Credentials are stored in encrypted form in the Rumble database. Credentials which are used by the Rumble explorers, such as SNMP passwords, are transmitted to the explorers in encrypted form. For security reasons, the secret part of any credential cannot be viewed once entered.

When adding a credential, you can choose whether to make it a global credential which can be used for all organizations, or to allow access only by specific organizations.

The specific fields and options for a credential depend on the type of credential.

VMware and SNMP credentials, which are used by the Rumble Explorer, allow a CIDR allow list to be specified. This can be used to limit which scanned IP addresses the credential will be used with. This feature allows you to avoid sending SNMP or VMware credentials to all scanned hosts on the network, and instead limit them to specific IP addresses or ranges.
Exporting HP iLO data

HP Integrated Lights-Out (iLO) provides remote management, configuration, and monitoring capabilities for HP servers. These capabilities centralize operations for your server environments and streamline tasks like rebooting servers, booting into single user mode, and bypassing authentication.

Being able to identify and find the serial number for HP iLO devices is useful for tracking warranties for support and contract management. If you have Rumble Enterprise, you can export the Rumble data as a CSV to feed into warranty tracking tools.

How to export HP iLO CSV data in Rumble Enterprise

In your Rumble Console, go to your inventory. From the Export menu, choose the HP iLO CSV format. This method downloads all HP iLO data from the Rumble inventory to a CSV file.

If you want to refine the results in your exported data, you can filter the inventory first. For example, if you only want to export iLOs that have the ProLiant DL360p Gen8 hardware, you can query:

```
alive:t AND ilo.hardware:"=ProLiant DL360p Gen8"
```

And then, from the Export menu, choose the HP iLO CSV format from the Export Search Results submenu.

HP iLO CSV export data

When you export HP iLO data, the CSV file will contain the following fields:

- IP address
- MAC address
- Name
- Serial number
- Product ID
- Model
- Version
- Health status
- IRS
- iLO product name
- Serial number
- Boot block
- HW revision
- FW revision
• PWRM
• Auth local
• Auth Kerberos
• Auth LDAP
• Auth license
• Security message
• Alive
• ID
• Site ID
• Last seen

There is a lot of data in the export that you may not need for your warranty tracker. Fields, such as serial numbers, physical hardware information, health status, and firmware version may be the most useful to import into your tracker.
Enriching scans with EC2

As part of a discovery scan, Rumble will automatically enrich scanned assets with data from the AWS EC2 API when available. Rumble assets will be updated with internal IP addresses, external IP addresses, hostnames, MAC addresses, and tags, along with other EC2-specific attributes, such as the account ID and instance type.

No additional configuration is needed in Rumble to get this data enrichment. However, you may need to modify the permissions associated with the instance's IAM role.

Find explorers with EC2 enrichment capabilities

To use the EC2 enrichment capabilities, the Rumble Explorer must be running on an instance with permissions to describe your EC2 instance list. This can be configured through an IAM role associated with the instance as well as by configuring the AWS credentials for the root user account.

To identify the explorers with this capability, view your registered explorers. Any explorer with the cloud icon indicates that it can enumerate EC2 instances.

Scans run from these explorers will merge any EC2 instance fields into the asset automatically for any in-scope targets matched against the instance list.

Add permissions to describe instances

To allow for EC2 instance data enrichment, you will need to add the EC2 ec2:DescribeInstances permission for the instance role configured for your instance. Visit the Amazon docs to learn how to create and update policies.

Basically, your policy will look like:

```json
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "VisualEditor0",
    "Effect": "Allow",
    "Action": ["ec2:DescribeInstances"],
    "Resource": "*"
  }]
}
```

From the IAM UI, go to Roles > Permissions > Attach policies and search for the EC2 service. From the actions, select DescribeInstances, which is located under List.
You can also configure credentials on the instance by running `aws configure` as root, instead of using the IAM instance role.

After you save your policy, you'll need to restart your explorer. The easiest way to do this is to force an update from the Explorer menu.

Keep in mind if your configuration uses one region, but the instance is located in another, it will use the instance's region instead for all API requests.

**Attributes Rumble gets from the EC2 API**

When Rumble determines that an IP address is also an EC2 instance, it will enrich the existing Rumble data with EC2 metadata. Rumble will be able to pull in more hostnames based on AWS asset tags, MAC addresses, internal and external IPs.

Some attributes that Rumble is able to get from the EC2 API metadata:

- `aws.accountID`
- `aws.architecture`
- `aws.availabilityZone`
- `aws.hypervisor`
- `aws.imageID`
- `aws.instanceID`
- `aws.instanceType`
- `aws.ipv4`
- `aws.ipv6`
- `aws.keyName`
- `aws.launchTimeTS`
- `aws.macs`
- `aws.privateDNS`
- `aws.privateIP`
- `aws.publicDNS`
- `aws.publicIP`
- `aws.region`
- `aws.rootDeviceName`
- `aws.rootDeviceType`
- `aws.scanner.instanceID`
- `aws.scanner.instanceType`
- `aws.state`
- `aws.subnetID`
- `aws.tags`
- `aws.tenancy`
- `aws.virtualizationType`
- `aws.vpcID`
Additionally, Rumble will also report on other things that respond to the scan, but don't match an EC2 entry. You may see things like Amazon RDS, temporary ELBs, and Lambdas in your inventory, but you will only see EC2 metadata on EC2 assets.
Viewing coverage reports

Coverage reports help you understand potential blind spots on your network by identifying which IP spaces have been scanned, which ones contain assets, and which ones still are unknown. With this information, you can find things like missing subnets, rogue devices, and misconfigurations.

To access the coverage reports, go to Reports on the main menu and view the Coverage tab.

RFC1918 coverage report

The RFC1918 coverage report helps you better track and identify the subnets that are in use on your internal network, the ones that have been scanned, and the ones that haven't been scanned.

TCP/IP version 4 reserves three ranges of IP addresses for private use. Specified in RFC1918, they are:

<table>
<thead>
<tr>
<th>CIDR</th>
<th>Address range</th>
<th>Number of addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0.0.0/8</td>
<td>10.0.0.0 - 10.255.255.255</td>
<td>16,777,216</td>
</tr>
<tr>
<td>172.16.0.0/12</td>
<td>172.16.0.0 - 172.31.255.255</td>
<td>1,048,576</td>
</tr>
<tr>
<td>192.168.0.0/16</td>
<td>192.168.0.0 - 192.168.255.255</td>
<td>65,536</td>
</tr>
</tbody>
</table>

Most companies use these address ranges for their internal IPv4 networks, connecting them to the Internet via Network Address Translation (NAT).

To help you visualize and assess your RFC1918 coverage, the RFC1918 report includes:

- Coverage maps
- Subgrids
- statistics

RFC1918 coverage maps

A common network security and administration goal is to scan all of the available private IP addresses, to detect which subnetworks are in use on your internal network. Because of the large number of addresses, this can take a long time, leaving the problem of tracking which addresses have been scanned and which have not.

To solve this problem, Rumble's coverage report shows a graphical map of the RFC1918 private address spaces, showing which pieces have been scanned and to what
percentage of completion. This is found under Reports on the main menu, as the Coverage tab.

In the following example, we're looking at the top left corner of a 10.0.0.0/8 coverage map.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5%</td>
<td>0%</td>
<td>17%</td>
<td>71%</td>
<td>11%</td>
</tr>
<tr>
<td>16</td>
<td>0%</td>
<td>32%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>

The 10.0.0.0/16 subnet has been scanned to 5% completion, 10.1.0.0/16 hasn't been scanned at all, 10.2.0.0/16 has been 17% scanned, and so on. On the second row, 10.17.0.0/16 has been scanned 32% (row value 16 + column value 1 = 17).

Rumble will sometimes detect that a device has additional IP addresses which are not part of the range being scanned. This can indicate that the device is present on an unscanned part of the private IP address space. The coverage reports show this by drawing a red border around the appropriate grid cell.

So in the above image, 10.17.0.0/16 contains at least one IP address which was detected as belonging to a device, but which hasn't been scanned yet – it's part of the 68% of the address range remaining unscanned. Meanwhile, 10.18.0.0/16 is completely unscanned, and also apparently contains at least one IP address that is in use.

You can hover the mouse cursor over a cell to see a tooltip showing the CIDR address range the cell represents, how many unscanned hosts are believed to be in that range, and what percentage of the entire range has been scanned.

**RFC1918 sub-grids**

For the 192.168.0.0/16 map, each cell on the grid represents a /24 (256 addresses). Clicking a cell will take you to the subnet analysis report for that range and list the assets found.

For a map that shows a large address range, such as 10.0.0.0/8, each cell represents an entire /16 range of 65,536 addresses. To help narrow down the search for assets and unscanned hosts, you can click on any cell that represents a /16 range to go a grid map that showing just that range. From a /16 grid, you can use the link at top right to go back to the full range map.
On the /16 sub-grids, each cell is a /24, so clicking one takes you to the subnet analysis for that specific cell’s address range, like on the 192.168.0.0/16 map.

Statistics

At the top of the coverage report page, you can see statistics showing how much of the RFC1918 address space you have scanned. Another box breaks the coverage down by the three blocks of reserved addresses.

Clicking the binoculars icon in the summary box will create a sample scan task, covering the unscanned address ranges. The refresh buttons create scan tasks to rescan all of the appropriate range.
Search query syntax

Rumble supports a deep searching across the Asset, Service, and Wireless Inventory, across Organizations and Sites, and through the Query Library. The Rumble Export API uses the same Inventory search syntax to filter results.

Boolean operators

Search queries can be combined through AND and OR operators and be grouped using parenthesis.

AND

For example, a Asset Inventory query of os:"Windows 10" AND protocols:http AND protocols:smb2 will show only those assets where Windows 10 was identified and both SMB and a web server were discovered. Search values that contain spaces must be placed in double quotes.

OR

By contrast, the example query of os:"Windows 10" AND protocols:http OR protocols:smb2 will search for Windows 10 running a web server or any assets with the SMB service exposed. In addition to AND and OR, the NOT operator can be used to filter a query. For example, the query os:"Windows 10" AND NOT protocols:http will show Windows 10 systems without a web server. If the negation should happen as the first term the AND should be dropped. The query NOT protocol:http AND os:"Windows 10" is equivalent to the previous search, with the terms reversed.

Wildcard and fuzzy searches

Most keywords are a fuzzy match by default. To force an exact match, prefix match, or suffix match, the = prefix can be applied to the search term, with the % character used as a wildcard. To search an operating system name of just Windows, the Asset Inventory query would be os:="Windows", while to specify a prefix match of Ubuntu Linux, the query os:="Ubuntu Linux%" can be used.

Time and date values

Time and date (timestamp) fields can be searched using < (less than) and > (greater than) operators to compare against the current time. Supported units:

- hours
minutes
seconds
months
years

A special value of now can also be used.

For example, an asset search of first_seen:<1year would search for assets first detected this year. Other examples:

```
first_seen:<3days
first_seen:>2019-08-01
first_seen:>8/1/2019
last_seen:<1week
last_seen:<2months
last_seen:<1year
created_at:>2weeks
created_at:<30minutes
updated_at:>1year
updated_at:<12hours
os_eol:<now
os_eol:>4weeks
os_eol_extended:>now
os_eol_extended:>90days
```

**Empty values**

To search for an empty value, the = prefix can be used with no value after. For example, the query os:= will find assets with no identified operating system.

Note that this only works for single-valued attributes such as os and type; it won't work for multi-value attributes such as names or addresses.

**Asset and service inventory searches**

Asset and Service attributes support two special search types in addition to the documented keywords:

- Asset Inventory searches treat unknown keywords as filters against individual Asset attributes.
- Service Inventory searches treat unknown keywords as filters against individual Service data values.

In situations where an Asset keyword conflicts with a Service data key, or an Asset attribute conflicts with a Service keyword, the prefixes \_asset. and \_service. can be used to disambiguate.
Searches are handled slightly differently. Service queries can filter against Asset attributes \texttt{(os:linux)} and Service attributes \texttt{(banner:Password)}, but the Asset queries are limited to summary information about services \texttt{(protocol:ssh)}.

\textbf{View search query list}
Search queries

Asset search keywords

When viewing assets, you can use the keywords in this section to search and filter.

User specified fields

Asset comments

Use the syntax comment:<text> to search comments on an asset.

comment:"contractor laptop"
comment:"imaging server"

Asset tags

Use the syntax tag:<term> to search tags added to an asset. The term can be the tag name, or the tag name followed by an equal sign and the tag value. Tag value matches must be exact.

tag:"group"
tag:"group=production"

Site name or ID

Use the syntax site:<term> to filter by site name or ID.

site:Primary
site:"Branch Office"
site:ad67d649-041b-439d-af59-f200053a8899

Explorer name or ID

Use the syntax explorer:<term> to filter by explorer name or ID.

explorer:DESKTOP-AB451F
explorer:8b927a8e-d405-40e9-aa47-d6afc9bff237

Asset fields
Asset ID

The ID field is the unique identifier for a given asset, written as a UUID. Using the syntax id:<uuid> to filter by ID field.

id: cdb084f9-4811-445c-8ea1-3ea9cf88d536

Asset OS

The operating system field is a string describing the detected operating system software. This field is searched using the syntax os:<text>. The OS version, if available, can be searched using os_version:<number>.

os: "Windows"
os: "Ubuntu Linux"
os_version: 8

Asset type

The type field is a string describing the detected system type, such as Desktop, Laptop, Server, BMC, or Mobile. Use the syntax type:<text> to search this field.

type: Desktop
type: BMC
type: "Game Console"

Asset hardware

The hardware field is a string describing the detected physical hardware, such as macMini or Nintendo Switch. Use the syntax hardware:<text> to search this field.

hardware: Switch
hardware: macMini

Asset hostnames

The hostnames associated with an asset are obtained from DNS and exposed services. Use the syntax name:<text> to search these names.

name: "www"
name: "TV"

To search an asset where any asset has a specific prefix or suffix, start the term with = and use % as a wildcard match:

name: = "FTP.%"
name: = "%-09"
Use the syntax `name_count:<number>` to search the hostname count. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

`name_count:>1`

**Asset domains**

The domains associated with an asset are obtained from DNS and exposed services. Use the syntax `domain:<domainname>` to search the domain names.

```plaintext
domain:"amazon.com"
domain:"corp.lan"
domain:"WORKGROUP"
```

The domain count can be searched using the syntax `domain_count:<number>`. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

`domain_count:>1`

**Asset addresses**

Use the syntax `address:<ip>` to search the addresses (both primary and secondary) associated with an asset. This keyword also allows for wildcard matches using `%'`. A comma-separated list of addresses will be used as an efficient multiple-match.

```plaintext
address:192.168.0.1
address:10.0.0
address:%.0.1
address:10.%.254
address:10.0.0.1,10.0.0.2,10.0.0.3
```

Use the syntax `address_count:<term>` and `address_extra_count:<number>` to search address primary and secondary counts. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

`address_extra_count:0`

**Asset networks**

Use the syntax `net:<cidr>` to search the addresses (both primary and secondary) associated with an asset by CIDR mask.

`net:192.168.0.0/24`
**Asset default community**

Use the syntax community:<text> to search for assets with a default SNMP community (public or private).

community:public

**Asset public address**

Use the keyword has_public and syntax has_public:<boolean> to locate any asset with a non-reserved IP address.

The term is a boolean value:

- true, t, 1, and yes represent true
- false, f, 0, and no represent false

has_public:true

**Asset private address**

Use the keyword has_private and syntax has_private:<boolean> to locate any asset with a private IP address.

The term is a boolean value:

- true, t, 1, and yes represent true
- false, f, 0, and no represent false

has_private:false

**Asset IPv6 address**

Use the keyword has_ipv6 and the syntax has_ipv6:<boolean> to locate any asset with an identified IPv6 address.

The term is a boolean value:

- true, t, 1, and yes represent true
- false, f, 0, and no represent false

has_ipv6:false
Asset link local IPv6 address

Use the keyword has_link_local and syntax has_link_local:<boolean> to locate any asset with an identified IPv6 link local (fe80::) address.

The term is a boolean value:
- true, t, 1, and yes represent true
- false, f, 0, and no represent false

has_link_local:true

Asset MAC addresses

Use the syntax mac:<term> to search MAC addresses associated with an asset.

mac:00:5c:04
mac:00:00:1c

Use the syntax mac_count:<number> to search the MAC address count. This search term supports numerical comparison operators (>, >=, <, <=, =).

mac_count:>2

Asset MAC address vendors

The vendor associated with the MAC addresses of an asset can be searched using the syntax mac_vendor:<text>.

mac_vendor:Apple
mac_vendor:"Intel Corporate"

The MAC address vendor count can be searched using the syntax mac_vendor_count:<number>. This search term supports numerical comparison operators (>, >=, <, <=, =).

mac_vendor_count:0

Asset MAC address age

Use the syntax mac_age:<term> to search the allocation date of the newest MAC address associated with an asset. The term supports the standard Rumble time comparison syntax.

mac_age:>1year
mac_age:<6months
mac_age:2019-12-31
Asset attributes

Use the syntax attribute:<term> to search the asset attribute fields, such as the port used to detect the TTL.

attribute:"ip.ttl.port"
attribute:"cpe:/a:isc:bind:9.11.3"
attribute:"9.11.3"

To determine if an asset has any attribute defined, use the has:<attribute-name> keyword. The has keyword can be inverted to find missing fields with not has:<term>.

has:"ip.ttl.port"
not has:"rdns.names"

In addition to the standard fields, the following special attributes are available:

- has:screenshot returns assets where at least one screenshot was obtained.
- has:icons returns assets where at least one icon was obtained (HTTP, UPnP, or similar).
- has:uplink returns assets seen in the CAM table of a network switch.
- has:downlink returns assets where the CAM table was queried at least one other asset was connected.
- has:unmapped returns assets where the CAM table was queried at least one other asset was connected but not identified by IP.

The attribute can be specified as a term directly. If the attribute name conflicts with an existing term, the prefix _asset. can be specified to disambiguate the query.

ip.ttl.port:80
rdns.names:"router"
_asset.ip.ttl.hops:"1"

Asset services

Asset service ports

The TCP and UDP services associated with an asset can be searched by port number using the syntax port:<number>.

port:80
port:161
Asset service TCP ports

Use the syntax tcp:<number> to search the TCP services associated with an asset by port number.

tcp:443

Asset service UDP ports

Use the syntax udp:<number> to search UDP services associated with an asset by port number.

udp:53

Asset service protocols

Use the syntax protocol:<term> to search the identified service protocols associated with an asset.

protocol:http
protocol:telnet

The protocol count can be searched using the syntax protocol_count:<number>. This search supports numerical comparison operators (>, >=, <, <=, =).

protocol_count:>1

Asset service products

Use the syntax product:<term> to search for the identified service products associated with an asset.

product:openssh
products:nginx

The product count can be searched using the syntax product_count:<number>. This search term supports numerical comparison operators (>, >=, <, <=, =).

product_count:>3

Asset service counts

Use the following keywords to search the number of services associated with an asset can be searched by port number:

- service_count_tcp:<number>
• service_count_udp:<number>
• service_count_icmp:<number>
• service_count_arp:<number>

These keywords support numerical comparison operators (>, >=, <, <=, =).

Examples include:

service_count_tcp:>=5
service_count_arp:0
service_count_udp:<=1

**Asset tracking fields**

**Asset timestamps**

Use the following syntaxes to search the asset timestamp fields (first_seen, last_seen, created_at, updated_at, os_eol, os_eol_extended):

• first_seen:<term>
• last_seen:<term>
• created_at:<term>
• updated_at:<term>
• os_eol:<term>
• os_eol_extended:<term>

The term supports the standard Rumble time comparison syntax.

first_seen:<3days
first_seen:>2019-08-01
first_seen:>8/1/2019
last_seen:<1week
last_seen:<2months
last_seen:<1year
created_at:>2weeks
created_at:<30minutes
updated_at:>1year
updated_at:<12hours
os_eol:<now
os_eol:>4weeks
os_eol_extended:>now
os_eol_extended:>90days

**Asset online status**

Use the syntax online:<boolean> or the inverse syntax offline:<boolean> to search the online status of an asset.

The term is a boolean value:
true, t, 1, and yes represent true
false, f, 0, and no represent false

online:t
online:1
offline:0

Asset detection method

The detected by attribute of an asset can be searched using the syntax det:<term> or detected_by:<term>. The term is one of arp, icmp, <portnumber>-tcp, or <portnumber>-udp. In the case of multiple detections, the priority goes arp, icmp, and then the first detected service.

det:arp
detected_by:80-tcp
det:53-udp

Asset Time to Live (TTL) comparisons

Use the syntax ttl:<term> and lowest_ttl:<term> to search the lowest TTL of an asset. TTL is the estimated number of hops between the scan source and the asset.

This search term supports numerical comparison operators (>, >=, <, <=, =).

lowest_ttl:>3

Asset Round Trip Time (RTT) comparisons

Use the syntax rtt:<term> and lowest_rtt:<term> to search the lowest RTT for an asset. RTT is the round-trip response time of a given probe measured in nanoseconds (1,000,000 == 1ms).

This search term supports numerical comparison operators (>, >=, <, <=, =).

lowest_rtt:>50000000

Asset multiple MAC address status

Use the syntax multi_mac:<boolean> to determine if an asset has multiple MAC addresses.

The term is a boolean value:

• true, t, 1, and yes represent true
• false, f, 0, and no represent false
multi_mac:t

Asset any MAC address status

Use the syntax has_mac:<boolean> to find assets with any MAC addresses.

The term is a boolean value:

- true, t, 1, and yes represent $true$
- false, f, 0, and no represent $false$

has_mac:yes
has_mac:f

Asset multiple IP address status

Use the syntax multi_home:<boolean> to determine if an asset has multiple IP addresses.

The term is a boolean value:

- true, t, 1, and yes represent $true$
- false, f, 0, and no represent $false$

multi_home:t

Asset multiple hostname status

Use the syntax multi_name:<boolean> to find assets with multiple hostnames.

The term is a boolean value:

- true, t, 1, and yes represent $true$
- false, f, 0, and no represent $false$

multi_name:yes
multi_name:false

Service search keywords

When viewing services, you can use the keywords in this section to search and filter.
Service ports

The TCP and UDP services associated with a service can be searched by port number using the syntax port:<number>. This search term supports numerical comparison operators (> , >= , < , <= , =).

port:<=25

Service UDP ports

Use the udp:<number> syntax to search UDP services associated with a service by port number.

udp:443

Service TCP ports

Use the syntax tcp:<number> to search TCP service associated with a service by port number.

tcp:53

Service transport

Use the syntax transport:<term> to search the transport associated with a service by name.

transport:tcp
transport:udp
transport:icmp

Service Virtual Host (VHost)

Use the syntax vhost:<text> to search for virtual hosts associated with a service by name.

vhost:"www"

Service address

Use the keyword service_address to match against the service IP address.

service_address:192.168.0.1
Service public address

Use the keyword service_has_public and syntax service_has_public:<boolean> to locate any service with a non-reserved IP address.

The term is a boolean value:

- true, t, 1, and yes represent true
- false, f, 0, and no represent false

service_has_public:true

Service private address

Use the keyword service_has_private and syntax service_has_private:<boolean> to locate any service with a private IP address.

The term is a boolean value:

- true, t, 1, and yes represent true
- false, f, 0, and no represent false

service_has_private:false

Service IPv6 address

Use the keyword service_has_ipv6 and the syntax service_has_ipv6:<boolean> to locate any service with an identified IPv6 address.

The term is a boolean value:

- true, t, 1, and yes represent true
- false, f, 0, and no represent false

service_has_ipv6:false

Service link local IPv6 address

Use the keyword service_has_link_local and syntax service_has_link_local:<boolean> to locate any service with an identified IPv6 link local (fe80::) address.

The term is a boolean value:

- true, t, 1, and yes represent true
- false, f, 0, and no represent false
service_has_link_local:true

**Service attributes**

You can search all service attributes with the syntax `<attribute>:<term>`. This search term supports numerical comparison operators (>, >=, <, <=, =).

If the attribute name conflicts with an existing term, the prefix `_service`. can be added to disambiguate the query.

Note that service attributes can be slow and it is often better to prefix `_asset.protocol:<term>` filter in front of the service attribute query. For example, to search for SSH banners, use the syntax `_assets.protocol:ssh AND banner:<term>`.

<table>
<thead>
<tr>
<th>Banner:password</th>
<th>service.product:&quot;OpenSSH&quot;</th>
<th>html.title:&quot;Apache2 Ubuntu Default Page&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>http.code:&gt;=500</td>
<td>screenshot.image.size:=&gt;100000</td>
<td><code>_service.arp.macVendor:Xerox</code></td>
</tr>
</tbody>
</table>

To determine if a service has an attribute at all, use the has keyword. The has keyword can be inverted to find missing fields, with not has:<term>.

| has:"http.head.server" | not has:"html.title" |

---

**Wireless search keywords**

When viewing WiFi networks, you can use the keywords in this section to search and filter.

**SSID (ESSID)**

The SSID/ESSID field can be searched using the syntax `ssid:<text>`.

| ssid:"Guest Network" | ssid:"Corporate" |

**BSSID (MAC)**

The BSSID field can be searched using the syntax `bssid:<text>` or `mac:<text>`.

| bssid:"00:01:02:03:04:05" | mac:"00:01:%" |
Vendor

The vendor field can be searched using the syntax mac_vendor:<text>.

mac_vendor:"Google"
mac_vendor:"Netgear"
mac_vendor:"Cisco"

Family

The family field can be searched using the syntax family:<term>.

family:"010304"

Channels

The channels field can be searched using the syntax channel:<term>.

channel:"11"

Type

The network type field can be searched using the syntax type:<text>.

type:"infrastructure"

Interface

The network interface field can be searched using the syntax interface:<text>.

interface:"wlan0"

Encryption

The encryption field can be searched using the syntax encryption:<term>.

encryption:"aes"
encryption:"none"

Authentication

The authentication field can be searched using the syntax authentication:<term>.

authentication:"wpa2-psk"
authentication:"open"
Timestamps

The timestamp fields (first_seen, last_seen, created_at) timestamps can be searched using the syntax first_seen:<term>, last_seen:<term> and created_at:<term>. The term supports the standard Rumble time comparison syntax.

first_seen:<30seconds
first_seen:>2019-08-01
last_seen:<1week
last_seen:<2months
created_at:>2weeks
created_at:<30minutes

Signal

The signal field can be searched using the syntax signal:<number> or sig:<number>. The term can include the operators >, =, <=, and =. The default operator is =.

signal:">75"
signal:"<=25"
signal:99

Site name or ID

The site name or ID can be used as a filter with the syntax site:<term>

site:Primary
site:"Branch Office"
site:ad67d649-041b-439d-af59-f200053a8899

Explorer name or ID

The explorer name or ID can be used as a filter with the syntax explorer:<term>

eplorer:DESKTOP-AB451F
eplorer:8b927a8e-d405-40e9-aa47-d6afc9bff237

Wireless ID

The ID field is the unique identifier for a given wireless network, written as a UUID. This field is searched using the syntax id:<uuid>.

id:cd8084f9-4811-445c-8ea1-3ea9cf88d536
Last task ID

The Last Task ID field defines which task most recently reported the wireless network and is written as a UUID. This field is searched using the syntax task:<uuid>.

    task:39ab0e71-3cf1-4176-b6b0-4ed495288229

Wireless attributes

All wireless attributes can be searched using the syntax <attribute>:<term>.

    radio_type:"802.11n"

Analysis report search keywords

When viewing generated analysis reports, you can use the keywords in this section to search and filter.

Name

The Name field can be searched using the syntax name:<text>.

    name:"main"

Description

The Description field can be searched using the syntax description:<text>

    description:"compare secondary"

Report ID

The ID field is the unique identifier for a given analysis report, written as a UUID. This field is searched using the syntax id:<uuid>.

    id:cbd08f9-4811-445c-8ea1-3ea9cf88d536

Query library search keywords

When viewing saved queries, you can use the keywords in this section to search and filter.
Name

The Name field can be searched using the syntax name:<text>.

name:"smb2"

Description

The Description field can be searched using the syntax description:<text>.

description:"smb version 1"
description:"wep"

Type

The Type field can be searched using the syntax type:<term>.

type:"services"

Category

The Category field can be searched using the syntax category:<term>.

category:"security"
category:"audit"

Severity

The Severity field can be searched using the syntax severity:<term>.

severity:"info"
severity:"critical"

Created by

The Created By field can be searched using the syntax created_by:<term>.

created_by:"rumble"

Timestamps (created at, updated at)

The timestamp fields, created_at and updated_at, can be searched using the syntax created_at:<term> and updated_at:<term>. The term supports the standard Rumble time comparison syntax.
Explorer search keywords

When viewing deployed explorers, you can use the keywords in this section to search and filter.

**Name**

The Name field can be searched using the syntax `name:<text>`.

```plaintext
name:"main"
```

**Site**

The Site can be searched using the syntax `site:<text>`.

```plaintext
site:Primary
```

**Up**

Whether the explorer is up can be searched using the syntax `up:<boolean>`.

```plaintext
up:true
```

**Address**

The IP address(es) the explorer is deployed on can be searched using the syntax `address:<IP address>`.

```plaintext
address:10.0.1.200
```

**Version**

The software version of explorers can be searched using `version:<text>`.

```plaintext
version:2.9.7
```

**Npcap version**

The version of the npcap library for Windows explorers can be searched using `npcap_version:<text>`.
ncap_version:1.60

Architecture

The machine architecture explorers are deployed on can be searched using architecture:<text>.
architecture:amd64

OS

The operating system explorers are deployed on can be searched using os:<text>. Note that macOS is recorded as darwin, the underlying Unix core of macOS.
os:windows
os:darwin

Capability

The capabilities of the explorers can be searched using the syntax capability:<keyword>. Two keywords are supported:

- screenshot for explorers which can screenshot web pages
- ec2 for explorers which can describe AWS EC2 instances

Example:
capability:screenshot

Explorer tags

Use the syntax tag:<term> to search tags added to an explorer. The term can be the tag name, or the tag name followed by an equal sign and the tag value. Tag value matches must be exact.
tag:"admin"
tag:"group=cloud"

Task search keywords

When viewing all tasks, you can use the keywords in this section to search and filter.

Name

The Name field can be searched using the syntax name:<text>. 
name: "test scan"

Description

The Description field can be searched using the syntax description:<text>.

description: "full scan"

Created by

The Created By field can be searched using the syntax created_by:<term>.

created_by: "admin"

Type

The task type can be searched using type:<text>.

type: scan

Status

The task status can be searched using status:<text>.

status: error

Error

The task error message can be searched using error:<text>.

error: "no disk space"

Recurrence frequency

The frequency tasks recur at (the "Freq" column) can be searched using recur_frequency:<text> or freq:<text>. The term recurring:<boolean> or recur:<boolean> can be used to search based on whether tasks recur at all.

recur_frequency: hourly
freq: daily
freq: continuous
recur: true

To search for tasks with a frequency of Nth Weekday of Month, you can use (for example) freq: nth_weekday, 2 freq: monday to find tasks which repeat on the second monday of each month.
**Timestamps (created at, updated at)**

The timestamp fields, created_at and updated_at, can be searched using the syntax created_at:<term> and updated_at:<term>. The term supports the standard Rumble time comparison syntax.

created_at:>2weeks  
created_at:<30minutes  
updated_at:>1month  
updated_at:2hours

**Next/last run time**

You can search by next recurrence and last recurrence using the terms recur_last:<term> and recur_next:<term>. The term supports the standard Rumble time comparison syntax.

recur_last:<2hours  
recur_next:>1day

**Start time**

You can search by start time using the syntax start_time:<term>. The term supports the standard Rumble time comparison syntax.

start_time:<2hour

**Grace period**

The grace period can be searched using the syntax grace_period:<term> or just grace:<term>. The term supports the standard Rumble time comparison syntax.

grace:<2hour

**Site name or ID**

Use the syntax site:<term> to filter by site name or ID.

site:Primary  
site:"Branch Office"  
site:ad67d649-041b-439d-af59-f200053a8899

**Source**

The data source for tasks can be searched using the term source:<text> or source_id:<number>.
source:censys

Sources are:

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>rumble</td>
<td>Rumble scan</td>
</tr>
<tr>
<td>2</td>
<td>miradore</td>
<td>Miradore MDM</td>
</tr>
<tr>
<td>3</td>
<td>aws</td>
<td>AWS EC2 API</td>
</tr>
<tr>
<td>4</td>
<td>crowdstrike</td>
<td>CrowdStrike Falcon</td>
</tr>
<tr>
<td>5</td>
<td>azure</td>
<td>Microsoft Azure</td>
</tr>
<tr>
<td>6</td>
<td>censys</td>
<td>Censys Search API</td>
</tr>
<tr>
<td>7</td>
<td>vmware</td>
<td>VMWare</td>
</tr>
</tbody>
</table>

**Credential ID**

You can search for tasks which use a specific set of credentials using `credential_id:` <id>.

credential_id:d7931a68-6e56-11ec-ad72-f875a414a63a

**Parameters**

Tasks can be searched for task parameters using `params:`<text>. This can be useful for searching for scan tasks which had specific probes enabled.

params:bacnet

**Site search keywords**

When viewing sites, you can use the keywords in this section to search and filter.

**Name**

The Name field can be searched using the syntax `name:`<text>.

name:"Primary"

**Description**

The Description field can be searched using the syntax `description:`<text>.
description: "wireless"
description: "vlan 50"

Scope

The Scope field can be searched using the syntax scope:<term>.
scope: "10.10.10."

Excludes

The Excludes field can be searched using the syntax excludes:<term>.
excludes: "192.168.0."

Timestamps (created at, updated at)

The timestamp fields (created_at, updated_at) timestamps can be searched using the syntax created_at:<term> and updated_at:<term>. The term supports the standard Rumble time comparison syntax.

created_at: >2weeks
created_at: <30minutes
updated_at: >1month
updated_at: 2hours

Organization search keywords

Name

The Name field can be searched using the syntax name:<text>.
name: "main"

Description

The Description field can be searched using the syntax description:<text>
description: "branch office"
description: "pci"

Timestamps (created at, updated at)

The timestamp fields (created_at, updated_at) timestamps can be searched using the syntax created_at:<term> and updated_at:<term>. The term supports the
standard Rumble time comparison syntax.

created_at:>2weeks
created_at:<30minutes
updated_at:>1month
updated_at:2hours

---

**Credential search keywords**

When viewing saved credentials, you can use the keywords in this section to search and filter.

**Credential fields**

**Credential ID**

The ID field is the unique identifier for a given credential, written as a UUID. This field is searched using the syntax id:<uuid>.

```
id:cbd084f9-4811-445c-8ea1-3ea9cf88d536
```

**Credential name**

The credential name can be searched using the syntax name:<text>.

```
name:"AWS read-only account"
name:"Miradore API key"
```

**Credential type**

The credential type can be searched using the syntax name:<text>.

```
type:aws_access_secret
type:miradore_api_key_v1
```

**Credential global property**

The global property describes the level of access for all organizations. If a credential is global, all organizations have access to it. The global property can be searched using the syntax global:<boolean>.

The term is a boolean value:

- true, t, 1, and yes represent true
- false, f, 0, and no represent false
global:true
global:0

Credential timestamps

Credential timestamp fields (created_at and last_used_at) can be searched using the syntax:

- created_at:<term>
- last_used_at:<term>

The term supports the standard Rumble time comparison syntax.

created_at:<3days
created_at:>2019-08-01
created_at:>8/1/2019
created_at:<1week
created_at:<2months
last_used_at:<1year
last_used_at:>2weeks
last_used_at:<30minutes
last_used_at:<12hours
last_used_at:0

Credential created by

The created_by_email holds the email address for the user that created the credential. It can be searched using the syntax created_by_email:<term>.

created_by_email:user@example.com

Your team search keywords

When viewing your team, you can use the keywords in this section to search and filter.

Email

Use the syntax email:<address> to search for someone by email address.

eemail:john@example.com

Name

Use the syntax name:<text> to search for someone by name.

name:john
name:"John Smith"
**Superuser**

To search for people based on whether they have superuser access, use the term `superuser:<boolean>`.

`superuser:true`

`superuser:f`

**Access**

Use the syntax `access:<term>` to search for users with a specific access level. Possible access levels are admin, user, viewer, billing and none.

`access:admin`

**Status**

To search for users by invitation status, use the term `status:<text>`. Possible status values are activated, pending and expired.

`status:pending`

**SSO**

To search for people based on whether they can only log in via SSO, use the term `sso:<boolean>`.

`sso:true`

**MFA**

To search for people based on whether they have enrolled an MFA token, use the term `mfa:<boolean>`.

`mfa:f`

---

**Event search keywords**

When viewing system events under alerts, you can use the keywords in this section to search and filter.

**Action**

Use the syntax `action:<text>` to search by the action which caused the event.
action:agent-reconnected

Created timestamp

The timestamp fields created_at can be searched using the syntax created_at:<term>. The term supports the standard Rumble time comparison syntax.

created_at:>2weeks
created_at:<30minutes
updated_at:>1month
updated_at:2hours

Details

The details in the event record can be searched using the syntax details:<text>. This can be useful for searching for IP addresses.

details:192.168.0.1

Source and target name

The source (src) column can be searched using the syntax src:<text> or source:<text>. The target (tgt) column can be searched using tgt:<text> target:<text>.

src:crowdstrike
target:primary

Source and target type

The source type (shown at the start of the src column) can be searched using the syntax src_type:<text> or source_type:<text>.

Similarly, the target type can be searched using tgt_type:<text> or target_type:<text>.

src_type:task
target_type:site

Organization, site, source and target IDs

The IDs of organizations, sites, sources and targets mentioned in event details can be searched using the following search terms:

- organization_id:<uuid>
- site_id:<uuid>
- source_id:<uuid> or src_id:<uuid>
- **target_id:<uuid> or tgt_id:<uuid>**

The IDs are unique and are written as UUIDs.

```org
organization_id:0eacf412-6e69-11ec-88b9-f875a414a63a
```
Automating queries

Rumble's query language allows you to search and filter your asset inventory, based on asset fields and values. Rumble includes a query library of prebuilt searches which can be browsed from the Queries page. You can apply these queries after a scan to investigate discovery findings.

In addition to a flexible query language, the same search syntax can be used to track and monitor events across your network, based on any combination of fields. You can save your custom queries to reuse over and over again. Review the query syntax documentation for a refresher on how to build a query.

If there are certain queries you always want to run after a scan, you can turn on the Automatic option. After the query runs, you will be able to view its results in the Queries table.

From the Queries table, any filter that has checkmark in the Automatic column means that it is active. The Matches column is next to it. You can click on the numbered result in the Matches column to view the results of the query. If there were no matches, the filter will show 0.

Turn on automatic search queries

1. Go to the Reports page.
2. Click the Queries tab to view the Queries table.
3. Open or create the query you want to turn on.
4. Select the Automatic option.
5. Save the query.

Search query

```
alive:true AND _asset.protocol:"tls" AND
(tls.jarm:22b22b09b22b22b22b22b22b22b352842cd5d6b0278445702035e06875c OR
  tls.jarm:1dd40d40d00040d1dc1dd40d1dd40d3df2d6a0c2caa0dc59908f0d3602943 OR
  tls.jarm:07d14d16d21d21d00042d43d0000000a99ce74e2c6d013c745aa52b5cc042d 0R
```

The search query string.

☑ Automatic

Automatically run this query after each scan completes and store the count for the associated site.
Managing alerts

Rumble can trigger automatic alerts when certain events occur through a combination of Channels and Rules.

Rumble currently supports Internal, Email, and Webhook channel types.

Internal channels store events within the Alerts list within the Rumble Console. Internal alerts support explicit acknowledgement. Internal channels can be bulk acknowledged and cleared from within the Rumble Console.

Email channels can be configured to deliver mail to one or more recipients. These email messages contain a summary of the alert and a link to the specifics within the Rumble Console. Email is sent from the Rumble infrastructure using the Sendgrid service.

Webhook channels allow Rumble to post alerts to internet-reachable web services. The post request contains a standard text message for use with platforms like Slack and Mattermost, but also additional fields containing the full alert details. Webhooks are a great way to tie Rumble alerts into third-party platforms.

To trigger an alert on a channel, a Rule must be created. Rules define which events lead to alert on which channels. The name of the rule will be included in the alert content and should describe the type of event that it monitors.

The following event types can be used to create rules:

- Scan completed
- New assets found
- Assets back online
- Assets now offline
- Assets changed

Scan completion and assets changed rules can be noisy but may be useful to keep a running log of network changes over time. For a typical monitoring use case, a rule would be created to trigger on Assets now offline, Assets back online, and New assets found, automatically alerting an email alias or a Slack channel.

Alert rules, when combined with recurring scans, can be a simple way to track network changes over time.
Using the rules engine

The Rules Engine is an automation framework for monitoring, alerts, and workflow management. You can use the Rules Engine to customize alerts for the events that matter most to your organization and automate repetitive tasks. At the heart of the Rule Engine are rules. A rule defines the action that is taken based on a set of conditions. You can create rules to proactively alert your team when there are changes to things like explorers, assets, scans, organizations, and sites. You can also automate tagging and modification of asset fields based on the results of a query.

Some ways you can use the Rules Engine to help automate your workflow:

- Alert your team when new policy violations are identified.
- Modify asset fields when the assets match specific criteria.
- Bulk tag assets that match a specific query.
- Get a Slack notification when a query returns new results.
- Monitor when an explorer goes offline in the Rumble console.
- Know when there are changes to organizations, sites, and users.

Key concepts

Rules can help you stay on top of events as they happen and get better visibility across your network, assets, and your Rumble deployment. To build a rule, you need to define three things: events, conditions, and actions. A rule determines that when a specific event happens, and certain conditions are met, the system will automatically perform the configured action.

Events

Each rule begins with an event. The event sets off the trigger and puts your rule into motion. An event can be based on a query or a system-defined event. Rumble offers a library of system-defined events you can use to create your rules. Choosing any of these events will show the conditions and actions available.

Conditions

A condition narrows the scope of your rule. Unless the condition is met, the rule will not execute the action. You will only see conditions that apply for the event you have chosen. Generally, conditions specify sites, organizations, and asset attributes for the event.
Actions

An action executes your rule, if the event occurs, and the conditions meet all the criteria. An action can be a notification to a channel or it can be a modification to an asset. What you will need to configure depends on the action type. For notifications, you'll need to specify the notification channel and template. For asset modification, you can edit field like the OS vendor, OS product, OS version, hardware vendor, hardware product, hardware version, asset tags, and asset type.

Channels

A channel provides a way for you to communicate when a specific event has occurred. You can create multiple channels to support different types of communication needs. For example, you may want to create a Slack channel for one team, and an email list for another. It depends on what communication channels you prefer, and who you are trying to reach.

The body of the message uses default text from Rumble. Customizations for messaging is currently unavailable.

Create a rule

Rules set the criteria for actions to to take place. To create a rule, you need to choose an event, define the conditions, and choose a resulting action.

Step 1: Open the Rules Engine

- Go to the Alerts page and select the Rules tab.
- Click Create Rule to open the editor.

Step 2: Choose an event

- Choose an event you want to use as your trigger.
- You can browse the list of available predefined events. Use the search to quickly filter by keyword.
- Choosing ‘asset-query-results’ or ‘service-query-results’ will allow you to modify the fields for the resulting assets.
- After you've chosen an event, click Configure rule.

Step 3: Define the conditions

- Provide a descriptive name for the rule. Something that quickly that tells you what the rule does.
The conditions you can configure depend on the event you have selected. If you have an asset or service based query selected, you'll need to provide a query for the rule. This query will run against the site after the scan completes. You will also need to set the scope to an organization and site, and sometimes, depending on the event, minimum asset counts.

**Step 4: Choose an action**

- Actions can execute a notification to the channel of your choice or modify assets. For example, you can choose to send notifications via email when orphaned devices are found.

**Step 5: Turn on and save the rule**

- Turn the rule on if you want to activate it immediately. Otherwise, you can save the rule and turn it on later.
- Save the rule when you're done.

**Keep in mind**

Using scan and asset event types can be noisy, but they are useful for tracking network changes over time. To help you focus on the events that matter most, track assets that go offline, assets that come back online, and newly discovered assets.

**Monitoring the status of rules**

The rules tab of the Alerts page displays a list of all rules that have been created. For each rule, you can see:

- Whether the rule is enabled.
- The event that triggers processing of the rule.
- The organization the rule applies to, if the rule has been limited to a specific organization.
- When the rule was last triggered.
- Whether the rule resulted in an action being processed or not.

A status of “skipped” means that last time the rule was processed, its preconditions weren’t met, so no action was taken. A status of “processed” means that the rule’s preconditions were met, and its action was processed.

If there is an error processing a rule or sending a notification, the action status of the rule will be set to “error”. The error message can be seen as a tooltip on the error status.
Creating alert templates

With the Rules Engine, you can define rules that alert you on specific events, such as changes to scans, assets, and explorers. To customize the alert messages, you can create custom templates to standardize and format alerts triggered from rules. With custom templates, you can include more context and data for your alerts.

Templates can output in HTML, JSON, and text for use in emails, internal notifications, or webhooks. You can customize the contents of these templates as needed.

Note: The Rules Engine and templates are available with Rumble Professional and Enterprise editions.

Template building basics

You can define the contents of an alert message using the Mustache templating language. As long as you know a little bit about how the Mustache syntax works, you can build custom HTML and JSON templates to reference and pull in Rumble data.

- For Slack notifications, you can use Slack's Block Kit and their interactive Block Kit Builder to construct a rich message in JSON format, and then use Mustache to insert the relevant data.

- For Microsoft Teams, you can use the Adaptive Cards format to build rich messages, and again insert data via Mustache.

Our templates have two fields for template data: subject and body. Both subject and body can be customized using the Mustache syntax.

Inserting a data in a template

A standard set of objects is passed to the {{template engine}}, you just have to indicate the fields from the objects you want to insert into your template. Use the Mustache template syntax, {{variable}}, to include alert values when a rule matches certain conditions.

For HTML and JSON templates, values inserted using {{  }} are automatically escaped according to the appropriate rules. To avoid escaping a particular value, use triple curly braces, like this: {{{task.name}}}.

Inserting a value

To insert a value, put the variable name in double curly brackets, like this {{variablename}}.
The following example shows how to insert the console address:

The Rumble Console is at {{console}}.

**Inserting a value from an object**

To insert a value from a specific object, separate the object name and field name with a dot, like this {{object.fieldname}}.

The following example shows how to insert the organization name:

The organization name is {{organization.name}}.

**Inserting multiple values from an object**

To insert multiple values from an object easily, use a section. You will need to start the section with {{#objectname}} and close it with a matching {{/objectname}}. The following example shows how to insert the results from a scan that include total assets and number of assets changed:

Here are the results:
{{#scan}}
Scan found {{assets_total}} assets and changed {{assets_changed}} of them.
{{/scan}}

What happens if a field contains multiple values?

If a section refers to a field which contains multiple values, the template engine will loop through the values in the field, processing the section inside for each individual value.

The following example loops through all of the assets in report.new, and for each one, outputs its names and addresses fields. If there is nothing stored in report.new, the section between the tags will not be rendered.

{{#report.new}}
{{names}} {{addresses}}
{{/report.new}}

Note that within a loop, you can still refer to values from the outer object. If a named value isn’t found in the current loop object, the template engine will check the outer object. For example, this can be useful for referring to the {{console}} variable, which provides the root Rumble console URL.
Using boolean values

You can use boolean values with the {{#field}}{{/field}} tags. If the value of the field is false, the section between the tags is not rendered.

For example:

```plaintext
{{#query.truncated}}
(Additional results were found but not included in this report)
{{/query.truncated}}
```

Objects and fields reference

To include Rumble data and details in your alerts, you can build your template using the following objects and fields.

**globals**

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>console</td>
<td>The base URL of the Rumble web console.</td>
<td><a href="https://console.rumble.run">https://console.rumble.run</a></td>
</tr>
</tbody>
</table>

**event**

The following fields are available in the event object:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>The action which triggered the event</td>
<td>task-completed</td>
</tr>
<tr>
<td>created_at</td>
<td>When the event was created</td>
<td>2021-04-02 12:50:26 -0500</td>
</tr>
<tr>
<td>id</td>
<td>The UUID of the event</td>
<td>b4b871db-bdf1-4a42-b82d-18ae99972228</td>
</tr>
<tr>
<td>source_name</td>
<td>Name of the thing which caused the event</td>
<td>Weekly security scan</td>
</tr>
<tr>
<td>source_type</td>
<td>Type of thing which caused the event</td>
<td>task</td>
</tr>
<tr>
<td>success</td>
<td>Whether the event succeeded</td>
<td>true</td>
</tr>
<tr>
<td>target_name</td>
<td>Name of the object targeted by the event</td>
<td>Head Office</td>
</tr>
<tr>
<td>target_type</td>
<td>The type of the object targeted by the event</td>
<td>site</td>
</tr>
</tbody>
</table>

**organization**

The following fields are available in the organization object:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
</table>
id | The UUID of the organization | 86f12ee1-f0f1-419a-8799-63ff555777a
---|---|---
name | The name of the organization | IT Dept.

**site**

The following fields are available in the site object:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The UUID of the site</td>
<td>49f9323a-fea1-4af0-b490-2414c3aaeee</td>
</tr>
<tr>
<td>name</td>
<td>The name of the site</td>
<td>Head Office</td>
</tr>
</tbody>
</table>

**rule**

The following fields are available in the rule object:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>The action the rule said to take</td>
<td>notify</td>
</tr>
<tr>
<td>created_at</td>
<td>When the rule was created</td>
<td>2021-03-08 12:43:59 -0600</td>
</tr>
<tr>
<td>created_by</td>
<td>The user who created the rule</td>
<td><a href="mailto:user@example.com">user@example.com</a></td>
</tr>
<tr>
<td>event</td>
<td>The event triggering the rule</td>
<td>scan-completed</td>
</tr>
<tr>
<td>id</td>
<td>The UUID of the rule</td>
<td>b2269a2c-69a1-4652-bcdf-899938886c17</td>
</tr>
<tr>
<td>name</td>
<td>The name of the Rule</td>
<td>Alert on scan</td>
</tr>
<tr>
<td>updated_at</td>
<td>When the rule was last updated</td>
<td>2021-04-01 17:09:40 -0500</td>
</tr>
</tbody>
</table>

**scan**

For events triggered by a scan task, the following additional fields are available in a scan object:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>explorer_id</td>
<td>The UUID of the Rumble Explorer which carried out the scan</td>
<td>0fd44a62-d827-41c0-b26c-4837222d8888</td>
</tr>
<tr>
<td>assets_changed</td>
<td>The number of assets changed as a result of the scan</td>
<td>9</td>
</tr>
<tr>
<td>assets_ignored</td>
<td>The number of assets ignored by the scan</td>
<td>2</td>
</tr>
<tr>
<td>assets_new</td>
<td>The number of new assets detected by the scan</td>
<td>2</td>
</tr>
<tr>
<td>assets_offline</td>
<td>The number of assets which were previously online but now offline</td>
<td>1</td>
</tr>
<tr>
<td>Field</td>
<td>Contents</td>
<td>Example</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>assets_online</td>
<td>The number of assets previously offline but now online again</td>
<td>2</td>
</tr>
<tr>
<td>assets_total</td>
<td>The total number of assets for the site scanned (including offline)</td>
<td>11</td>
</tr>
<tr>
<td>assets_unchanged</td>
<td>The number of assets unchanged by the scan</td>
<td>1</td>
</tr>
<tr>
<td>assets_updated</td>
<td>The total number of assets up-to-date as a result of the scan (changed + unchanged)</td>
<td>15</td>
</tr>
<tr>
<td>duration</td>
<td>Duration of the scan in seconds</td>
<td>26</td>
</tr>
<tr>
<td>end_time</td>
<td>When the scan ended</td>
<td>2021-04-02 12:50:26-0500</td>
</tr>
<tr>
<td>excludes</td>
<td>Any IP addresses excluded from the scan</td>
<td>10.0.1.123</td>
</tr>
<tr>
<td>id</td>
<td>The UUID of the scan task</td>
<td>894a112c-3fb9-4301-8da7-8ce7fff4443</td>
</tr>
<tr>
<td>name</td>
<td>The name of the scan task</td>
<td>Weekly security scan</td>
</tr>
<tr>
<td>rate</td>
<td>The scan rate</td>
<td>1000</td>
</tr>
<tr>
<td>recv_bytes</td>
<td>How many bytes were received during the scan</td>
<td>45176</td>
</tr>
<tr>
<td>recv_error</td>
<td>How many receive errors were detected</td>
<td>1</td>
</tr>
<tr>
<td>recv_packets</td>
<td>How many data packets were received during the scan</td>
<td>555</td>
</tr>
<tr>
<td>scheduled_time</td>
<td>When the scan was scheduled to run</td>
<td>2021-04-02 12:48:00-0500</td>
</tr>
<tr>
<td>sent_bytes</td>
<td>How many bytes were sent during the scan</td>
<td>44740</td>
</tr>
<tr>
<td>sent_error</td>
<td>How many send errors were detected</td>
<td>0</td>
</tr>
<tr>
<td>sent_packets</td>
<td>How many data packets were sent during the scan</td>
<td>577</td>
</tr>
<tr>
<td>start_time</td>
<td>When the scan actually started</td>
<td>2021-04-02 12:49:55-0500</td>
</tr>
<tr>
<td>tags</td>
<td>An array of tags associated with the scan task</td>
<td></td>
</tr>
<tr>
<td>targets</td>
<td>The CIDR address ranges scanned</td>
<td>10.0.1.0/24</td>
</tr>
<tr>
<td>type</td>
<td>The type of operation</td>
<td>scan</td>
</tr>
</tbody>
</table>

### explorer

For events triggered by a scan task, the following additional fields are available in an `explorer` object (Rumble 2.1.8+):

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The UUID of the Rumble Explorer which carried out the scan</td>
<td>0fd44a62-d827-41c0-b26c-4837222d8888</td>
</tr>
<tr>
<td>name</td>
<td>The name of the explorer which carried out the scan</td>
<td>MM34B-2</td>
</tr>
</tbody>
</table>
internal_ip | The internal IP address of the explorer which carried out the scan | 10.0.1.200

search

For events triggered by a search query rule, the following additional fields are available in a search object:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>A link to perform the same search</td>
<td><a href="https://console.rumble.run/inventory/?search=">https://console.rumble.run/inventory/?search=</a>...</td>
</tr>
<tr>
<td>found</td>
<td>The number of matches found</td>
<td>3</td>
</tr>
<tr>
<td>comparator</td>
<td>The operation used to compare the</td>
<td>&gt;=</td>
</tr>
<tr>
<td>value</td>
<td>The value the number of matches was</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>compared against</td>
<td></td>
</tr>
</tbody>
</table>

report

For “scan completed” events, a report object contains the following results from the scan:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>truncated</td>
<td>Whether the set of objects was truncated due to large numbers of assets</td>
</tr>
<tr>
<td>changed</td>
<td>An array of changed assets (see below)</td>
</tr>
<tr>
<td>new</td>
<td>An array of new assets (see below)</td>
</tr>
<tr>
<td>offline</td>
<td>An array of assets now offline (see below)</td>
</tr>
<tr>
<td>online</td>
<td>An array of assets now online (see below)</td>
</tr>
</tbody>
</table>

The limit on the number of objects passed to the template is 25 for email notifications, 10 for Webhook notifications.

asset (from a scan report)

Each asset returned as part of a scan report has the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>addresses</td>
<td>The IP address(es) of the asset</td>
<td>10.0.1.123</td>
</tr>
<tr>
<td>alive</td>
<td>Whether the asset responded to probes</td>
<td>true</td>
</tr>
<tr>
<td>created_at</td>
<td>The timestamp when the asset record was created</td>
<td>2021-04-02 12:50:26 -0500</td>
</tr>
</tbody>
</table>
For query events, the following data is provided in the `query` variable:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>The number of rows matching the search query</td>
</tr>
<tr>
<td>assets</td>
<td>If it was an asset search, the array of assets matching the query</td>
</tr>
<tr>
<td>services</td>
<td>If it was a service search, the array of services matching the query</td>
</tr>
<tr>
<td>wlans</td>
<td>If it was a wireless network search, the array of wireless networks matching the query</td>
</tr>
<tr>
<td>truncated</td>
<td>Whether the set of assets was truncated due to the query returning a large number</td>
</tr>
</tbody>
</table>

The limit on the number of assets passed to the template from a query is 25 for email notifications, 10 for Webhook notifications.

**asset (from a query)**

For assets returned from a query rule, the following fields are available:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>addresses</td>
<td>The IP address(es) of the asset</td>
</tr>
<tr>
<td>alive</td>
<td>Whether the asset responded to probes</td>
</tr>
<tr>
<td>comments</td>
<td>Any comments set in the asset record</td>
</tr>
<tr>
<td>created_at</td>
<td>The timestamp when the asset record was created</td>
</tr>
</tbody>
</table>
detected_by | The method by which the asset was detected | arp
---|---|---
domains | Any domains the asset was found in | WORKGROUP
first_seen | The timestamp when the asset was first seen | 2021-04-02 12:50:26 -0500
hw | A summary of the asset’s hardware | ThinkPad X1
id | The UUID of the asset | b38295fb-bef1-fa42-b82d-18ae99972228
last_seen | The timestamp when the asset was most recently seen | 2021-04-02 12:50:26 -0500
macs | The list of MAC addresses associated with the asset | [F4:F5:E8:92:31, 00:D0:2D:9F:47:77]
names | Any names detected for the asset | laptop.local
organization | The organization the asset belongs to | IT
os | A summary of the asset’s operating system | Windows 10
service_count | How many services the asset is running | 4
site | The site the asset was detected at | New York
tags | An array of tags set on the asset |
type | The type of asset | Thermostat
updated_at | The timestamp when the asset record was last updated | 2021-04-02 12:50:26 -0500

**service (from a query)**

For services returned from a query rule, the following fields are available, some of which are taken from the associated asset:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The UUID of the service (not the asset)</td>
<td>b38efadb-61f1-f332-b92d-18ae99972228</td>
</tr>
<tr>
<td>created_at</td>
<td>When the service record was created</td>
<td>2021-04-02 12:50:26 -0500</td>
</tr>
<tr>
<td>summary</td>
<td>A summary of the service</td>
<td>ciscoSystems</td>
</tr>
<tr>
<td>port</td>
<td>The TCP/UDP port the service is on</td>
<td>53</td>
</tr>
<tr>
<td>vhost</td>
<td>The vhost of the service</td>
<td>ftp.example.com</td>
</tr>
<tr>
<td>address</td>
<td>The TCP/IP address of the service</td>
<td>192.168.33.44</td>
</tr>
<tr>
<td>transport</td>
<td>The transport</td>
<td>udp</td>
</tr>
<tr>
<td>protocol</td>
<td>The name of the protocol, if known</td>
<td>ssh</td>
</tr>
<tr>
<td>organization</td>
<td>The name of the organization the service’s asset belongs to</td>
<td>HR</td>
</tr>
<tr>
<td>site</td>
<td>The name of the site the service’s asset belongs to</td>
<td>Lab</td>
</tr>
</tbody>
</table>
alive | Whether the asset offering the service was alive | true
---|---|---
last_seen | When the asset was last seen | 2021-04-02 12:50:26 -0500
first_seen | When the asset was first seen | 2021-02-11 09:38:17 -0500
type | The asset type offering the service | Laptop
os | The OS offering the service | Linux
hw | The hardware offering the service | APC UPS
addresses | A list of other IP addresses associated with the asset offering the service | [192.168.0.2, 192.168.0.3]
macs | The list of MAC addresses associated with the asset | [F4:F5:E8:89:92:31, 00:D0:2D:9F:47:77]
names | Any names associated with the asset | [fw-3, fw-3a]
tags | Tags set on the asset |
domains | Any domain associated with the asset | LOCAL
service_count | A count of how many services the asset offers | 4
comments | Any comments set on the asset |

### wlan (wireless LAN, from a query)

For wireless lans (wlans) returned from a query rule, the following fields are available:

<table>
<thead>
<tr>
<th>Field</th>
<th>Contents</th>
<th>Example</th>
</tr>
</thead>
</table>
id | The UUID of the wireless LAN in Rumble’s database | f938934b-a2e3-f112-b23d-18ae9972228
last_seen | When the wlan was last seen | 2021-04-02 12:50:26 -0500
essid | The ESSID of the network | Free WiFi
bssid | The ESSID of the network | c4:41:1e:99:88:77
type | The type of wireless network | infrastructure
authentication | The authentication used to access the network | WPA2-PSK
encryption | The encryption used to protect data | AES
signal | The signal strength as a percentage | 86
channels | The channel of the network | 11
organization | The name of the organization the service’s asset belongs to | HR
site | The name of the site the service’s asset belongs to | Lab
Rumble User Guide

Example: Alert when scan completes

Let’s take an example of something you might want to get alerted on: completed scans.
We can create a rule that emails us when a scan completes, and provides us with some
details, such as the number of new, online, offline, and modified assets. We’ll build
these details into our template.

Step 1. Create a template.

You can create a template from the Alerts page.
Name the template something like Email the team when a scan completes.
For the template type, choose HTML, since we want to use this template for emails.
For the subject line, enter something that’s descriptive, like Rumble scan
{{scan.name}} completed at {{scan.end_time}}. You can use the
Mustache syntax for the subject.

Step 2. Create the body message

Now, let’s create the email body. We want the email to tell us how many new, online,
offline, and modified assets there are, as well as give us details on the new assets
discovered.
The body looks like this:
<h1>{{site.name}}</h1>
<h2>Scan Results</h2>
{{#scan}}
<ul>
<li>{{assets_new}} new assets</li>
<li>{{assets_online}} online assets</li>
<li>{{assets_offline}} offline assets</li>
<li>{{assets_changed}} modified assets</li>
</ul>
{{/scan}}
<h2>New assets</h2>
<ul>
{{#report.new}}
<li>{{names}} {{addresses}} {{os}}</li>
{{/report.new}}
{{^report.new}}
<li>No new assets were discovered.</li>
{{/report.new}}
</ul>
<p><a href="{{task.url}}">View the scan results</a></p>

128


**See that caret (^)?** It represents inverted sections. These sections only render when the list is empty or the value is empty or false.

**Step 3. Save the template and create a rule**

The template is now available for you to choose when you create a rule.

**Data type accepted by each channel**

When you create a rule, the channel you select must accept the data type used by the template. For example, if you want to send a Slack notification, the template must be in plain text or JSON.

Here are the data types accepted by each channel:

<table>
<thead>
<tr>
<th>Channel</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td>Plain text, HTML, JSON</td>
</tr>
<tr>
<td>Webhook</td>
<td>Plain text, JSON</td>
</tr>
<tr>
<td>Internal notification</td>
<td>Plain text</td>
</tr>
</tbody>
</table>

**Managing templates**

Go to Alerts > Templates. From this page, you can create, edit, and delete templates. Before deleting templates, make sure they are not in use by any rules. If you delete a template that is in use by a rule, the rule will revert to the default template.
Managing licenses

As a Rumble superuser or billing user, you can access and manage your organization's licensing, plan, and billing information.

Not a superuser or billing user? Please contact your organization's superuser or billing user to get help with licensing and billing information. These users are tagged with a yellow star.

How do I view my license?

If you're a superuser or billing user, go to Account > License to view your Rumble licensing information.

Your licensing information will show:

- The edition you have.
- The number of organizations, sites, explorers, and user accounts you have.
- The number of live assets you have across all organizations.
- The number of project assets you have.
- The total number of live assets and projects your license supports.

Rumble offers Starter, Pro, and Enterprise editions. What you can see and do in Rumble depends on the edition and license that you have. See what your edition includes.

When does my Professional or Enterprise subscription expire?

To see when your subscription or license expires, go to Account > License.

Find the line: This is a Rumble [edition] subscription that expires at [date and time].

If your subscription has expired, you will see: This is a Rumble [edition] subscription that expired on [date and time]. You will no longer be able to run discovery scans. To continue using Rumble, you will need to renew your subscription.

Does the Starter Edition license expire? Nope, it's part of our free tier and has no expiration date. You can continue to use it as long as you don't have more than 256 live assets.
How do I renew my Professional subscription?

When your Professional subscription expires, you can choose to renew, convert to the Starter edition, or work with us on other licensing options.

If you are using a credit card to renew the Professional edition, you can do it directly from the console. Go to the License page and find the plan you want. Plan pricing is based on live asset counts. You can subscribe on a monthly basis, or opt-in annually for a discounted rate.

Choose the plan you want from the following table. You’ll be taken through the checkout process.

<table>
<thead>
<tr>
<th>Live Assets</th>
<th>Monthly</th>
<th>Annually</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>Subscribe</td>
<td>Subscribe</td>
</tr>
<tr>
<td>2500</td>
<td>Subscribe</td>
<td>Subscribe</td>
</tr>
<tr>
<td>5000</td>
<td>Subscribe</td>
<td>Subscribe</td>
</tr>
<tr>
<td>7500</td>
<td>Subscribe</td>
<td>Subscribe</td>
</tr>
<tr>
<td>10000</td>
<td>Subscribe</td>
<td>Subscribe</td>
</tr>
<tr>
<td>12500</td>
<td>Subscribe</td>
<td>Subscribe</td>
</tr>
</tbody>
</table>

Need to pay another way? For other payment methods, like POs, please contact us for help.

How do I renew my Enterprise plan?

For help renewing your Enterprise plan, please contact us. We’ll work with you to find a custom solution for your organizational needs.

How do I convert to the Starter Edition?

No longer need the Professional Edition? You can convert it to the Starter Edition. This plan is on our free tier and supports up to 256 live assets.

How do I find my invoices?

Super users and billing users can access invoices from the Rumble Console. Go to the License page to see all invoices.
How do I change or cancel my subscription?

Please contact us if you need to modify or cancel your subscription.
Using the scanner

Rumble includes a standalone command-line scanner that can be used to perform network discovery without access to the internet. The scanner has the same options and similar performance characteristics to the explorer. The scanner output file named scan.rumble.gz can be uploaded to the Rumble Console through the Inventory Import menu.

The scanner works best with root privileges on Linux/macOS and Administrator privileges on Windows. Although the scanner will function without privileged access, many probe types will be unavailable. The sudo command can be used to run the scanner as root on Linux and macOS, while the tool is best run from an elevated command shell on Windows. On the Windows platform, the Rumble Scanner will look for an existing npcap installation and try to install it if the software is not found. This behavior can be disabled with the --nopcap flag.

The Rumble Scanner defaults to a semi-interactive terminal interface that writes multiple output files to a directory. The default directory name is rumble-[current-date]. To switch to plain text output, use the --text option. To skip artifact generation and only produce the raw JSON output file, use the flags --text -o disable --output-raw scan.rumble.

Input can be provided as arguments on the command-line or by specifying an input file using the --input (or -i) parameter. Input can consist of specific IPv4 addresses or IPv4 CIDRs. Supported formats include:

- 10.0.0.1
- 10.0.0.0/24
- 10.0.0.0/255.255.255.0
- 10.0.0.1-10.0.0.255
- example.com
- example.com/24

For hostnames, each IPv4 address in the response will be expanded with the optional mask. IPv6 is not yet supported.

The example below downloads and runs the scanner on a Linux x86_64 host. This URL will be different for your installation. The current download links for your organization are available from the scanner page of the Rumble Console.

$ wget https://console.rumble.run/download/scanner/[unique-link]/rumble-scanner-linux-amd64.bin
$ chmod +x rumble-scanner-linux-amd64.bin
$ sudo ./rumble-scanner-linux-amd64.bin 192.168.0.0/24 -o output-dir
Please note that the hexadecimal values in the download URL are specific for your account and organization.

**Starter Edition limits**

Rumble Starter Edition includes access to a limited version of the Rumble Scanner. This edition has the following restrictions:

- Each scan is limited to 1024 targets at a time
- Scanner downloads expires after 90 days
- No baseline support for asset tracking
- No API or automatic upload support
- No product or service integrations
- Limited report generation

**Performance & scope**

The default speed of Rumble scans is limited to 1,000 packets per second with a single pass. This setting works great for reliable wired networks without stateful firewalls between the scanning system and the destination networks. This rate can be changed via the `--rate` (or `-r`) option, with a reasonable maximum being 10000 for most networks. On slow unreliable networks, a rate of 300 with `--passes` set to 3 may provide better results.

A second parameter, `--max-host-rate` limits how many packets are sent per second to each individual host. This defaults to 40, which is low, but necessary when scanning low-power embedded devices. In cases where a small number of hosts (or a single host) should be scanned quickly, the `--max-host-rate` parameter can be increased to match the `--rate`.

The following example demonstrates a scan of 65,535 TCP ports on all hosts of the 192.168.0.0/24 subnet running at 10,000 packets per second:

```
$ sudo ./rumble-scanner-linux-amd64.bin 192.168.0.0/24 -r 10000 --tcp-ports 1-65535 -o output-dir
```

**Automatic web screenshots**

The `--screenshots` option (default true) introduced in version 0.6.6 tells Rumble to obtain a screenshot of all web services identified during the scan. This feature depends on the system running the explorer having a local installation of the Google Chrome or Chromium browsers. The acquired screenshots will be reported as a base64 string, stored in the “screenshot.image” field of the containing service scan result.
Rumble scanner commands and options

The Rumble Scanner supports a wide range of options, including the ability to limit scans to specific ports, probes, and snmp communities. The --help output provides basic documentation on the available options.

Commands

completion
  generate the autocompletion script for the specified shell
help
  Help about any command
license
  Display license information
upgrade
  Upgrade to the latest version of the Rumble Scanner
verify
  Perform an internal signature verification
version
  Print the version number of Rumble

Flags

--api-key string
  Specify the Rumble API key
--api-no-verify
  Disable TLS verification for API communication
--api-url string
  Specify the Rumble API server hostname (default
  "https://console.rumble.run/api/v1.0")
--arp-fast
  Enables fast mode by ARP scanning at the specified rate without delay
--bacnet-ports string
  The destination ports for BACnet probes (default “46808,47808,48808”)
-b, --baseline string
  Use the specified file as an asset baseline for tracking
--cpu string
  Write a cpu profile after the scan completes
--crestron-port uint
  The destination port for Crestron probes (default 41794)
--dns-disable-meraki-detection
   Disables detection of Meraki DNS interception

--dns-port uint
   The destination port for DNS probes (default 53)

--dns-resolve-name string
   The target hostname for DNS queries (‘off’ to disable) (default “www.google.com”)

--dns-trace-domain string
   The subdomain to use for trace requests (‘off’ to disable) (default “helper.rumble.network”)

--dtls-ports string
   The destination ports for DTLS probes (default “443,3391,4433,5246,5349,5684”)

--echo-report-errors
   Report errors from intermediate in-scope hosts

--exclude string
   Specify scan exclusions

--excludefile string
   Read exclusions from an input file

-f,--fingerprints string
   Use the specified directory as an alternate fingerprint database

--fingerprints-debug
   Enable debug output for the fingerprint processor

--goroutines string
   Write a goroutine dump after the scan completes

--heap string
   Write a heap profile after the scan completes

-h,--help
   help for Rumble

--host-ping
   Only scan hosts that respond to a ping scan using the host-ping settings

--host-ping-arp-fast
   Enables fast mode by ARP scanning at the specified rate without delay (host ping)

--host-ping-bacnet-ports string
   The destination ports for BACnet probes (host ping) (default “46808,47808,48808”)

--host-ping-crestron-port uint
   The destination port for Crestron probes (host ping) (default 41794)

--host-ping-dns-disable-meraki-detection
   Disables detection of Meraki DNS interception (host ping)

--host-ping-dns-port uint
   The destination port for DNS probes (host ping) (default 53)
--host-ping-dns-resolve-name string
    The target hostname for DNS queries (‘off’ to disable) (host ping) (default “www.google.com”)

--host-ping-dns-trace-domain string
    The subdomain to use for trace requests (‘off’ to disable) (host ping) (default “helper.rumble.network”)

--host-ping-dtls-ports string
    The destination ports for DTLS probes (host ping) (default “443,3391,4433,5246,5349,5684”)

--host-ping-echo-report-errors
    Report errors from intermediate in-scope hosts (host ping)

--host-ping-ike-port uint
    The destination port for IKE probes (host ping) (default 500)

--host-ping-ipmi-port uint
    The destination port for IPMI probes (host ping) (default 623)

--host-ping-max-attempts int
    Set the maximum number of attempts for each probe (default 2)

--host-ping-max-ttl int
    Set the default TTL on host-ping probe packets (default 255)

--host-ping-mdns-port uint
    The destination port for MDNS probes (host ping) (default 5353)

--host-ping-memcache-port uint
    The destination port for memcached probes (host ping) (default 11211)

--host-ping-mssql-port uint
    The destination port for MSSQL probes (host ping) (default 1434)

--host-ping-natpmp-port uint
    The destination port for NATPMP probes (host ping) (default 5351)

--host-ping-netbios-port uint
    The destination port for NetBIOS probes (host ping) (default 137)

--host-ping-ntp-port uint
    The destination port for NTP probes (host ping) (default 123)

--host-ping-openvpn-ports string
    The destination ports for OpenVPN probes (host ping) (default “1194”)

--host-ping-passes int
    Set the number of passes for the host-ping phase (default 1)

--host-ping-pca-port uint
    The destination port for PCAnywhere probes (host ping) (default 5632)

--host-ping-probes string
    Launch a subset of the probes for the host-ping, comma-delimited (default “arp,echo,syn,connect,netbios,snmp,ntp,sunrpc,ike,openvpn,mdns”)
--host-ping-rdns-max-concurrent int
   The maximum number of concurrent DNS lookups (host ping) (default 64)

--host-ping-rpcbind-port uint
   The destination port for RPCBind probes (host ping) (default 111)

--host-ping-rpcbind-port-nfs uint
   The destination port for NFS probes (host ping) (default 2049)

--host-ping-sip-port uint
   The destination port for SIP probes (host ping) (default 5060)

--host-ping-snmp-comms string
   The comma-separated list of SNMP v1/v2c communities (host ping) (default "public, private")

--host-ping-snmp-poll-interval uint
   The minimum number of seconds between polling each host after initial discovery (host ping) (default 300)

--host-ping-snmp-port uint
   The destination port for SNMP probes (host ping) (default 161)

--host-ping-snmp-timeout uint
   The maximum number of seconds for each individual SNMP operation (host ping) (default 5)

--host-ping-snmp-v3-auth-passphrase string
   The authentication passphrase (host ping)

--host-ping-snmp-v3-auth-protocol string
   The authentication protocol (none, md5, sha, sha224, sha256, sha384, sha512) (host ping) (default "none")

--host-ping-snmp-v3-context string
   The optional SNMP v3 context to supply (host ping)

--host-ping-snmp-v3-privacy-passphrase string
   The privacy passphrase (host ping)

--host-ping-snmp-v3-privacy-protocol string
   The privacy protocol (none, des, aes, aes192, aes256, aes192c, aes256c) (host ping) (default "none")

--host-ping-snmp-v3-username string
   The username to use for SNMP v3 authentication (host ping)

--host-ping-ssdp-port uint
   The destination port for UPNP/SSDP probes (host ping) (default 1900)

--host-ping-syn-max-retries uint
   The maximum number of retries trace and SYN requests (host ping) (default 2)

--host-ping-syn-report-resets
   This determines whether RST responses are reported (host ping) (default true)

--host-ping-syn-udp-trace-port uint
The UDP port number to use for UDP trace requests (host ping) (default 65535)
--host-ping-tcp-ports string
   The list of TCP ports to host-ping using the syn and connect probes (default
   “22,80,135,179,443,1720,3389,5040,7547,62078”)
--host-ping-tftp-ports string
   The destination ports for TFTP probes (host ping) (default “69”)
--host-ping-ubnt-port uint
   The destination port for Ubiquiti probes (host ping) (default 10001)
--host-ping-verbose
   Display verbose output for the host-ping
--host-ping-very-verbose
   Display very verbose output for the host-ping
--host-ping-wlan-list-poll-interval uint
   The minimum number of seconds between polling the access point list (host ping) (default
   300)
--host-ping-wsd-port uint
   The destination port for WSD probes (host ping) (default 3702)
--ike-port uint
   The destination port for IKE probes (default 500)
-I, --import stringArray
   Import existing scan data from the specified input files (‘scan.rumble’ format)
-i, --input-targets string
   Read scan targets from the specified input file
--ipmi-port uint
   The destination port for IPMI probes (default 623)
--max-attempts int
   Set the maximum number of attempts for each probe (default 3)
-G, --max-group-size int
   Set the maximum number of targets to process in each group (default 4096)
-R, --max-host-rate int
   Set the maximum packets-per-second rate for each individual target (default 40)
--max-sockets int
   Set the maximum number of concurrent sockets (default 2048)
--max-ttl int
   Set the default TTL on probe packets (default 255)
--mdns-port uint
   The destination port for MDNS probes (default 5353)
--memcache-port uint
   The destination port for memcached probes (default 11211)
--mssql-port uint
    The destination port for MSSQL probes (default 1434)

--nameservers string
    One or more nameservers to use for DNS resolution

--natpmp-port uint
    The destination port for NATPMP probes (default 5351)

--netbios-port uint
    The destination port for NetBIOS probes (default 137)

--nowait
    Exit the user interface immediately upon completion

--ntp-port uint
    The destination port for NTP probes (default 123)

--openvpn-ports string
    The destination ports for OpenVPN probes (default “1194”)

-o, --output string
    Output directory for scan results and analysis (‘disable’ to skip)

--output-raw string
    Set the raw output file for scan data

--overwrite
    Overwrite and replace the output directory if it already exists

--passes int
    Set the number of passes for each probe (default 1)

--pca-port uint
    The destination port for PCAnywhere probes (default 5632)

--probes string
    Launch a subset of the probes, comma-delimited (default “arp,aws-instances,bacnet,connect,crestron,dns,dtls,echo,ike,ipmi,mdns,memcache,mssql,natpmp,netbios,ntp,openvplist,wsd”)

-r, --rate int
    Set the maximum packets-per-second rate for the scan (default 1000)

--rdns-max-concurrent int
    The maximum number of concurrent DNS lookups (default 64)

--rpcbind-port uint
    The destination port for RPCBind probes (default 111)

--rpcbind-port-nfs uint
    The destination port for NFS probes (default 2049)

-S, --screenshots
    Capture screenshots from scan target web services (default true)

--sip-port uint
    The destination port for SIP probes (default 5060)
Rumble User Guide

--snmp-comms string
   The comma-separated list of SNMP v1/v2c communities (default “public,private”)

--snmp-poll-interval uint
   The minimum number of seconds between polling each host after initial discovery (default 300)

--snmp-port uint
   The destination port for SNMP probes (default 161)

--snmp-timeout uint
   The maximum number of seconds for each individual SNMP operation (default 5)

--snmp-v3-auth-passphrase string
   The authentication passphrase

--snmp-v3-auth-protocol string
   The authentication protocol (none, md5, sha, sha224, sha256, sha384, sha512) (default “none”)

--snmp-v3-context string
   The optional SNMP v3 context to supply

--snmp-v3-privacy-passphrase string
   The privacy passphrase

--snmp-v3-privacy-protocol string
   The privacy protocol (none, des, aes, aes192, aes256, aes192c, aes256c) (default “none”)

--snmp-v3-username string
   The username to use for SNMP v3 authentication

--ssdp-port uint
   The destination port for UPNP/SSDP probes (default 1900)

--subnet-ping
   Only scan subnets that have at least one active response using the subnet-ping settings

--subnet-ping-arp-fast
   Enables fast mode by ARP scanning at the specified rate without delay (subnet ping)

--subnet-ping-bacnet-ports string
   The destination ports for BACnet probes (subnet ping) (default “46808,47808,48808”)

--subnet-ping-crestron-port uint
   The destination port for Crestron probes (subnet ping) (default 41794)

--subnet-ping-dns-disable-meraki-detection
   Disables detection of Meraki DNS interception (subnet ping)

--subnet-ping-dns-port uint
   The destination port for DNS probes (subnet ping) (default 53)

--subnet-ping-dns-resolve-name string
   The target hostname for DNS queries (‘off’ to disable) (subnet ping) (default “www.google.com”)
--subnet-ping-dns-trace-domain string
    The subdomain to use for trace requests ('off' to disable) (subnet ping) (default “helper.rumble.network”)

--subnet-ping-dtls-ports string
    The destination ports for DTLS probes (subnet ping) (default “443,3391,4433,5246,5349,5684”)

--subnet-ping-echo-report-errors
    Report errors from intermediate in-scope hosts (subnet ping)

--subnet-ping-ike-port uint
    The destination port for IKE probes (subnet ping) (default 500)

--subnet-ping-ipmi-port uint
    The destination port for IPMI probes (subnet ping) (default 623)

--subnet-ping-max-attempts int
    Set the maximum number of attempts for each probe (default 1)

--subnet-ping-max-ttl int
    Set the default TTL on subnet-ping probe packets (default 255)

--subnet-ping-mdns-port uint
    The destination port for MDNS probes (subnet ping) (default 5353)

--subnet-ping-memcache-port uint
    The destination port for memcached probes (subnet ping) (default 11211)

--subnet-ping-mode string
    Set the subnet-ping discovery profile: auto (default “auto”)

--subnet-ping-mssql-port uint
    The destination port for MSSQL probes (subnet ping) (default 1434)

--subnet-ping-natpmp-port uint
    The destination port for NATPMP probes (subnet ping) (default 5351)

--subnet-ping-net-size int
    Set the subnet size to use for the subnet ping (default 256)

--subnet-ping-netbios-port uint
    The destination port for NetBIOS probes (subnet ping) (default 137)

--subnet-ping-ntp-port uint
    The destination port for NTP probes (subnet ping) (default 123)

--subnet-ping-openvpn-ports string
    The destination ports for OpenVPN probes (subnet ping) (default “1194”)

--subnet-ping-passes int
    Set the number of passes for the subnet-ping phase (default 1)

--subnet-ping-pca-port uint
    The destination port for PCAnywhere probes (subnet ping) (default 5632)

--subnet-ping-probes string
Launch a subset of the probes for the subnet-ping, comma-delimited (default “arp,echo,syn,connect,netbios,snmp,ntp,sunrpc,ike,openvpn,mdns”)

--subnet-ping-rdns-max-concurrent int
   The maximum number of concurrent DNS lookups (subnet ping) (default 64)

--subnet-ping-rpcbind-port uint
   The destination port for RPCBind probes (subnet ping) (default 111)

--subnet-ping-rpcbind-port-nfs uint
   The destination port for NFS probes (subnet ping) (default 2049)

--subnet-ping-sample-rate int
   Set the sample rate of addresses within each subnet as a percentage (default 4)

--subnet-ping-sip-port uint
   The destination port for SIP probes (subnet ping) (default 5060)

--subnet-ping-snmp-comms string
   The comma-separated list of SNMP v1/v2c communities (subnet ping) (default “public,private”)

--subnet-ping-snmp-poll-interval uint
   The minimum number of seconds between polling each host after initial discovery (subnet ping) (default 300)

--subnet-ping-snmp-port uint
   The destination port for SNMP probes (subnet ping) (default 161)

--subnet-ping-snmp-timeout uint
   The maximum number of seconds for each individual SNMP operation (subnet ping) (default 5)

--subnet-ping-snmp-v3-auth-passphrase string
   The authentication passphrase (subnet ping)

--subnet-ping-snmp-v3-auth-protocol string
   The authentication protocol (none, md5, sha, sha224, sha256, sha384, sha512) (subnet ping) (default “none”)

--subnet-ping-snmp-v3-context string
   The optional SNMP v3 context to supply (subnet ping)

--subnet-ping-snmp-v3-privacy-passphrase string
   The privacy passphrase (subnet ping)

--subnet-ping-snmp-v3-privacy-protocol string
   The privacy protocol (none, des, aes, aes192, aes256, aes192c, aes256c) (subnet ping) (default “none”)

--subnet-ping-snmp-v3-username string
   The username to use for SNMP v3 authentication (subnet ping)

--subnet-ping-ssdp-port uint
   The destination port for UPNP/SSDP probes (subnet ping) (default 1900)

--subnet-ping-syn-max-retries uint
The maximum number of retries trace and SYN requests (subnet ping) (default 2)

--subnet-ping-syn-report-resets
This determines whether RST responses are reported (subnet ping) (default true)

--subnet-ping-syn-udp-trace-port uint
The UDP port number to use for UDP trace requests (subnet ping) (default 65535)

--subnet-ping-tcp-ports string
The list of TCP ports to subnet-ping using the syn and connect probes (default “22,80,135,179,443,1720,3389,5040,7547,62078”)

--subnet-ping-tftp-ports string
The destination ports for TFTP probes (subnet ping) (default “69”)

--subnet-ping-ubnt-port uint
The destination port for Ubiquiti probes (subnet ping) (default 10001)

--subnet-ping-verbose
Display verbose output for the subnet-ping

--subnet-ping-very-verbose
Display very verbose output for the subnet-ping

--subnet-ping-wlan-list-poll-interval uint
The minimum number of seconds between polling the access point list (subnet ping) (default 300)

--subnet-ping-wsd-port uint
The destination port for WSD probes (subnet ping) (default 3702)

--syn-max-retries uint
The maximum number of retries trace and SYN requests (default 2)

--syn-report-resets
This determines whether RST responses are reported (default true)

--syn-udp-trace-port uint
The UDP port number to use for UDP trace requests (default 65535)

--tcp-excludes string
The list of TCP ports to always exclude

-p, --tcp-ports string
The list of TCP ports scan using the syn and connect probes (see below for default)

--tcp-skip-protocol
Set this to skip protocol detection on TCP ports

--text
Force text-only mode (no console ui)

--tftp-ports string
The destination ports for TFTP probes (default “69”)

--ubnt-port uint
The destination port for Ubiquiti probes (default 10001)
-upload
  Automatically upload scan results to the Rumble Console
-u, --upload-site string
  Specify an optional Site ID or Name to upload the raw scan results to (default “Primary”)
-v, --verbose
  Display verbose output
--very-verbose
  Display very verbose output
--wlan-list-poll-interval uint
  The minimum number of seconds between polling the access point list (default 300)
--wsd-port uint
  The destination port for WSD probes (default 3702)

**Standard ports scanned**

<table>
<thead>
<tr>
<th>1</th>
<th>7</th>
<th>9</th>
<th>13</th>
<th>19</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>25</th>
<th>37</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>43</td>
<td>49</td>
<td>53</td>
<td>69</td>
<td>70</td>
<td>79</td>
<td>80</td>
<td>81</td>
<td>82</td>
</tr>
<tr>
<td>83</td>
<td>84</td>
<td>85</td>
<td>88</td>
<td>102</td>
<td>105</td>
<td>109</td>
<td>110</td>
<td>111</td>
<td>113</td>
</tr>
<tr>
<td>119</td>
<td>123</td>
<td>135</td>
<td>137</td>
<td>139</td>
<td>143</td>
<td>161</td>
<td>179</td>
<td>222</td>
<td>264</td>
</tr>
<tr>
<td>384</td>
<td>389</td>
<td>402</td>
<td>407</td>
<td>443</td>
<td>444</td>
<td>445</td>
<td>465</td>
<td>500</td>
<td>502</td>
</tr>
<tr>
<td>512</td>
<td>513</td>
<td>515</td>
<td>523</td>
<td>524</td>
<td>540</td>
<td>548</td>
<td>554</td>
<td>587</td>
<td>617</td>
</tr>
<tr>
<td>623</td>
<td>631</td>
<td>636</td>
<td>689</td>
<td>705</td>
<td>771</td>
<td>783</td>
<td>873</td>
<td>888</td>
<td>902</td>
</tr>
<tr>
<td>903</td>
<td>910</td>
<td>912</td>
<td>921</td>
<td>990</td>
<td>993</td>
<td>995</td>
<td>998</td>
<td>1000</td>
<td>1024</td>
</tr>
<tr>
<td>1030</td>
<td>1035</td>
<td>1080</td>
<td>1089</td>
<td>1090</td>
<td>1091</td>
<td>1098</td>
<td>1099</td>
<td>1100</td>
<td>1101</td>
</tr>
<tr>
<td>1102</td>
<td>1103</td>
<td>1128</td>
<td>1129</td>
<td>1158</td>
<td>1199</td>
<td>1211</td>
<td>1220</td>
<td>1270</td>
<td>1234</td>
</tr>
<tr>
<td>1241</td>
<td>1300</td>
<td>1311</td>
<td>1352</td>
<td>1433</td>
<td>1440</td>
<td>1468</td>
<td>1494</td>
<td>1521</td>
<td>1530</td>
</tr>
<tr>
<td>1533</td>
<td>1581</td>
<td>1582</td>
<td>1583</td>
<td>1604</td>
<td>1723</td>
<td>1755</td>
<td>1811</td>
<td>1883</td>
<td>1900</td>
</tr>
<tr>
<td>2000</td>
<td>2049</td>
<td>2082</td>
<td>2083</td>
<td>2100</td>
<td>2103</td>
<td>2121</td>
<td>2181</td>
<td>2199</td>
<td>2207</td>
</tr>
<tr>
<td>2222</td>
<td>2323</td>
<td>2362</td>
<td>2375</td>
<td>2379</td>
<td>2380</td>
<td>2381</td>
<td>2525</td>
<td>2533</td>
<td>2598</td>
</tr>
<tr>
<td>2601</td>
<td>2604</td>
<td>2638</td>
<td>2809</td>
<td>2947</td>
<td>2967</td>
<td>3000</td>
<td>3037</td>
<td>3050</td>
<td>3057</td>
</tr>
<tr>
<td>3128</td>
<td>3200</td>
<td>3268</td>
<td>3269</td>
<td>3217</td>
<td>3273</td>
<td>3299</td>
<td>3300</td>
<td>3306</td>
<td>3311</td>
</tr>
<tr>
<td>3312</td>
<td>3351</td>
<td>3389</td>
<td>3460</td>
<td>3500</td>
<td>3628</td>
<td>3632</td>
<td>3690</td>
<td>3780</td>
<td>3790</td>
</tr>
<tr>
<td>3817</td>
<td>4000</td>
<td>4092</td>
<td>4322</td>
<td>4343</td>
<td>4369</td>
<td>4433</td>
<td>4443</td>
<td>4444</td>
<td>4445</td>
</tr>
<tr>
<td>4567</td>
<td>4659</td>
<td>4679</td>
<td>4730</td>
<td>4786</td>
<td>4840</td>
<td>4848</td>
<td>5000</td>
<td>5022</td>
<td>5037</td>
</tr>
<tr>
<td>5038</td>
<td>5040</td>
<td>5051</td>
<td>5060</td>
<td>5061</td>
<td>5093</td>
<td>5168</td>
<td>5222</td>
<td>5247</td>
<td>5250</td>
</tr>
<tr>
<td>5275</td>
<td>5347</td>
<td>5351</td>
<td>5353</td>
<td>5355</td>
<td>5392</td>
<td>5400</td>
<td>5405</td>
<td>5432</td>
<td>5433</td>
</tr>
<tr>
<td>5498</td>
<td>5520</td>
<td>5521</td>
<td>5554</td>
<td>5555</td>
<td>5556</td>
<td>5580</td>
<td>5601</td>
<td>5631</td>
<td>5632</td>
</tr>
<tr>
<td>5666</td>
<td>5671</td>
<td>5672</td>
<td>5683</td>
<td>5800</td>
<td>5814</td>
<td>5900</td>
<td>5920</td>
<td>5938</td>
<td>5984</td>
</tr>
<tr>
<td>5985</td>
<td>5986</td>
<td>5988</td>
<td>5989</td>
<td>6000</td>
<td>6001</td>
<td>6002</td>
<td>6050</td>
<td>6060</td>
<td>6070</td>
</tr>
<tr>
<td>6080</td>
<td>6082</td>
<td>6101</td>
<td>6106</td>
<td>6112</td>
<td>6262</td>
<td>6379</td>
<td>6405</td>
<td>6502</td>
<td>6503</td>
</tr>
<tr>
<td>6504</td>
<td>6514</td>
<td>6542</td>
<td>6556</td>
<td>6660</td>
<td>6661</td>
<td>6667</td>
<td>6905</td>
<td>6988</td>
<td>7000</td>
</tr>
<tr>
<td>7001</td>
<td>7021</td>
<td>7071</td>
<td>7077</td>
<td>7080</td>
<td>7100</td>
<td>7144</td>
<td>7181</td>
<td>7210</td>
<td>7373</td>
</tr>
<tr>
<td>7443</td>
<td>7474</td>
<td>7510</td>
<td>7547</td>
<td>7579</td>
<td>7580</td>
<td>7676</td>
<td>7700</td>
<td>7770</td>
<td>7777</td>
</tr>
</tbody>
</table>
Scan outputs

The Rumble Scanner generates a directory of output files by default. This directory includes the following items.

- `scan.rumble.gz`: The raw scan data compressed via gzip, this can be imported or reprocessed via `--import`
- `assets.jsonl`: The new optimized format for correlated, fingerprinted assets.
- `nmap.xml`: A Nmap XML compatible data file that can be imported into various security tools.
- `urls.txt`: A list of discovered web services in URL format.
- `protocols.csv`: A list of protocols with their ports and URLs.
- `assets.html`: A rudimentary HTML report with screenshots.
- `screenshots`: A directory of raw screenshot images, headers in JSON format, and HTML bodies.
- Various lists including `addresses.txt`, `addresses_all.txt`, `hostnames.txt`, and `domains.txt`

Raw Scan Data

The Rumble Scanner raw data is stored in a file named `scan.rumble.gz` within the output directory. This file contains JSONL-formatted records. An example ARP response record is shown below.
The `info` field is a JSON map of strings to strings. Multiple values are encoded using the tab character (0x09), which are otherwise escaped as \t (along with \r and \n for carriage return and line feed bytes and \x00 for null bytes). Rumble scans may return more than one record of the same type for the same host if multiple responses were received.

In addition to the `result` type, there are also records for status messages, stats, and an initial `config` type that contains the scan parameters.
Site Comparison Report

The Site Comparison Report lets you generate a side-by-side analysis of two sites, so you can understand:

- **How assets change over time** such as their TCP/UDP services, TCP/UDP ports, and service protocols. You can leverage this data to evaluate historical changes for assets for a specific point in time.
- **How exposure changes** based on scanning your network from different locations. For example, if you use public IP addresses internally and externally, you may want to scan those addresses from inside and outside your network to understand your potential exposure.

The report provides a summary view of differences. It only captures certain attributes that were added or removed from an asset, such as IP addresses, TCP ports, TCP service counts, UDP ports, UDP service counts, service protocols, and service counts. It does not track every modification to an asset, such as fingerprint or service banner changes.

After you run the report, the data presented in the report will be static. Any changes to your current inventory may result in assets no longer being accessible from the report.

**Generate a Site Comparison Report**

Generating the Site Comparison Report requires selecting a current site and a comparison site. The sites can be in different organizations. You can also select “All sites” to compare all sites in an organization with a different site and organization.

When the report runs it will assemble the two sets of assets specified, compare them using Rumble’s asset matching algorithms, and generate a set of differences. You can then browse the summarized results in the report.

**To generate a Site Comparison Report:**

1. Verify that your current organization is one containing an inventory of assets you want to compare.
2. Go to the Site Comparison Report.
3. When the Site Comparison Report configuration page appears, the current organization will be set to the one you currently have selected. You can change the current site, if needed. For the comparison, choose the organization and site you want to run the analysis against.
4. Run the report. A task will be created to perform the comparisons, and you will be taken to the task page.
5. When the task is complete, the report will appear in the list of recent analysis reports at the top of the Reports page. You can then **view and search the results**.

**View how assets change over time**

To analyze how assets have changed over time, you can compare the data from an old scan task with your most recent inventory. Setting a point-in-time comparison requires creating a new project that you can import your old scan data into.

Here’s how you can set up a point-in-time analysis:

1. Go to the organization or project that contains the task scan data you’d like to use.

2. Go to your **completed tasks** and locate the task that contains the data for the point in time you’d like to compare.

3. From the task page, download the task data. It will be in a file with a name starting `scan_` and ending `.json.gz`. This is the file you’ll import into your new project. You don’t need to uncompress the file, unless you’re curious to look at the JSON data.

4. Next, **create a new project** for your import. You can create an organization if you intend to perform the analysis regularly.

5. After you create the project, go to the Inventory page and **import your scan task data** into it.

Now, you’re ready to compare your current inventory with a previous version of it. Go to the **Site Comparison Report**. You’ll need to select the organization and site for your current inventory as the site to compare against.

After the report runs, it shows a table that **highlights the differences** between the two sites, which in this case will represent two points in time, going from past to present.

You can also run the comparison by selecting your current organization first, and then choosing the project with the past data as the comparison site. The results will be the same, but with their sense reversed – that is, services which show as added when going from past to present, will show as removed when going from present to past.
View how exposure differs between networks

You can run the Site Comparison Report to compare how exposure varies based on where you scan your network from. By comparing two inventories from two different perspectives, you can obtain better insights on your attack surface, which can help with active defense or risk reduction. For example, if you use external IPs internally and externally, you may want to scan those addresses from inside your network and outside your network. Then, you can run the Site Comparison Report to compare the results from those two sites.

Here’s how you can set up a diff for exposures between networks:

1. Set up a site with an explorer on one network.
2. Set up another site with an explorer on a different network. This site can be in the same organization as the first site.
3. Run a scan of the same address range with each explorer. Verify you have the correct site selected for each scan.
4. After the scans complete, run the Site Comparison Report to generate the diff. The report will show a table that highlights the differences between the two sites.

Analyze the results in the Site Comparison Report

The Site Comparison Report generates a table to show how assets' addresses, names, services, ports, and protocols differ between sites. Red text, denoted by the minus (-) sign, indicates that attributes were removed going from left to right. Green text, denoted by the plus (+) sign, indicates attributes were added.

<table>
<thead>
<tr>
<th>Address</th>
<th>Name</th>
<th>Other Address</th>
<th>Other Name</th>
<th>TCP services</th>
<th>TCP ports</th>
<th>UDP services</th>
<th>UDP ports</th>
<th>Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.86.77</td>
<td>IPAD-5</td>
<td>192.168.86.77</td>
<td>IPAD-5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>192.168.86.31</td>
<td>BEDROOM-1</td>
<td>192.168.86.31</td>
<td>BEDROOM-1</td>
<td>6 + 5</td>
<td>-51009</td>
<td>-51009</td>
<td></td>
<td></td>
</tr>
<tr>
<td>192.168.86.24</td>
<td>APPLE-TV-2</td>
<td>192.168.86.24</td>
<td>APPLE-TV-2</td>
<td>7 + 5</td>
<td>-49153 -52674</td>
<td>-49153 -52674</td>
<td></td>
<td></td>
</tr>
<tr>
<td>192.168.86.52</td>
<td>[absent]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Here are some things you should know about the results:

- **Address and Other Address** - The first address column contains the asset's addresses for the organization currently selected when the report was requested. The second address column, Other Address, contains the asset's addresses for the organization and site that was compared against the current organization.

- **Name and Other Name** - The first name column contains the asset's names in the organization currently selected when the report was generated. The second name column, Other Name, contains the asset's names for the organization and site compared against.

- **TCP and UDP services** - These columns show how the total number of TCP and UDP services differ between sites.
- **TCP and UDP ports** - These columns show the ports that have been added or removed between sites.

You can click on the green info (i) icon to view a more detailed comparison of the asset.

### Asset differences

**Compare test-1 - All sites with test-2 - All sites**

Report created Dec 7 2021 4:56AM [UTC-6] (Tue) by thao@rumble.run

<table>
<thead>
<tr>
<th>Attribute</th>
<th>test-1 - All sites - 192.168.86.77</th>
<th>test-2 - All sites - 192.168.86.77</th>
</tr>
</thead>
<tbody>
<tr>
<td>macs</td>
<td>e4:dc:88:3d:70:57</td>
<td></td>
</tr>
<tr>
<td>newest_mac</td>
<td>e4:dc:88:3d:70:57</td>
<td></td>
</tr>
<tr>
<td>service_count</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>service_count_udp</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Clicking on one of the asset addresses in the report will bring up the full current asset record, if the asset still exists in the appropriate organization and site.

### Search the Site Comparison Report

You can search the report using the Rumble search query language. In the following descriptions, the main set refers to the assets in the organization that was current when you generated the report (i.e., the address and name columns). The comparison set refers to the assets in the organization and site that you chose to compare against (i.e., the other address and other name columns).

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>address:</td>
<td>Search for assets in the main set with a specified IP address.</td>
</tr>
<tr>
<td>net: or cidr:</td>
<td>Filter assets by their network CIDR range in the main set.</td>
</tr>
<tr>
<td>other_address:</td>
<td>Search for assets in the comparison set with a specified IP address.</td>
</tr>
<tr>
<td>other_net: or other_cidr:</td>
<td>Filter assets by their network CIDR range in the comparison set.</td>
</tr>
<tr>
<td>id:</td>
<td>Search by asset ID in the main set.</td>
</tr>
<tr>
<td>other_id:</td>
<td>Search by asset ID in the comparison set.</td>
</tr>
<tr>
<td>tcp:</td>
<td>Search for a TCP port change by number.</td>
</tr>
<tr>
<td>udp:</td>
<td>Search for a UDP port change by number.</td>
</tr>
<tr>
<td>protocol:</td>
<td>Search for a TCP or UDP port change by service name.</td>
</tr>
</tbody>
</table>
ServiceNow CMDB

Rumble Enterprise integrates with ServiceNow Configuration Management Database (CMDB) through a Rumble JSON endpoint, with asset data formatted as CMDB Configuration Items (CIs). This integration brings Rumble data into ServiceNow, allowing for specific fields and CI class mappings to be fine-tuned from the ServiceNow console.

A basic import can be performed using ServiceNow IntegrationHub ETL. ETL allows you to create transform maps via point-and-click operations without programming. It processes incoming data via the ServiceNow Robust Transform Engine (RTE), and ensures that the Identification and Reconciliation Engine (IRE) is used to integrate the data into your existing ServiceNow CMDB instance.

For more advanced imports, you can of course develop transform maps manually, and import the data that way.

This document will briefly cover the process of setting up ServiceNow IntegrationHub ETL to fetch data from Rumble via REST API.

Before you begin

- Because ServiceNow is so customizable, it is impossible to make integration a “one click” process. At a minimum, you will need to spend time deciding which CI classes you want to use, and how you want to map data to their fields.

- You will definitely want to test the import process in a development instance of ServiceNow. You can obtain one by signing up for ServiceNow’s developer program at developer.servicenow.com.

- IntegrationHub ETL may need to be installed from the ServiceNow store.

- ServiceNow ITOM is licensed based on the number of CIs in your CMDB, regardless of whether they were discovered by ServiceNow Discovery or by another tool such as Rumble. You will need a subscription to ITOM Visibility or ITOM Discovery, with enough Subscription Units to cover the number of CIs in your organization.

Set up an ETL import

To set up an ETL import, you’ll need to:

- Install IntegrationHub ETL
- Set up Rumble as a discovery source
• Set up credentials
• Set up a connection and credential alias
• Set up a connection
• Set up a data source
• Create a new data flow action
• Set up an ETL import

Install IntegrationHub ETL

For a development ServiceNow instance, you will need to install the IntegrationHub ETL application:

1. Go to System Applications > All Available Applications > All.
2. Search for IntegrationHub ETL by name, and wait until it appears.

3. Click Install on the info page, then Install again on the confirmation.

ServiceNow will automatically install Integration Commons for CMDB and Integration Studio API as well. Once installation is complete, you will need to shift-reload the browser page before InstallationHub ETL shows up in the left navigator.

Set up Rumble as a discovery source

The first step is to set up a discovery source to let you flag CIs which were added from Rumble data.

Navigate to System Definition > Choice Lists in ServiceNow. Search for the table cmdb_ci, and the element discovery_source. Assuming there isn’t one for Rumble, click New to create a new choice.

In the Table search box, enter [cmdb_ci] (including the square brackets), and click Configuration Item [cmdb_ci]. The remaining required values are:
Click *Submit* to create the new choice.

**Set up credentials**

You will need a Rumble organization API key. These are set up via the Organizations page in Rumble. You can click on an organization, scroll down to the bottom of the page, and generate a new API key or copy an existing one. The API key looks like a string of hexadecimal digits.

Once you have your API key, navigate to *Connections & Credentials > Credentials* in ServiceNow, click *New*, and select *API Key* as the type of credential to create.

Give the new credential a sensible name like ‘Rumble export API key’, paste in the hex string as the API Key value, then click *Submit*. 
Set up a connection and credential alias

Next, you will want to set up a connection and credential alias. To do this, go to Connections & Credentials > Connection & Credential Aliases and click on New. Provide the necessary information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>rumble_servicenow_export</td>
</tr>
<tr>
<td>Type</td>
<td>Connection &amp; Credential</td>
</tr>
<tr>
<td>Connection type</td>
<td>HTTP</td>
</tr>
</tbody>
</table>

The defaults should be OK for the other fields.

Set up a connection

Now it's time to set up the actual HTTP connection based on the credentials and alias you've created. Navigate to Connections & Credentials > Connections, click New, and choose to set up an HTTP(S) Connection.

Enter the required field values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Rumble Export API</td>
</tr>
<tr>
<td>Credential (from the earlier step)</td>
<td>Rumble export API key</td>
</tr>
<tr>
<td>Connection alias (from the previous step)</td>
<td>rumble_servicenow_export</td>
</tr>
<tr>
<td>Connection URL</td>
<td><a href="https://console.rumble.run/">https://console.rumble.run/</a></td>
</tr>
</tbody>
</table>
Note that the connection alias name will have had spaces and other punctuation replaced with underscores.

Set up a data source

Now you should be ready to start setting up a data source for the Rumble data.

Go to System Import Sets > Administration > Data Sources and click New.

Some suitable field values are:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Rumble JSON export HTTPS</td>
</tr>
<tr>
<td>Import set table name</td>
<td>u_rumble</td>
</tr>
<tr>
<td>Type</td>
<td>REST (IntegrationHub)</td>
</tr>
<tr>
<td>Format</td>
<td>JSON</td>
</tr>
<tr>
<td>Path for each row</td>
<td>//</td>
</tr>
</tbody>
</table>

Note that the import set table name is required to start with u_. There shouldn’t be any arrays in the exported data, but you can check the ‘Discard Arrays’ option to be safe.
With the basic info in place, check towards the top right of the form next to the *Request action* field. Clicking the button to the right of that field will launch the ServiceNow Flow Designer to create a new data flow action.

Create new data flow action with the flow designer

The flow designer will prompt you to name your new action, so give it a descriptive name like “Fetch Rumble assets”.

Once you confirm the action name and other basic info, you should find yourself looking at an action outline with three steps:
Click on *REST step* on the left to edit the REST action details.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Use Connection Alias</td>
</tr>
<tr>
<td>Connection Alias</td>
<td>The connection you created earlier</td>
</tr>
<tr>
<td>Base URL</td>
<td>This will be read from the connection</td>
</tr>
<tr>
<td>Build Request</td>
<td>Manually</td>
</tr>
<tr>
<td>Resource Path</td>
<td>/api/v1.0/export/org/assets.servicenow.json</td>
</tr>
<tr>
<td>HTTP Method</td>
<td>GET</td>
</tr>
</tbody>
</table>

Under *Headers*, you need to click to add a new header. The header name is *Authorization*, and for the value you should type in *Bearer* followed by a space. You can then drag the lozenge labeled *Credential Value* from the right side of the screen into the Value box, and it should show up as per the screenshot below:
Note that you can build the resource path using a similar method and the Attachment Name lozenge, but in this case we suggest hard-coding it as it doesn’t need to change for different attachment names.

At this point it’s probably a good idea to use the Test button at the top of the screen to test the action. You can provide a fake value for the attachment name and click Run Test. ServiceNow should (eventually) say:

Your test has finished running. View the action execution details.

To do so, close the dialog and click the Executions button to open the Operation List tab. Click the link to open the most recent test operation into another tab. Under Output Data, make sure you got status_code 200 (HTTP OK) from Rumble. If you get an error in the 400 range, there’s probably an error with the Authorization header or your API key.
You might optionally want to check the Response Payload Size by twisting the disclosure triangle next to *Steps*.

If the action is working, close both the Operation tabs to go back to the “Fetch Rumble asset data” action, and click the *Publish* button.

Now you can close the Flow Designer tab in your browser and go back to setting up the ServiceNow data source.

Back on the data source page, you should now be able to use the 'Request action' field to add the request action you just created, and save the data source.

**Set up an ETL import**

Go to *Configuration > IntegrationHub ETL* and click *New* to create a new set of transform maps. The ETL tool leads you through the setup in numbered stages.

In section 1, click on *Import Source Data and Provide Basic Details*. 
Note that the name is used for transform map names.

Once you've filled out the basic information, click **Save**, and also **Mark as Complete** so that ServiceNow will allow you to proceed to step 2.

In section 2, click **Preview and Prepared Data**. The IntegrationHub ETL tool should load in the first 10 rows of your Rumble data.

At this point, you can proceed with the normal process for importing data via IntegrationHub ETL. You can transform fields, decide which asset types become which classes of CI, and map fields in the normal way. A good introduction to the rest of the process is the ServiceNow Knowledge2020 course “Use IntegrationHub ETL to Import Data”.

**Things to know about Rumble export data**

**MAC addresses and network adapters**

ServiceNow supports modeling computers with multiple network cards via the Network Adapter CI class. Unfortunately, the Network Adapter CI requires a name for the network adapter. Rumble has no way to obtain that information, and it differs between OSs and OS versions.

One option is to set a standard name for all Network Adapter CIs. You can then use ETL’s lookup tools or Glide Lookups to search existing computer data by MAC address and avoid duplicates.
Specific fields

Here are some field-by-field notes on the data in Rumble's ServiceNow export.

Note that some fields can contain multiple values, in which case they are separated with semicolons.

- **asset_id**: This is a universally unique ID (UUID) for the Rumble asset. It will remain the same across Rumble scans, so it can be used as the *source native key* in ServiceNow.

- **type**: This is the Rumble asset type.

- **sys_class_name**: This is a suggested ServiceNow CI class that might be suitable for the asset. Not all assets will have a *sys_class_name* value supplied, and you might disagree with some of the suggestions. We hope to improve the suggestions over time.

- **os_vendor**: The vendor responsible for the OS running on the asset.

- **os**: The specific OS product running.

- **os_version**: The version of the OS product, if it could be determined.

- **manufacturer**: The manufacturer or vendor of the primary hardware.

- **model**: The specific product. You will most likely want to use a Glide Lookup or the ETL field mapping features to look up the manufacturer and model, and convert them to references to the manufacturer and product tables.

- **ip_address**: The primary IP address the asset was detected at.

- **addresses_scope**: A list of all of the IP addresses of the asset that were directly detected by scanning.

- **addresses_extra**: Additional IP addresses that we infer the asset could have, but which haven’t been confirmed by direct scan.

- **mac_address**: The primary MAC address for the asset.

- **mac_manufacturer**: The manufacturer assigned to the MAC address, generally the manufacturer of the network hardware.

- **newest_mac_age**: MAC address allocations have dates associated with them. This field has the approximate allocation date of the newest MAC address found
associated with the asset. This can be used to provide an approximate age for the asset.

- **macs**: A list of all MAC addresses found for the asset.
- **mac_vendors**: A list of all the MAC vendors found for the asset.
- **names, name**: For convenience we provide a single-value name field, as well as names listing all of the identified names.
- **last_updated**: This is when the asset record in Rumble was last updated. It can be used as an indication of data recency in ServiceNow.

**Useful links**

- [ServiceNow CMDB developer information](#)
- [CMDB schema information](#)
- [ServiceNow Knowledge2020 course: Use IntegrationHub ETL to Import Data](#)
Splunk Search

Rumble Enterprise integrates with Splunk using a dedicated Splunk Addon, compatible with Splunk 7, Splunk 8, and Splunk Cloud. With this add-on, you’ll be able to pull new or updated hosts into a Splunk index, where you’ll be able to analyze, visualize, and monitor them there.

This add-on uses the Splunk API from the Rumble Network Discovery platform. It supports syncing assets into Splunk, with multiple inputs supported, global API key management, and optional search filters for each input. For example, you can track new assets as one input, and SMBv1 enabled assets as another input.

To set up this add-on, you’ll need an Export API or Organization API key, which you can generate from your Organization page in the Rumble Console.

Get the Rumble add-on for Splunk

1. Log in to Splunk.
2. Go to Find More Apps.
3. Search for Rumble Network Discovery.
4. Install the add-on for Rumble.
5. Splunk will prompt you to log in again. After you log back in again, the add-on will be installed. You’ll be able to open the Rumble Asset Sync app. Splunk might also prompt you to restart your server.

Asset sync modes

Two asset sync modes are available: New Assets Only and All Updated Assets. You can export asset inventory that contains newly discovered assets or updated assets, since the last poll, in a sync-friendly format for Splunk. You can leverage the same capabilities from the Asset Sync API to pull data in Splunk, such as search filters, fields, and time-based checkpoints.

Once data is pulled into Splunk, you can create Splunk inputs with filters. This allows you to sync specific assets with a certain protocol, discovery date, or open service.
CrowdStrike Falcon

Rumble Enterprise integrates with CrowdStrike by importing data through the CrowdStrike Falcon API. This integration allows you to sync and enrich your asset inventory. Adding your CrowdStrike data to Rumble makes it easier to find things like endpoints that are missing an EDR agent.

- For CrowdStrike Falcon hosts which can be matched to an existing Rumble asset, asset level attributes such as operating system, hardware platform, hostname, and MAC address will be updated, and CrowdStrike-specific attributes will be added.
- For hosts which cannot be matched with an existing Rumble asset, a new asset will be created in the site specified when the integration task is set up.

Rumble is able to merge existing assets with Falcon data when the MAC address or hostname overlaps. Falcon devices can also be manually merged into Rumble assets using the Merge button on the Asset Inventory screen.

Please note that any IP address reported by Falcon will be treated as a secondary address, not a primary address, since these IPs can be stale and may not be associated with a specific network or site.

Getting started

To set up the CrowdStrike integration, you'll need to:

1. Configure CrowdStrike to allow API access through Rumble.
2. Add the CrowdStrike credentials, which will include the client ID and client secret, and CrowdStrike base API URL in Rumble.
3. Activate the CrowdStrike connection to sync your data with Rumble.

Requirements

Before you can set up the CrowdStrike integration:

- Verify that you have Rumble Enterprise.
- Make sure you have access to the CrowdStrike admin portal.

Step 1: Configure CrowdStrike to allow API access to Rumble

1. Log in to CrowdStrike.
2. Go to Support > API Clients and Keys. When the API Key page appears, choose to add a new API client.
3. Provide the following details for the API client:
Step 1: Set up the CrowdStrike API client in Rumble

- **Client name** - API client name, such as Rumble.
- **API scope** - Read permissions for Hosts and Host Groups.

4. When you are done, add the client. An API client created window appears and shows you the client ID and client secret. You’ll need them to configure the integration in Rumble.
5. Copy the client ID and client secret now. You may not be able to get them later.

**Step 2: Add the CrowdStrike credentials to Rumble**

1. Go to the **Credentials page** in Rumble. Provide a name for the credentials, like CrowdStrike Falcon.
2. Choose **CrowdStrike Falcon API key** from the list of credential types.
3. Provide the following information:
   - **CrowdStrike client ID and CrowdStrike client secret** - To generate your client ID and client secret, go to Support > API Clients and Keys > OAuth2 API clients > Add new API Client in your CrowdStrike portal.
   - **CrowdStrike API URL** - Your organization-specific base URL, which will depend on your account type. It will be something like api.crowdstrike.com.
4. If you want other organizations to be able to use these credentials, select the **Make this a global credential** option. Otherwise, you can configure access on a per organization basis.
5. Save the credentials. You’re now ready to set up and activate the connection to bring in data from CrowdStrike.

**Step 3: Set up and activate the CrowdStrike connection to sync data**

After you add your CrowdStrike credentials, you’ll need to set up a connection to sync your data from CrowdStrike. A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new CrowdStrike-only assets are created.

1. Activate a connection to CrowdStrike. You can access all available third-party connections from your **inventory** or **tasks page**.
2. Choose the credentials you added earlier. If you don’t see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Enter a name for the task, like CrowdStrike sync.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under **Task configuration**, choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
6. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the **Scheduled tasks** to see when the next sync will occur.
Step 4: View CrowdStrike assets

After a successful sync, you can go to your inventory to view your CrowdStrike assets. These assets will have a CrowdStrike icon listed in the Source column.

To filter by CrowdStrike assets, consider running the following queries:

- **View all CrowdStrike assets:**
  
  `source:crowdstrike`

- **Find assets that have a CrowdStrike EDR agent installed:**
  
  `edr.name:CrowdStrike`

- **Find Windows assets, excluding servers, that are missing a CrowdStrike EDR agent:**
  
  `os:windows and not type:server and not edr.name:CrowdStrike`

Click into each asset to see its individual attributes. Rumble will show you the attributes returned by the CrowdStrike API, with the exception of policies.

CrowdStrike attributes

Rumble will enrich your assets with the following attributes, if the information is available:

- `bios_manufacturer`
- `bios_version`
- `build_number`
- `cid`
- `config_id_base`
- `config_id_build`
- `config_id_platform`
- `detection_suppression_status`
- `device_id`
- `email`
- `external_ip`
- `first_login_timestamp`
- `first_login_user`
- `first_seen`
- `group_hash`
- `groups`
- `host_hidden_status`
- `hostname`
- `instance_id`
- `last_login_timestamp`
- `last_login_user`
- `last_seen`
local_ip
mac_address
machine_domain
meta.version
modified_timestamp
notes
os_version
ou
platform_id
platform_name
pointer_size
product_type
product_type_desc
provision_status
reduced_functionality_mode
serial_number
service_pack_major
service_pack_minor
service_provider
service_provider_account_id
site_name
slow_changing_modified_timestamp
status
system_manufacturer
system_product_name
tags
Azure Virtual Machines

Rumble Enterprise integrates with Microsoft Azure to deliver greater visibility into your cloud assets. This integration imports data through the Azure Virtual Machines API to enrich your asset inventory. Syncing with Azure allows you to view information about your device’s OS profile, storage profile, and more.

This integration imports all Azure virtual machines that are in a running state. It sets the Azure-specific attributes and updates asset-level attributes including the virtual machine type, hardware platform, hostname, and MAC address. Rumble is able to merge existing assets with Azure data when the MAC address is the same. Azure devices can also be manually merged into Rumble assets using the Merge button on the Asset Inventory screen.

Getting started

To set up the Azure integration, you’ll need to:

1. Configure Azure to allow API access through Rumble.
2. Add an Azure credential to Rumble.
3. Activate the Azure VM connection to sync your data with Rumble.

Requirements

Before you can set up the Azure VM integration:

- Verify that you have Rumble Enterprise.
- Make sure you have access to the Microsoft Azure portal.

Step 1: Configure Azure to allow API access through Rumble.

1. Log into the Microsoft Azure portal.
2. Go to Azure Active Directory > App registrations and click on New registration.
   - Provide a name.
   - Select the supported account types.
   - Optionally add a redirect URI.
3. Click register to register the application.
4. Once the application is created, you should see the Overview dashboard. Save the following information:
   - Application (client) ID
   - Directory (tenant) ID
5. Give the client access to the subscriptions you want to sync. From the subscription details page, go to Access Control (IAM) and select Add > Add role
assignment. Enter the following:
- Role: Reader
- Assign access to: User, group, or service principal
- Under Select, search for the name of the application you created. Click on your application to add it to the Selected members list below.

6. Click Save to save the role assignment.

7. Navigate to Azure Active Directory > App registrations and select the application you created.

8. Go to Certificates & secrets and click on New client secret.
   - Enter a description.
   - Select the expiration.

9. Click Add to create the client secret. Save the following information:
   - Client secret value

Step 2: Add the Azure credential to Rumble.

Add an Azure Client Secret credential to Rumble

This type of credential can be used to sync all VMs in a single directory (across multiple subscriptions).

1. Go to the Credentials page in Rumble and click Add Credential.
2. Provide a name for the credential, like Azure Client Secret.
3. Choose Azure Client Secret from the list of credential types.
4. Provide the following information:
   - Azure application (client) ID - The unique ID for the registered application. This can be found in the Azure portal if you go to Azure Active Directory > App registrations and select the application.
   - Azure client secret - To generate a client secret, go to Azure Active Directory > App registrations, select your application, go to Certificates & secrets and click on New client secret.
   - Azure directory (tenant) ID - The unique ID for the tenant. This can be found in the Azure portal if you go to Azure Active Directory > App registrations and select the application.
   - Select the Access all subscriptions in this directory (tenant) option to sync all VMs in your directory. Otherwise, specify the Azure subscription ID - The unique ID for the subscription that you want to sync. This can be found in the Azure portal if you go to Subscriptions and select the subscription.
5. If you want other organizations to be able to use this credential, select the Make this a global credential option. Otherwise, you can configure access on a per organization basis.
6. Save the credential. You’re now ready to set up and activate the connection to bring in data from Azure.
Add an Azure Username & Password credential to Rumble

This type of credential can be used to sync all VMs across directories. Alternatively, you can add one Azure Client Secret credential for each Azure directory you want to sync.

1. Go to the Credentials page in Rumble and click Add Credential.
2. Provide a name for the credential, like Azure User/Pass.
3. Choose Azure Username & Password from the list of credential types.
4. Provide the following information:
   - **Azure application (client) ID** - The unique ID for the registered application. This can be found in the Azure portal if you go to Azure Active Directory > App registrations and select the application.
   - **Azure directory (tenant) ID** - The unique ID for the tenant. This can be found in the Azure portal if you go to Azure Active Directory > App registrations and select the application.
   - **Azure username** - The username for your Azure cloud account. This cannot be a federated user account.
   - **Azure password** - The password for your Azure cloud account.
5. If you want other organizations to be able to use this credential, select the Make this a global credential option. Otherwise, you can configure access on a per organization basis.
6. Save the credential. You’re now ready to set up and activate the connection to bring in data from Azure.

Step 3: Set up and activate the Azure VM connection to sync data

After you add your Azure credential, you’ll need to set up a connection to sync your data from Azure. A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where the data is organized.

1. Activate a connection to Azure. You can access all available third-party connections from your inventory or tasks page.
2. Choose the credential you added earlier. If you don’t see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like Azure sync.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under Task configuration, choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
6. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the Scheduled tasks to see when the next sync will occur.
Step 4: View Azure assets

After a successful sync, you can go to your inventory to view your Azure assets. These assets will have an Azure icon listed in the Source column.

To view all your Azure assets, run the following query:

- View all Azure assets:

  source:azure

Click into each asset to see its individual attributes. Rumble will show you the attributes returned by the Azure VM API.

Azure attributes

Rumble will enrich your assets with the following attributes, if the information is available:

availabilityZone
hardwareProfile.vmSize
id
ipv4
ipv6
keyNames
launchTimeTS
location
macs
name
osProfile.adminUsername
osProfile.computerName
privateIP
publicDNS
publicIP
resourceGroup
state
storageProfile.imageReference.exactVersion
storageProfile.imageReference.offer
storageProfile.imageReference.publisher
storageProfile.imageReference.sku
storageProfile.imageReference.version
storageProfile.osDisk.caching
storageProfile.osDisk.createOption
storageProfile.osDisk.diskSizeGB
storageProfile.osDisk.managedDisk.id
storageProfile.osDisk.managedDisk.storageAccountType
storageProfile.osDisk.name
storageProfile.osDisk.osType
subnetID
tags
type
vmID
Amazon Web Services

Rumble Enterprise integrates with Amazon Web Services to provide better visibility across your cloud environment. This integration imports data from the AWS EC2 API as well as the AWS ELB APIs (Elastic Load Balancing - Version 1/Elastic Load Balancing - Version 2), adding detailed information to your asset inventory. Syncing with AWS allows you to quickly identify the number of EC2 instances and elastic load balancers you have running, their region, account, and more.

This integration supports the import of all running EC2 instances and active application, network, gateway, or classic load balancers. It can be configured to connect to a single AWS account or to all accounts in your organization and imports data across multiple regions. It sets the AWS-specific attributes and updates asset-level attributes as well. Rumble is able to merge existing assets with AWS data when the MAC address is the same. AWS devices can also be manually merged into Rumble assets using the Merge button on the Asset Inventory screen.

Getting started

To set up the AWS integration, you’ll need to:

1. Configure AWS to allow API access through Rumble.
2. Add the AWS credential to Rumble, which includes the access key and secret key.
3. Activate the AWS EC2 connection to sync your data with Rumble.

Requirements

Before you can set up the AWS EC2 integration:

- Verify that you have Rumble Enterprise.
- Make sure you have access to the AWS console.
- Make sure you are using AWS Organizations if you want to connect to multiple accounts.

Step 1: Configure AWS to allow API access through Rumble

To connect to a single AWS account:

1. Log into the AWS console.
2. Go to Identity and Access Management (IAM) > Users and select the user that will provide API access to Rumble.
3. Click Add permissions > Attach existing policies directly. Search for and attach the AmazonEC2ReadOnlyAccess policy.
4. From the user summary screen, open the **Security credentials** tab and click on **Create access key**.
5. Save the **Access key ID** and **Secret access key**.

**To connect to all accounts in your organization:**

1. Log into the AWS console.
2. For each account in your organization, create a role and assign the AmazonEC2ReadOnlyAccess policy. You can do this one at a time for each account or use StackSets to deploy the role if you have a large number of accounts:
   1. Using StackSets:
      1. While logged into your organization's management account, go to **CloudFormation > StackSets** and click **Create StackSet**.
      2. Select **Template is ready** and upload a file with the following JSON template. Replace `<accountID>` with the account ID where the role was created and `<roleName>` with the name of the role. Click next.

```
{
  "Resources": {
    "IAMRumbleRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [{
            "Action": "sts:AssumeRole",
            "Effect": "Allow",
            "Principal": {
              "AWS": "arn:aws:iam::<ac-
```
3. Enter a name for the StackSet. Click next.
4. Optionally set tags. Click next.
5. Set the deployment options. Click next.
6. Review and create the StackSet.

2. Using IAM:
1. Go to Identity and Access Management (IAM) > Roles and click Create role.
2. Choose Another AWS Account for the type of trusted entity.
3. For Account ID, enter the ID for your organization's management account.
4. Click Next: Permissions.
5. Attach the AmazonEC2ReadOnlyAccess policy.
6. Click Next: Tags and add tags optionally.
7. Click Next: Review and provide a name for the role. (The role must be named the same for all accounts)
8. Click Create role.

3. For your organization's management account, create an inline policy to allow the STS AssumeRole action.
   1. Go to Identity and Access Management (IAM) > Users and select the user that will provide API access to Rumble.
   2. Click Add inline policy.
3. In the JSON tab, enter the following, replacing `<rolename>` with the role name:

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::*:role/<rolename>"
        }
    ]
}
```

4. From the user summary screen, open the Security credentials tab and click on Create access key.

5. Save the Access key ID and Secret access key.

6. From the user summary screen, open the Permissions tab and click Add permissions. Attach the AWSOrganizationsReadOnlyAccess policy.

7. (Optional) Attach the AmazonEC2ReadOnlyAccess policy to your organization's management account if it has EC2 instances you would like to sync.

**Step 2: Add the AWS credential to Rumble**

1. Go to the Credentials page in Rumble and click Add Credential.
2. Provide a name for the credential, like AWS EC2.
3. Choose AWS Access & Secret from the list of credential types.
4. Provide the following information:
   - **AWS access key** - Access key ID obtained from User > Security credentials > Create access key.
   - **AWS secret access key** - Secret access key obtained from User > Security credentials > Create access key.
   - Click the Use cross-account access for my AWS organization checkbox if you want to connect to all accounts in your organization.
   - **AWS role** - the name of the role to assume for all accounts. It should be named the same across accounts.
   - Select the region(s) that you want to sync.
5. If you want other organizations to be able to use this credential, select the Make this a global credential option. Otherwise, you can configure access on a per organization basis.
6. Save the credential. You're now ready to set up and activate the connection to bring in data from AWS.
Step 3: Set up and activate the AWS EC2 connection to sync data

After you add your AWS credential, you'll need to set up a connection to sync your data from AWS. A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where the data is organized.

1. Activate a connection to AWS. You can access all available third-party connections from your inventory or tasks page.
2. Choose the credential you added earlier. If you don’t see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like AWS sync.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under Task configuration, choose the site you want to add your assets to. All newly discovered assets will be stored in this site. You can also choose to Automatically create a new site per VPC, and Rumble will take care of creating the sites for newly discovered assets.
6. Under Service options, select the services you would like to sync data from. You must choose at least one.
7. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the Scheduled tasks to see when the next sync will occur.

Step 4: View AWS assets

After a successful sync, you can go to your inventory to view your AWS assets. These assets will have an AWS icon listed in the Source column.

To filter by AWS assets, consider running the following queries:

- View all AWS assets:
  source:aws

- View all AWS EC2 instances:
  source:aws AND has:"@aws.ec2.instanceID"

- View all AWS Elastic Load Balancers:
  source:aws AND (has:"@aws.elb.loadBalancerArn" OR has:"@aws.elb.loadBalancerArn")

Click into each asset to see its individual attributes. Rumble will show you the attributes returned by the AWS APIs.
**AWS EC2 instance attributes**

Rumble will enrich your assets with the following attributes, if the information is available:

id
instanceID
privateIP
privateDNS
publicIP
publicDNS
subnetID
vpcID
architecture
bootMode
hypervisor
imageID
instanceLifecycle
instanceType
ipv4
ipv6
kernelID
keyName
launchTimeTS
macs
outpostARN
platform
ramdiskID
rootDeviceName
rootDeviceType
spotInstanceRequestID
state
affinity
availabilityZone
groupName
hostID
hostResourceGroupArn
spreadDomain
tenancy
virtualizationType

**AWS ELB attributes**

Rumble will enrich your assets with the following attributes, if the information is available:

accountId
availabilityZones
canonicalHostedZoneId
canonicalHostedZoneName
createdTimeTS
customerOwnedIpv4Pool
dnsName
healthCheck.healthyThreshold
healthCheck.interval
healthCheck.target
healthCheck.timeout
healthCheck.unhealthyThreshold
instances
ipAddressType
ipv4
ipv6
loadBalancerArn
loadBalancerName
macs
privateIPs
publicIPs
region
scheme
securityGroups
sourceSecurityGroup.groupName
sourceSecurityGroup.ownerAlias
state
subnets
tags
type
type
vpcID
Miradore MDM

Rumble Enterprise integrates with Miradore mobile device management to deliver greater visibility into your mobile assets. This integration imports data through the Miradore API to enrich your asset inventory. Syncing with Miradore allows you to view information about your device’s hardware, OS version, associated user, and more.

This integration imports all enrolled devices. It sets the Miradore-specific attributes and updates asset-level attributes including the hardware platform, operating system, hostname, and MAC address. Rumble is able to merge existing assets with Miradore data when the hostname or MAC address is the same. Miradore devices can also be manually merged into Rumble assets using the Merge button on the Asset Inventory screen.

Getting started

To set up the Miradore integration, you’ll need to:

1. Login to your Miradore web portal and create a new API key.
2. Add the Miradore credential to Rumble, which includes the endpoint hostname and API key.
3. Activate the Miradore connection to sync your data with Rumble.

Requirements

Before you can set up the Miradore integration:

- Verify that you have Rumble Enterprise.
- Make sure you have access to the Miradore MDM portal as an administrator.

Step 1: Create a Miradore API key.

1. Log into the your customer-specific Miradore endpoint as an administrator (ex: yourcompany.online.miradore.com).
2. Click on the System > Infrastructure diagram link from the left-side navigation.
3. Scroll down to the find the API node in the diagram. Click this and choose Create API key.
4. Give this new key a name and copy the secret value.
Step 2: Add the Miradore API key to Rumble.

1. Go to the Credentials page in Rumble and click Add Credential.
2. Provide a name for the credential, like Miradore MDM.
3. Choose Miradore MDM API Key from the list of credential types.
4. Provide the following information:
   - **Name** - Give this credential a unique name (ex: Miradore)
   - **Miradore endpoint hostname** - This is the hostname (without https:// or the path) for your Miradore portal.
   - **Miradore API key** - This is the API key created in step 1.
5. Save the credential. You’re now ready to set up and activate the connection to bring in data from Miradore.

Step 3: Set up and activate the Miradore MDM connection to sync data

After you add your Miradore credential, you'll need to set up a connection to sync your data. A connection requires you to set a schedule and choose a default site. The schedule determines when the sync occurs, and the site determines where the data is organized.

1. Activate a connection to Miradore. You can access all available third-party connections from your inventory or tasks page.
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like Miradore sync.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under Task configuration, choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
6. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the Scheduled tasks to see when the next sync will occur.

**Step 4: View Miradore assets**

After a successful sync, you can go to your inventory to view your Miradore assets. These assets will have a Miradore icon listed in the Source column.

To view all your Azure assets, run the following query:

- **View all Miradore assets**:
  ```none```
  source:miradore
  ```

Click into each asset to see its individual attributes. Rumble will show you the attributes returned by the Miradore API.

**Miradore attributes**

Rumble will enrich your assets with the following attributes (specific attributes depend on the device OS):

```none```
id
ip
mac
platform
status
last Reported At
osVersion Name
os.language
os.platform
os.version
device.bluetoothMAC
device.deviceName
device.deviceType
device.manufacturer
device.model
device.product Name
device.serialNumber
device. softwareVersion
device.storeAccount Active
device.UDID
```
device.wifiMAC
user.email
user.name
Censys Search & Data

Rumble supports importing assets from the Censys Search API and the Censys Internet Dataset.

- Importing assets from the Censys Search API
- Importing assets from the Censys Universal Internet Dataset

Censys Search API

To get started with the Censys Search API, you will need to register for a Censys Search account. Once you have done so, you can find your API credentials in the My Account section.

In Rumble, go to the Credentials page, and click Add Credential. Select Censys Search API Key as the credential type, and enter your API ID and API secret.

You can now go to your asset inventory, click the Connect button, and choose Censys Search API. Select the credential you just created from the Censys Search credential dropdown.

Configuration

There are two modes for connecting Rumble to the Censys Search API.

- **Custom Query** mode - Rumble runs a Censys search query you specify, and then imports all of the results into Rumble. The search query should be in Censys Search Language. It is a good idea to test your query using the main Censys Search 2.0 interface before running an import task.

- **All Assets** mode - Rumble assembles a list of public IP addresses from all of the assets in the selected site, and then uses the API to find Censys Search information for those addresses. The information found is imported into Rumble and merged into the appropriate assets.

As with a Rumble scan, you'll need to select a site to contain the scan data. The usual task scheduling options are available.

When you have finished editing the Censys Search configuration, click Activate Connection.
Censys Universal Internet Dataset

To get started with the Censys Universal Internet Dataset API, you will need a paid Censys Data account and the associated API credential. You can find your API credentials in the My Account section.

The dataset can be downloaded by following the instructions in the Censys documentation. The Search API is used to get a list of files for a given date and those individual files should downloaded into a local directory backed by SSD or NVMe storage.

Creating the database

The raw files are in Apache Avro format and need to be converted into a database for efficient queries.

To process Censys data files, you use the Rumble CLI scanner's censys-db-convert command. This command takes two parameters:

- The path to a directory containing the .avro files from Censys
- The path to write the computed database

$ nice rumble-scanner censys-db-convert /home/censys/avro /home/censys/db

The default configuration requires substantial computing resources:

- At least 8 CPU cores, but 16 or more is better
- At least 64GiB of RAM, but more is better
- At least 3Tb of storage backed by SSD or NVMe (1Tb+2Tb or single volume)

An AWS m5.4xlarge with a 3Tb GP2 SSD volume meets these requirements and can process a full dataset (single day) in about 13 hours. The resulting database is about twice the size of the source data (1.3TiB database from 640GiB of Avro). Using the database requires additional disk overhead and overprovisioning the storage also improves throughput.

Querying the database

After the Avro files have been converted to a local database, the censys-db command can be used to import data into Rumble.

The scanner queries the local database, and writes a file in Rumble scan format containing the appropriate host records. By default, the file has a name matching censys-* .rumble.gz and is written to the current directory. Alternatively you can
specify an output filename with the --output-raw option, as if performing a Rumble scan.

The Rumble scan file can be uploaded to the Rumble console like any other scan file.

If you have more IP addresses or CIDRs than will fit on a command line, you can use the --input-targets option to specify that the scanner should read them from a file. The file is expected to be ASCII text, and contain CIDRs or IP addresses separated by whitespace (which can include newlines).

You can also use the scanner to process data, upload it, and then delete the scan data file if everything succeeded. For example:

```bash
% rumble-scanner censys-db /home/censys/db \
  12.216.190.0/24 --upload --api-key=YOUR_ORGANIZATION_API_KEY \
  --upload-site="Primary site"
```

If you are using self-hosted Rumble, you can use the --api-url option to specify your console's API endpoint.

The censys command also supports the --verbose option, which will make it list host addresses as they are written to the output file.

### Creating a local Censys Search API server

The computed database can also be used to serve a limited, local version of the Censys Search API using the Rumble scanner's censys-db-server command.

```bash
$ rumble-scanner censys-db-server /home/censys/db
```

This will start a local web service on port 55555 by default (changeable via --port <val>) that responds to the /api/v2/hosts/search and /api/v2/hosts/<ip> endpoints. Once this server is running, it can be queried using the Rumble Censys Search API connector, and through other HTTP clients, such as curl:

```bash
$ curl http://127.0.0.1:55555/api/v2/hosts/search?q=ip%3A8.8.8.0/24
$ curl http://127.0.0.1:55555/api/v2/hosts/8.8.8.8
```

### Querying the raw Avro files without database processing

Rumble also supports direct queries of the unprocessed Avro files. These queries are slow and may take hours or days to complete depending on the query and local storage speed. To query the raw Avro files, you use the Rumble scanner's censys command. It takes any number of arguments, which can be:

- Names of Avro files, which must end in .avro
- CIDRs or IP addresses to search for in the files
The scanner reads the Avro files specified, and writes a file in Rumble scan format containing the appropriate host records. By default, the file has a name matching censys-*\_rumble\_gz and is written to the current directory. Alternatively, you can specify an output filename with the \texttt{--output-raw} option, as if performing a Rumble scan.

The Rumble scan file can be uploaded to the Rumble console like any other scan file.

If you have more IP addresses or CIDRs than will fit on a command line, you can use the \texttt{--input-targets} option to specify that the scanner should read them from a file. The file is expected to be ASCII text, and contain CIDRs or IP addresses separated by whitespace (which can include newlines).

You can also use the scanner to process data, upload it, and then delete the scan data file if everything succeeded. For example:

\begin{verbatim}
% rumble-scanner censys universal-internet-dataset-20210923-000000000000.avro \
12.216.190.0/24 --upload --api-key=YOUR\_ORGANIZATION\_API\_KEY \
--upload-site="Primary site"
\end{verbatim}

If you are using self-hosted Rumble, you can use the \texttt{--api-url} option to specify your console's API endpoint.

The censys command also supports the \texttt{--verbose} option, which will make it list host addresses as they are written to the output file.
Leveraging the API

Rumble provides three primary APIs as well as integration-specific endpoints:

- The Export API provides read-only access to a specific organizations.
- The Organization API provides read-write access to a specific organizations.
- The Account API provides read-write access to all account settings and organizations.

To get started, you will need an export token or API key. Export API tokens and Organization API keys can be generated by going to the Organizations section in the Rumble web console, clicking on the appropriate organization name, and scrolling down to the export token or API keys section. A button there will let you generate a secure API key, in the form of a long random token. You must have administrator access to generate API keys.

Account API keys are generated from the Accounts settings page. Note that the Account API requires an Enterprise license.

Once you have generated a token, your REST client should use it with the `Authorization: Bearer standard header` to authenticate.

To use an Account API key with the Organization API, specify the additional parameter `_oid=[organization-id]` in the query parameters.

Please see the Swagger documentation and Rumble OpenAPI specification for details on the individual API calls.
Data formats

Rumble consumes and produces a handful of data formats. This page provides examples of these formats and describes the fields and use cases for each.

Formats

- Scan data (sample)
- Asset data (sample)
- Change reports (sample)

Scan data

The raw output produced by the Rumble Explorer and the Rumble Scanner is the scan data. This is newline-delimited JSON – JSONL – that represents the unprocessed output of the scan engine. This format is returned when downloading the task data for an explorer-run scan and correlates to the scan.rumble.gz file created by the command-line scanner. The Rumble Inventory view is built by processing scan data in chronological order to create the current state at a given point in time.

Scan data can be imported into an existing site through the Inventory Import menu of the web console and through the --import parameter of the command-line scanner. Each line of the file is a JSON object that specifies a type and a 64-bit Unix timestamp.

The example below is the raw scan data for a single Apple Mac Mini:

```json
```
The data contains four types of objects:

- **Scan config**: The \{"type": "config\}" object contains the full set of parameters for the scan as well as the version of the scan engine, and on Windows, the version of npcap installed. This record is used to determine the scan targets, which is used by the analysis engine to determine whether a given IP address was in scope.

- **Scan status**: The \{"type": "status\}" object contains diagnostic output from the scan engine. This can highlight issues that occurred while the scan was running.

- **Scan stats**: The \{"type": "stats\}" object represents point in time statistics for the scan. This will include the number of packets sent, received, and the progress estimate.

- **Scan result**: The \{"type": "result\}" object is target response for a specific probe. This can include TCP SYN+ACK replies, ICMP replies, or the result of application-layer probes, such as SNMP query responses, or HTTP screenshots. Scan Results are analyzed and correlated to create to the Asset Data format.

The scan stats sub-fields are defined below:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpu</td>
<td>CPU Core Percent * 100. 100% of one core would be 100000.</td>
</tr>
<tr>
<td>elapsed</td>
<td>The number of seconds since the scan started.</td>
</tr>
<tr>
<td>fdcount</td>
<td>The number of open file descriptors.</td>
</tr>
<tr>
<td>memory</td>
<td>The current memory usage in bytes.</td>
</tr>
<tr>
<td>progress</td>
<td>The estimated progress as a percentage (90 = 90%).</td>
</tr>
<tr>
<td>rateLimitTime</td>
<td>The number of Unix nanoseconds spent idling in the rate limiter.</td>
</tr>
<tr>
<td>recv</td>
<td>The number of packets received from the network.</td>
</tr>
<tr>
<td>recvBytes</td>
<td>The number of bytes received from the network.</td>
</tr>
<tr>
<td>recvError</td>
<td>The number of errors receiving from the network.</td>
</tr>
<tr>
<td>recvRate</td>
<td>The average packet receive rate for the scan.</td>
</tr>
<tr>
<td>resultCount</td>
<td>The total number of findings from the scan.</td>
</tr>
<tr>
<td>routines</td>
<td>The number of internal goroutines in the scan engine.</td>
</tr>
<tr>
<td>secondsLeft</td>
<td>The estimated seconds left to complete the scan.</td>
</tr>
<tr>
<td>sent</td>
<td>The number of packets sent the network.</td>
</tr>
</tbody>
</table>
sentBytes | The number of bytes sent the network.
--- | ---
sentError | The number of errors sending to the network.
sentRate | The average packet send rate for the scan.
startTime | The Unix timestamp in nanoseconds of when the scan started.

The scan result object type contains the following fields in addition to type and ts:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>The IP address associated with the response.</td>
</tr>
<tr>
<td>name</td>
<td>An optional hostname returned as part of this response.</td>
</tr>
<tr>
<td>port</td>
<td>The TCP or UDP port or zero for other protocols.</td>
</tr>
<tr>
<td>proto</td>
<td>The transport protocol, one of arp, icmp, tcp, or udp.</td>
</tr>
<tr>
<td>probe</td>
<td>The specific internal probe name that returned this response.</td>
</tr>
<tr>
<td>info</td>
<td>The result details object where all keys and values are strings.</td>
</tr>
</tbody>
</table>

The info object contains probe-specific response data. The key names are typically in the format of probe.subfield, with a few exceptions, and the values are always strings, even for numeric and array content. Multiple values for a key are represented as a tab-delimited array. Empty values are never reported for info keys. A given scan may return multiple result objects for a single probe, sometimes with duplicate values. These responses are correlated, deduplicated, and merged during the next phase of processing.

**Asset data**

The correlated and fingerprinted assets shown in the web console Inventory view and in the assets.jsonl file produced by the Rumble Scanner are the **asset data**. This data represents the state of each unique asset at a point in time and is built up by processing one or more sets of scan data.

Rumble supports a few variants of the asset data, including line-delimited JSON (JSONL), standard JSON documents, and a simplified CSV export. The JSONL format is the easiest to work with as it supports incremental processing without having to load the entire response into memory.

The example below is the correlated asset data for a scan of a single Apple Mac Mini:

```
{"id":"b73f8e09-78a6-4d2b-979d-e63908f28251","created_at":1597259778,"updated_at":1597259778,"organization_id":"b7fb13a7-701d-4ca5-b0e6-6f28f06cc866","site_id":"52d60c51-8dee-4f09-94e5-2dee30050a25","alive":true,"last_seen":1597259750,"first_seen":1597259738,"detected_by":"arp","type":"Desktop","os":"Apple macOS","os_version":"10.15","hw":"Apple Mac Mini (Late 2018)","addresses":["192.168.0.5"],"addresses_extra":null,"macs":null,"mac_vendors":null,"names":null,"mac":null}
```
Asset Data uses a number of data types for top-level fields, including string arrays, objects, strings, and integers. Rumble tracks multiple IP addresses and MACs per asset and these are represented as arrays. For asset-level attributes and services, these are stored as objects with additional structure. Assets are uniquely identified by the id field (a V4 UUID) and nearly every other field can be changed between scans, as assets move around the network, change IPs, and open and close services.

Every asset belongs to an Organization and a Site within that Organization.

The core asset data fields are defined below.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique ID of this asset defined as a v4 UUID</td>
</tr>
<tr>
<td>updated_at</td>
<td>The asset last update time represented as a 64-bit Unix timestamp in seconds.</td>
</tr>
<tr>
<td>organization_id</td>
<td>The Organization identifier defined as a v4 UUID.</td>
</tr>
<tr>
<td>site_id</td>
<td>The Site identifier defined as a v4 UUID.</td>
</tr>
<tr>
<td>alive</td>
<td>A boolean indicating whether this asset was found during the last scan of the site.</td>
</tr>
<tr>
<td>last_seen</td>
<td>The time the asset last responded represented as a 64-bit Unix timestamp in seconds.</td>
</tr>
<tr>
<td>detected_by</td>
<td>The protocol used to first detect that this asset was alive during the last scan.</td>
</tr>
<tr>
<td>type</td>
<td>A classification that represents a guess of the asset’s purpose.</td>
</tr>
<tr>
<td>os</td>
<td>The operating system name as determined by the fingerprinting engine.</td>
</tr>
<tr>
<td>os_version</td>
<td>The operating system version as determined by the fingerprinting engine.</td>
</tr>
<tr>
<td>hw</td>
<td>The hardware definition as determined by the fingerprinting engine.</td>
</tr>
<tr>
<td>addresses</td>
<td>An array of IP (v4/v6) addresses for the asset that were within the scan scope.</td>
</tr>
<tr>
<td>addresses_extra</td>
<td>An array of IP (v4/v6) addresses for the asset that were outside the scan scope.</td>
</tr>
<tr>
<td>macs</td>
<td>An array of MAC addresses associated with this asset.</td>
</tr>
<tr>
<td>mac_vendors</td>
<td>An array of MAC address vendors associated with this asset.</td>
</tr>
<tr>
<td>names</td>
<td>An array of unique hostnames associated with this asset (uppercase).</td>
</tr>
<tr>
<td>domains</td>
<td>An array of unique domain names associated with this asset (uppercase).</td>
</tr>
<tr>
<td>tags</td>
<td>A text representation of the user-specified tags associated with this asset.</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>attributes</td>
<td>An object containing a map key-value string attributes for this asset.</td>
</tr>
<tr>
<td>services</td>
<td>An object containing each associated service with the key representing the service description.</td>
</tr>
<tr>
<td>credentials</td>
<td>An object containing a map of any associated credentials (SNMP v2/v3).</td>
</tr>
<tr>
<td>rtts</td>
<td>An object containing a map of round-trip measurement times in milliseconds.</td>
</tr>
<tr>
<td>service_count</td>
<td>A sum of TCP, UDP, ARP, and ICMP services.</td>
</tr>
<tr>
<td>service_count_tcp</td>
<td>A sum of TCP services.</td>
</tr>
<tr>
<td>service_count_udp</td>
<td>A sum of UDP services.</td>
</tr>
<tr>
<td>service_count_arp</td>
<td>A sum of ARP services (0 or 1).</td>
</tr>
<tr>
<td>service_count_icmp</td>
<td>A sum of ICMP services.</td>
</tr>
<tr>
<td>lowest_ttl</td>
<td>The lowest observed source TTL for this asset.</td>
</tr>
<tr>
<td>lowest_rtt</td>
<td>The lowest observed source RTT for this asset.</td>
</tr>
<tr>
<td>last_explorer_id</td>
<td>The v4 UUID of the explorer responsible for the last scan of this asset.</td>
</tr>
<tr>
<td>last_task_id</td>
<td>The v4 UUID of the task responsible associated with the last scan of this asset.</td>
</tr>
<tr>
<td>newest_mac</td>
<td>The “newest” MAC address by registration date.</td>
</tr>
<tr>
<td>newest_mac_vendor</td>
<td>The “newest” MAC address vendor by registration date.</td>
</tr>
<tr>
<td>newest_mac_age</td>
<td>The “newest” MAC address registration date as a Unix timestamp in nanoseconds.</td>
</tr>
<tr>
<td>comments</td>
<td>User-specified comments associated with this asset.</td>
</tr>
<tr>
<td>service_ports_tcp</td>
<td>An array of strings representing the unique TCP ports found on this asset</td>
</tr>
<tr>
<td>service_ports_udp</td>
<td>An array of strings representing the unique UDP ports found on this asset</td>
</tr>
<tr>
<td>service_protocols</td>
<td>An array of strings representing the unique protocols found on this asset</td>
</tr>
<tr>
<td>service_products</td>
<td>An array of strings representing the unique products found on this asset</td>
</tr>
<tr>
<td>org_name</td>
<td>The name of the Organization associated with this asset</td>
</tr>
<tr>
<td>site_name</td>
<td>The name of the Site associated with this asset</td>
</tr>
<tr>
<td>explorer_name</td>
<td>The name of the Explorer associated with this asset</td>
</tr>
</tbody>
</table>

The `services` field contains string keys that contain the unique service identifier with values stored as strings. Multiple values may be stored as tab-delimited strings in the `service_values`. A typical service key looks like `192.168.0.5/22/tcp/`. The components of the service key name consist of address, port, transport, and virtual host (which can be blank).
Change reports

The Rumble platform calculates a change report after processing each scan. This is a JSON document available for download from the Task Details page with the following structure.

```json
{
    "new": { "<asset-UID>": { "Asset Data Fields":"" } },
    "online": { "<asset-UID>": { "Asset Data Fields":"" } },
    "offline": { "<asset-UID>": { "Asset Data Fields":"" } },
    "changed": { "<asset-UID>": { "Asset Data Fields":"" } },
    "summary": {
        "changed": 0,
        "new": 1,
        "offline": 0,
        "online": 0,
        "unchanged": 0
    },
    "truncated": false
}
```

The new, online, offline, and changed objects each contain keys consisting of the modified asset IDs with the values represented in the asset data format. The summary field indicates overall change statistics for this task. The truncated field is set to true if the change report is incomplete due to reaching the maximum change threshold (1000 asset changes today).
Frequently Asked Questions

Solutions to some common Rumble issues. Still can't find your answer? Let us know.

Why are there so many identical assets in my inventory?

Some enterprise routers and firewalls, like Cisco ASA devices, are designed to reply to all unexpected attempts on a particular port with a TCP reset (RST). On top of that, some routers listen to SIP traffic on all addresses and automatically respond to it.

Rumble will generally detect when a router or firewall is replying to every connection attempt and avoid creating assets based on those responses. However, if you have a network appliance that Rumble doesn't detect is spoofing response, there may be a substantial number of identical assets that will appear in your inventory.

Here are a few workarounds if you can't prevent your device from replying to all connections:

- Exclude the ports the device responds to from the scan configuration.
- Exclude all or part of the router's IP address range from the scan.
- Create a post-scan rule to delete any assets within the subnet that have the affected ports open.

These systems will respond to all requests on 1720/tcp and often 5060/tcp as well. Rumble tries to automatically detect and avoid most of the SIP helper implementations, but can't always do so without possibly losing real results. If you need help deleting unwanted records, please contact our support team.

How do I run Rumble without crashing my router?

The likely issue is your router is stateful, and it is keeping track of every connection going through it. Since our scanning process involves thousands of attempted connections, your router likely ran out of available stateful sessions. This usually occurs when a router is using Network Address Translation (NAT) or is acting as a stateful security firewall.

If this happens, here's what you can do:

- Avoid scanning across routed networks (wired and WiFi, multiple VLANs, etc) by deploying additional Explorers.
- Reduce the Max group size in your scan configuration. This limits the number of targets Rumble can scan at once, which correlates to the number of connections the router sees. Default is 4096.
• Reduce the scan speed. This will give failed connections more time to expire before new ones are attempted.

**How do I scan VMware virtual machines without crashing the host?**

Rumble can be used to scan VMware virtual machines. However, there are some precautions you should take.

VMnet interfaces normally use Network Address Translation (NAT) to route traffic between the host system and the virtual machines. The VMware software effectively operates as a stateful router. As explained above, this can cause problems when Rumble tries to open thousands of connections.

For scanning VMware systems, the best option is to deploy a Rumble Explorer inside VMware, on a virtual machine connected to the VMnet you want to scan. That explorer should be able to scan all VMs on the same VMnet without VMware needing to track all of the connections.

**Why didn’t the Rumble Explorer capture screenshots?**

The Rumble Explorer needs a working install of Google Chrome to obtain screenshots. To check for Google Chrome, the explorer looks in the following locations on each OS.

**Windows**

The Rumble Explorer looks for Chrome on Windows in:

c:\Program Files (x86)\Google\Chrome\Application\chrome.exe

The explorer also checks the following environment variables:

• ProgramFiles(x86)
• ProgramFiles
• ProgramW6432

Each may list another directory, in which case the explorer looks in \Google\Chrome\Application\chrome.exe under each of those directories as well.

To find what the environment variables are set to, open a Windows command prompt and entering the command set.

For a default Windows 10 install, the default value of ProgramFiles and ProgramW6432 is c:\Program Files, which means the explorer also checks the following location:

c:\Program Files\Google\Chrome\Application\chrome.exe
This is the default location for Chrome on a 64 bit Windows 10 system.

**MacOS**

On macOS, the explorer checks for Google Chrome in the following locations:

- /Applications/Google Chrome.app/Contents/MacOS/Google Chrome
- /Applications/Google Chrome Canary.app/Contents/MacOS/Google Chrome Canary
- /Applications/Chromium.app/Contents/MacOS/Chromium

**Linux**

On Linux systems, the explorer checks for Google Chrome at the following locations:

- /usr/bin/google-chrome
- /usr/bin/google-chrome-beta
- /usr/bin/google-chrome-unstable
- /usr/local/bin/chrome
- /usr/bin/chrome
- /opt/google/chrome/google-chrome
- /usr/bin/chromium-browser
- /usr/bin/chromium

**Chrome is installed, but screenshots still don’t work**

If Google Chrome is installed in one of the standard locations, but isn’t being found, it could be a permissions issue. It is also possible for Chrome to fail to run for other reasons, such as a corrupt Chrome profile. The next thing you can do is download the Rumble Scanner and run a scan to disk, which will write a log file that will have more detail about the scan operation.

**What protocols does Rumble scan for?**

Rumble supports the following list of protocols:

```
<table>
<thead>
<tr>
<th>airplay</th>
<th>adb</th>
<th>arp</th>
<th>activemq</th>
</tr>
</thead>
<tbody>
<tr>
<td>amqp</td>
<td>backupexec</td>
<td>bacnet</td>
<td>crestron</td>
</tr>
<tr>
<td>cassandra</td>
<td>checkmk</td>
<td>chromecast</td>
<td>citrix</td>
</tr>
<tr>
<td>click</td>
<td>couchdb</td>
<td>dcerpc</td>
<td>dns</td>
</tr>
<tr>
<td>dotnet-remoting</td>
<td>drbd</td>
<td>drobo-nasd</td>
<td>dtls</td>
</tr>
<tr>
<td>echo</td>
<td>elasticsearch</td>
<td>epm</td>
<td>ftp</td>
</tr>
<tr>
<td>giop</td>
<td>gpd</td>
<td>http</td>
<td>http2</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>ike</th>
<th>imap</th>
<th>ipmi</th>
<th>influxdb</th>
</tr>
</thead>
<tbody>
<tr>
<td>infinispan</td>
<td>irc</td>
<td>java-rmi</td>
<td>jdbc-hsqldb</td>
</tr>
<tr>
<td>jetdirect</td>
<td>jms</td>
<td>landesk</td>
<td>lockdownnd</td>
</tr>
<tr>
<td>lpd</td>
<td>l2t</td>
<td>mdns</td>
<td>memcache</td>
</tr>
<tr>
<td>minecraft</td>
<td>mountd</td>
<td>mongodb</td>
<td>mssql</td>
</tr>
<tr>
<td>mysql</td>
<td>neo4j</td>
<td>netbios</td>
<td>natpmp</td>
</tr>
<tr>
<td>ntp</td>
<td>nfs</td>
<td>openvpn</td>
<td>oracledb</td>
</tr>
<tr>
<td>pca</td>
<td>pptp</td>
<td>pop3</td>
<td>postgresql</td>
</tr>
<tr>
<td>rdp</td>
<td>redis</td>
<td>rexec</td>
<td>riak</td>
</tr>
<tr>
<td>riak-http</td>
<td>rpcbind</td>
<td>rsyncd</td>
<td>rtsp</td>
</tr>
<tr>
<td>sip</td>
<td>smb1</td>
<td>smb2</td>
<td>smb3</td>
</tr>
<tr>
<td>snmp</td>
<td>smtp</td>
<td>ssdp</td>
<td>sonicwall-sgms</td>
</tr>
<tr>
<td>spice</td>
<td>ssh</td>
<td>subversion</td>
<td>sunrpc</td>
</tr>
<tr>
<td>teamviewer</td>
<td>telnet</td>
<td>tftp</td>
<td>tls</td>
</tr>
<tr>
<td>ubnt</td>
<td>upnp</td>
<td>vmauthd</td>
<td>vnc</td>
</tr>
<tr>
<td>wsd</td>
<td>wsman</td>
<td>zabbix-agent</td>
<td>zookeeper</td>
</tr>
</tbody>
</table>

**What ports does Rumble scan?**

Rumble scans the following ports by default:

<table>
<thead>
<tr>
<th>1</th>
<th>7</th>
<th>9</th>
<th>13</th>
<th>19</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>25</th>
<th>37</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>43</td>
<td>49</td>
<td>53</td>
<td>69</td>
<td>70</td>
<td>79</td>
<td>80</td>
<td>81</td>
<td>82</td>
</tr>
<tr>
<td>83</td>
<td>84</td>
<td>85</td>
<td>88</td>
<td>102</td>
<td>105</td>
<td>109</td>
<td>110</td>
<td>111</td>
<td>113</td>
</tr>
<tr>
<td>119</td>
<td>123</td>
<td>135</td>
<td>137</td>
<td>139</td>
<td>143</td>
<td>161</td>
<td>179</td>
<td>222</td>
<td>264</td>
</tr>
<tr>
<td>384</td>
<td>389</td>
<td>402</td>
<td>407</td>
<td>443</td>
<td>444</td>
<td>445</td>
<td>465</td>
<td>500</td>
<td>502</td>
</tr>
<tr>
<td>512</td>
<td>513</td>
<td>515</td>
<td>523</td>
<td>524</td>
<td>540</td>
<td>548</td>
<td>554</td>
<td>587</td>
<td>617</td>
</tr>
<tr>
<td>623</td>
<td>631</td>
<td>636</td>
<td>689</td>
<td>705</td>
<td>771</td>
<td>783</td>
<td>873</td>
<td>888</td>
<td>902</td>
</tr>
<tr>
<td>903</td>
<td>910</td>
<td>912</td>
<td>921</td>
<td>990</td>
<td>993</td>
<td>995</td>
<td>998</td>
<td>1000</td>
<td>1024</td>
</tr>
<tr>
<td>1030</td>
<td>1035</td>
<td>1080</td>
<td>1089</td>
<td>1090</td>
<td>1091</td>
<td>1098</td>
<td>1099</td>
<td>1100</td>
<td>1101</td>
</tr>
<tr>
<td>1102</td>
<td>1103</td>
<td>1128</td>
<td>1129</td>
<td>1158</td>
<td>1199</td>
<td>1211</td>
<td>1220</td>
<td>1270</td>
<td>1234</td>
</tr>
<tr>
<td>1241</td>
<td>1300</td>
<td>1311</td>
<td>1352</td>
<td>1433</td>
<td>1440</td>
<td>1468</td>
<td>1494</td>
<td>1521</td>
<td>1530</td>
</tr>
<tr>
<td>1533</td>
<td>1581</td>
<td>1582</td>
<td>1583</td>
<td>1604</td>
<td>1723</td>
<td>1755</td>
<td>1811</td>
<td>1883</td>
<td>1900</td>
</tr>
<tr>
<td>2000</td>
<td>2049</td>
<td>2082</td>
<td>2083</td>
<td>2100</td>
<td>2103</td>
<td>2121</td>
<td>2181</td>
<td>2199</td>
<td>2207</td>
</tr>
<tr>
<td>2222</td>
<td>2323</td>
<td>2362</td>
<td>2375</td>
<td>2379</td>
<td>2380</td>
<td>2381</td>
<td>2525</td>
<td>2533</td>
<td>2598</td>
</tr>
<tr>
<td>2601</td>
<td>2604</td>
<td>2638</td>
<td>2809</td>
<td>2947</td>
<td>2967</td>
<td>3000</td>
<td>3037</td>
<td>3050</td>
<td>3057</td>
</tr>
<tr>
<td>3128</td>
<td>3200</td>
<td>3268</td>
<td>3269</td>
<td>3217</td>
<td>3273</td>
<td>3299</td>
<td>3300</td>
<td>3306</td>
<td>3311</td>
</tr>
<tr>
<td>3312</td>
<td>3351</td>
<td>3389</td>
<td>3460</td>
<td>3500</td>
<td>3628</td>
<td>3632</td>
<td>3690</td>
<td>3780</td>
<td>3790</td>
</tr>
<tr>
<td>3817</td>
<td>4000</td>
<td>4092</td>
<td>4322</td>
<td>4343</td>
<td>4369</td>
<td>4433</td>
<td>4443</td>
<td>4444</td>
<td>4445</td>
</tr>
<tr>
<td>4567</td>
<td>4659</td>
<td>4679</td>
<td>4730</td>
<td>4786</td>
<td>4840</td>
<td>4848</td>
<td>5000</td>
<td>5022</td>
<td>5037</td>
</tr>
<tr>
<td>5038</td>
<td>5040</td>
<td>5051</td>
<td>5060</td>
<td>5061</td>
<td>5093</td>
<td>5168</td>
<td>5222</td>
<td>5247</td>
<td>5250</td>
</tr>
<tr>
<td>5275</td>
<td>5347</td>
<td>5351</td>
<td>5353</td>
<td>5355</td>
<td>5392</td>
<td>5400</td>
<td>5405</td>
<td>5432</td>
<td>5433</td>
</tr>
<tr>
<td>5498</td>
<td>5520</td>
<td>5521</td>
<td>5554</td>
<td>5555</td>
<td>5560</td>
<td>5580</td>
<td>5601</td>
<td>5631</td>
<td>5632</td>
</tr>
<tr>
<td>5666</td>
<td>5671</td>
<td>5672</td>
<td>5683</td>
<td>5800</td>
<td>5814</td>
<td>5900</td>
<td>5920</td>
<td>5938</td>
<td>5984</td>
</tr>
<tr>
<td>5985</td>
<td>5986</td>
<td>5988</td>
<td>5989</td>
<td>6000</td>
<td>6001</td>
<td>6002</td>
<td>6050</td>
<td>6060</td>
<td>6070</td>
</tr>
</tbody>
</table>
Can I safely scan my IoT or OT environments?

Some organizations have IoT or OT equipment sensitive to high traffic rates or malformed packets that may have experienced issues with other scanning tools in the past, resulting in a “don’t scan” rule to be in effect.

Rumble is different, and should be able to scan in these environments. Rumble provides a lightweight active scan engine called an Explorer that can be deployed almost anywhere. Since the scan is active, there are no tap or span ports that need to be configured, nor device level agents that need to be installed, so you don’t have to modify your environment.

The Rumble Explorer was built with sensitive OT environments in mind. It is not based on any other commercial or open source tools such as nmap or masscan. The Explorer only sends normal traffic, nothing malformed that could cause an IDS or IPS to alert, or that might potentially crash a fragile system. Some of the controls in place also include:

- packets-per-second scan rates with sensible default values:
  - 1000 packets per second for overall maximum scan rate (adjustable; scan traffic is balanced across all hosts in the scan range)
  - 40 packets per second for per-host maximum scan rate (adjustable)
• IP and TCP port exclusions
• UDP service probes can be enabled or disabled individually
• The scan balances SYNs and ACKs and watches for port consumption issues on both the client & target
• Configurable max group size that limits the number of targets Rumble can scan at once, which correlates to the number of connections stateful devices such as firewalls or routers receive
• Only those TCP and UDP ports that provide actionable intelligence for fingerprinting a device are checked, not all 65535. This list is adjustable in case specialized equipment runs on a non-standard port (See the Port List above).
• Per port / protocol considerations engineered to avoid issues. Ex: Sending characters to port 9100 on a printer could print “garbage”. Rumble will collect a banner from some ports such as these but never actively probe them.

Many organizations opt to deploy the explorer to the same system that runs their vulnerability scans since there may already be allow-lists, full network connectivity, and considerations made for session table capacity on any session-aware middle boxes such as firewalls, proxies, or small routers. It may also be advisable to deploy additional explorers at remote sites to gather additional detail and avoid altogether any need to consider middle boxes.
Rumble release notes

Latest release notes

v2.9.14

2022-01-28

- HP iLOs will no longer be merged into their host assets when they share a MAC address.
- A bug that prevented services from displaying after a third-party import has been resolved.
- A bug that prevented Asset Modify rules from updating the HW field has been resolved.
- A bug that could cause the CLI scanner to stack trace has been resolved.
- The CLI Scanner censys-db sub-command now requires less memory.
- Fingerprint updates.

v2.9.13

2022-01-27

- Censys Avro files can now be converted to a database for faster lookups.
- Fingerprint updates.

v2.9.12

2022-01-25

- A regression that could lead to login errors after bulk permission updates has been fixed.
- A regression that removed the service names from the asset details page has been fixed.
- AWS internal hostnames are now reported in the asset name list.
- Fingerprint updates.

v2.9.11

2022-01-24

- Nmap XML exports are now much faster.
- Fingerprint updates.
v2.9.10
2022-01-20

- The scan engine now limits the SNMP enumeration speed to the Max Host Rate, reducing CPU usage on older switches.
- The scan engine now ignores additional cases of FortiGate HTTP interception.
- Fingerprint updates.

v2.9.9
2022-01-18

- The scan engine now accepts IPv6 addresses and resolves AAAA records for hostnames.
- The scan engine now skips protocol probes on TCP port 9106.
- A bug that prevented uploading very large scans has been fixed.
- Fingerprint updates.

v2.9.8
2022-01-14

- Added an option to export only selected assets, services, or wireless.
- Added a “Every N Hours” recurring task frequency option.
- Autocompletion of search keywords has been added for Organizations, Tasks, and Events.
- AWS and Azure connectors no longer set asset alive status and no longer are counted as offline or back online in the change report.
- Rules now show when they were last processed, whether they triggered their action, and any error that occurred as a result.
- The coverage report can be filtered by site.
- A bug that could allow duplicate CrowdStrike assets after an import has been fixed. Any resulting duplicates are eliminated on the next CrowdStrike task run.
- A bug which could lead to stalled rule processing has been fixed.
- A bug that prevented importing operating system information from CrowdStrike for some Linux devices has been fixed.
- A bug where scanning of some Lexmark printers interfered with the printer’s job queue has been fixed.
- Fingerprint updates.
v2.9.7
2022-01-10

- The scanner now supports configuration of reverse DNS timeouts and the SSH username.
- Scan tags can now be provided for scan import tasks.
- The closedPortsMap field has been removed from JSON exports.
- CrowdStrike connector tasks now move preexisting CrowdStrike-sourced assets into matching scanned assets across sites.
- A bug where task progress (on hover) could exceed 100% has been fixed.
- A bug that caused the Azure integration to occasionally skip public IPs has been fixed.
- A bug that caused a CrowdStrike connector task to send an API request exceeding length limits in specific instances has been fixed.
- Fingerprint updates.

v2.9.6
2021-12-23

- A bug that could lead to some events being processed incorrectly has been resolved.
- Event templates now truncate results correctly.

v2.9.5
2021-12-22

- A scan engine bug that could lead to an “invalid exclusions” error has been resolved.

v2.9.4
2021-12-21

- The Query search now supports result count selection and remembers the setting between views.
- The scan engine now correctly excludes broadcast addresses from the scan scope.
- The Azure connector now ignores canceled subscriptions automatically.
- The hostname selection logic has been improved for Canon printers.
- The Explorer service now starts up slightly faster on Windows.
- The Censys AVRO importer is now 4 to 8 times faster.
- Fingerprint updates.

v2.9.3
2021-12-15

- The CrowdStrike integration has been updated to improve correlation with existing assets.

v2.9.2
2021-12-13

- A bug that prevented some AWS organizations from working with STS AssumeRole has been fixed.
- A bug that persisted service products after asset changes has been fixed.
- A bug that hid the Task Change Report has been fixed.
- Improved product detection for Logstash and Neo4J.
- Fingerprint updates.

v2.9.1
2021-12-08

- The CrowdStrike integration now uses the Scroll API to better support large organizations.

v2.9.0
2021-12-06

This release is a roll-up of the 2.8.x updates in addition to the following changes.

- The new Site Comparison report shows differences in assets between two sites, which can be in different organizations.
- The team page now supports bulk user import and bulk permission management.
- The layer 2 topology report has been updated with a search filter, site filter, and visual improvements.
- The network bridges report has been updated with a site filter and visual improvements.
- A bug that caused the has_public search filter to flag certain IPv6 addresses has been fixed.
• A bug that caused project deletion to create new, blank organizations has been fixed.
• A bug that caused Censys imports to mark other assets as offline has been fixed.
• The bundled npcap version has been upgraded to 1.60.
• Fingerprint updates.

**v2.8.14**

2021-12-03

• A bug that prevented scans from running when non-loopback 127.x networks were present has been fixed.

**v2.8.13**

2021-11-24

• A regression in the TLS version enumeration has been fixed.
• Teredo addresses are no longer considered public IPs.
• Improved detection of Chromebooks and ChromeOS.
• Fingerprint updates.

**v2.8.12**

2021-11-21

• The self-hosted platform now supports internal proxies for external API connections.
• The self-hosted platform now supports internal webhook destinations for alerts.

**v2.8.11**

2021-11-19

• The annotator role is now available.
• Fingerprint updates.

**v2.8.10**

2021-11-18

• A bug that prevented the Azure connector from working in self-hosted mode has been resolved.
• The last hop calculation for the TCP traceroute is now more accurate.
• Fingerprint updates.

v2.8.9

2021-11-16

• Credentials can now be configured for single IP addresses and IP ranges in addition to CIDRs.
• The scan engine now performs a light traceroute when an open TCP port is found.
• The scan engine now tests for IP forwarding during scans of link-local targets.
• The scan engine now includes Rumble/2 in HTTP user-agent strings.
• The scan engine now limits ARP traffic to the Max Host Rate.
• The OS EOL date for Windows Server 2019 (1809) has been updated.
• Fingerprint updates.

v2.8.8

2021-11-12

• The self-hosted platform now supports custom CSP headers to support external resources.
• Tags with the case-insensitive key of “name” are now treated as additional hostnames.
• Tags containing spaces are now consistently handled by conversion to underscore.
• Meraki DNS interception is now ignored in a wider variety of configurations.
• The CrowdStrike integration now tracks the last 10 recent logins per asset.
• A regression in the Apple macOS end of life calculation has been fixed.
• A few missing icons have been restored to the inventory view.
• A panic in the FreeBSD scan processing chain has been fixed.
• Direct print services on 9002 are no longer fingerprinted.
• TLS versions are now consistently formatted.
• Fingerprint updates.

v2.8.7

2021-11-07

• A bug that caused daily scans to be scheduled incorrectly in UTC forward time zones has been resolved.
• The self-hosted platform now respects proxy settings for external data sources (CrowdStrike, etc.).
• Improved tvOS and Crestron fingerprints.
• Additional UPnP fingerprints.

v2.8.6
2021-11-06
• Improved tvOS, homepodOS, and bridgeOS fingerprints.

v2.8.5
2021-11-06
• The self-hosted installer now supports manual database configuration.
• The self-hosted platform now includes a database verify subcommand.
• Improved iOS device identification.
• Fingerprint updates.

v2.8.4
2021-11-05
• The VMware probe now handles vCenter instances configured with multiple datacenters.
• A race condition that could lead to Explorer updates mid-scan has been resolved.
• The RDP TLS fingerprint will no longer break matching during asset correlation.
• Stopped scans now indicate which user stopped them in the error message.
• Active probes are now disable for some Lantronix and Rockwell PLC ports.
• Fingerprint updates.

v2.8.3
2021-11-04
• SAML SSO now specifies that the required NameID Format is unspecified, for Azure AD compatibility.
• S3 storage operations which fail are retried.
• A bug in Azure credential validation is fixed.
• Additional detail has been added to the field help on the scan form.

v2.8.2
2021-11-02
• Recurring tasks no longer schedule jobs when the previous job is still queued.
• A bug that caused VMware-based OS detection to fail has been resolved.
• Fingerprint updates.

v2.8.1

2021-11-02

• A bug that caused out-of-scope assets to be marked as offline during scans has been resolved.

v2.8.0

2021-11-01

This release is a roll-up of the 2.7.x updates in addition to the following changes.

• Support for Censys Search API and Censys Data imports (Enterprise).
• Fingerprint updates.

v2.7.11

2021-10-27

• An issue with Windows explorer updates has been resolved.

v2.7.10

2021-10-24

• An issue with restrictive umasks on the self-hosted platform has been resolved.
• Support for VMware vCenter/ESXi virtual machine discovery (Enterprise).
• Asset merging from third-party data sources has been improved.
• TLS fingerprints are now reported as SHA256 hashes (base64).
• Credentials for SNMP v2/v3 and VMware can be managed globally.
• Serial numbers from A10 devices are now collected via SNMP.
• The SNMP v3 probe now supports multiple credentials.
• Scan configuration now has a Credentials tab.
• Detailed task logs can be downloaded.
• Light UX improvements and bug fixes.
• Fingerprint updates.

v2.7.9

2021-10-21
• A permissions issue with the self-hosted platform has been resolved.

v2.7.8

2021-10-15

• The Rumble Explorer on Windows now explicitly sets the service to automatic start.
• RFC 1918 scans can be launched from the main Scan menu (Professional/Enterprise).
• Fingerprint updates.

v2.7.7

2021-10-15

• Credential validation issues with Azure now log detailed errors.
• The self-hosted platform now supports Debian 9.
• Fingerprint updates.

v2.7.6

2021-10-13

• A bug that led to an application error during Azure connector configuration has been resolved.
• Fingerprint updates.

v2.7.5

2021-10-13

• A bug that prevented certain AWS and Azure assets from being imported has been resolved.
• Fingerprint updates.

v2.7.4

2021-10-08

• The self-hosted platform now supports Oracle Linux 7 and 8.
• Fingerprint updates.
v2.7.3

2021-10-07

- The AWS integration is now much faster for large numbers of accounts.

v2.7.2

2021-10-06

- A bug that made it difficult to update existing Azure tasks has been resolved.

v2.7.1

2021-10-05

- A bug that prevented the date picker from showing arrow icons has been resolved.
- A bug that made it difficult to update existing AWS tasks has been resolved.
- The sidebar is now collapsible using the chevron icon at the top.
- The “Processing” link states are now handled more consistently.

v2.7.0

2021-10-05

This release is a roll-up of the 2.6.x updates in addition to the following changes.

Integrations

- The Azure VM connector now supports multi-subscription and multi-directory access.
- The AWS EC2 connector now supports ELB load balancers as importable assets.
- Connector credentials are now automatically validated on save.
- The Splunk add-on now supports self-hosted console endpoints.
- The Splunk add-on now optionally imports asset services.
- The Splunk add-on has been updated to use jQuery 3.5.0.
- CrowdStrike asset merging has been improved.

Self-hosting

- The self-hosted installer now includes the explorer and scanner binaries.
- The self-hosted console now supports detailed TLS configuration.
The self-hosted console now runs as an isolated subprocess.
The self-hosted console no longer enforces API rate limits.

User experience

- Asset and service trends are now shown on the dashboard.
- The Scan configuration view has been overhauled and simplified.
- Imported scans are now tracked for the RFC 1918 coverage report.
- The Services inventory now supports new address-related search keywords.
- The Assets and Services inventory now support wildcard searches of hostnames with anchored patterns.
- The Screenshot inventory is now faster and shows the correct total count.
- A bug in the RFC 1918 coverage report that could lead to skipped IPs was fixed.

Authentication

- Users with standard accounts that authenticate using SSO are now converted into SSO-only accounts.
- SSO is supported for multiple domains using IdP or SP initiated authentication.
- The MFA challenge now provides a Retry button for browsers that require user interaction (Safari).
- The MFA enrollment now supports token or platform authentication mode as separate options.
- Admin users can now set the first and last names of other users.
- The Explorers, Scanners, MSI wrapper, and Verifier are now signed using a new EV certificate.

Scanner and fingerprinting

- Subnet ping and host ping are now included in the Professional tier.
- OS EOL tracking is now enabled for Windows 10 and APC firmware.
- Windows 10 and Server 2019 OS versions are now tracked by range.
- The Explorer and CLI scanner now detect and report an error when run within the WSL/WSL2 environments.
- The CLI scanner upgrade now supports the –force option.
- The scan engine now detects Bitdefender remotely.
- A regression in the ARP probe on newer Windows builds has been resolved.
- Fingerprint updates

v2.6.4

2021-09-16

- The scan engine now detects Azure’s OMI WSMAN implementation.
• Fingerprint updates.

v2.6.3

2021-09-16

• The scan engine now detects WSMAN, ADB, and InfluxDB services.
• Fingerprint updates.

v2.6.2

2021-09-14

• A regression in SMB v1 detection has been resolved.
• Fingerprint updates.

v2.6.1

2021-09-08

• A bug that prevented some Azure VMs from being imported has been fixed.

v2.6.0

2021-09-07

• Rumble Enterprise customers can now sync virtual machine inventory from the Microsoft Azure cloud.
• The CrowdStrike connector has been overhauled to improve asset merging and avoid duplicates.
• OS end-of-life dates for Windows, macOS, Ubuntu, Debian, and iLO assets are now tracked.
• The self-hosted version of Rumble now supports offline mode & offline updates.
• The self-hosted version of Rumble now supports RHEL 7 in offline mode.
• The scan engine now surfaces NFS exports via discovered mountd services.
• The scan engine now returns details for discovered PPTP services.
• The dashboard loads faster for large organizations.
• The UI now includes new, custom icons.
• Fingerprint updates.

v2.5.8

2021-08-30
- Stale asset expiration now applies to third-party sourced assets.
- The scan engine now reports PPTP services.
- Fingerprint updates.

**v2.5.7**

2021-08-27

- The CrowdStrike connector has been overhauled to improve merging and avoid duplicates.
- The scan engine now reports NFS exports.
- Fingerprint updates.

**v2.5.6**

2021-08-23

- The scan engine now implements the Cisco [layer 2 traceroute](https://github.com/Cisco/cisco-l2t) protocol thanks to Chris Marget's cisco-l2t project.
- TCP port 1720 is no longer included in the defaults. This may be re-enabled once H.323 is fully implemented.
- The scan engine now handles mangled SNMP responses better.
- The HTTP/2 protocol is now reported at the asset level.
- Fingerprint updates.

**v2.5.5**

2021-08-19

- A regression in the service attribute report has been fixed.
- The scan engine now reports additional SSH attributes.

**v2.5.4**

2021-08-19

- The Explorer and Scanner now support the Windows arm64 platform.
- A scan engine hang in the DCERPC probe has been resolved.
- Fingerprint updates

**v2.5.3**

2021-08-18
• Third-party data source attributes are now included in all exports.
• Third-party attributes now use the @source.type syntax for search.
• The Merge feature in the asset inventory is now more consistent.
• Large target exclusion lists are now supported for sites and scans.
• Unresolvable hostname excludes are now ignored automatically.
• The scan engine now records more information from McAfee ePO agents.
• Fingerprint updates

**v2.5.2**

2021-08-12

• Automatic queries are now available to Professional users as well as Enterprise.
• A bug that led to some Windows desktops having the wrong type has been fixed.
• CrowdStrike assets are now matched more accurately against Rumble assets.
• The scanner now skips active protocol detection on port 9999.
• Fingerprint updates

**v2.5.1**

2021-08-05

• The default TCP port list now includes more SolarWinds products as well as port 7676 for JMS/JMBroker.
• The estimated runtime for scans now takes into account the TCP port list (and excludes).
• Juniper switch fingerprinting now uses a Juniper-specific OID instead of sysDesc.
• Additional bogus SIP ALG services are now ignored by the Rumble scanner.
• A bug that prevented offline-agent events from being generated in certain situations has been resolved.
• A bug that could result in Explorers not reconnecting properly after an update has been resolved.
• A bug that showed an incorrect bandwidth calculation in the task view has been resolved.
• A bug that led to an error on CSV export with mixed-source assets has been resolved.
• Fingerprint updates.

**v2.5.0**

2021-08-03

• Rumble Enterprise customers can now sync AWS EC2 assets across accounts using STS roles.
Rumble Enterprise customers can now sync asset data from the CrowdStrike Falcon API.

- The scan engine now better differentiates between Windows workstation and server variants.
- The scan engine now detects various asset attributes and services using DCERPC.
- The scan engine now detects multi-homed assets using DCERPC.
- The dashboard can now show stats across all sites or just a specific site.
- The new Unmapped MAC report highlights unscanned assets by switch port.
- The Reports page has been improved with a new layout and inline search.
- The Queries tab has been moved to a new navigation item.
- Fingerprint updates.

**v2.4.4**

2021-07-26

- The superuser role is now available as a default permission for SSO users.
- The scan engine now gathers data from the Windows DCERPC endpoint mapper.
- Fingerprint updates.

**v2.4.3**

2021-07-18

- Fingerprint updates.

**v2.4.2**

2021-07-14

- Layer-2 topology graphs are now available for environments using Fortinet switches.
- The Topology and Network Bridges graphs can now be exported as PNG images.

**v2.4.1**

2021-07-13

- This build fixes a bug in the Go runtime that could allow a remote attacker to cause a recoverable panic in the Rumble services and scan engine (CVE-2021-34558).
v2.4.0
2021-07-13

- Rumble Enterprise customers can now sync asset data from Amazon Web Services EC2 and Miradore MDM data sources.
- Rumble Enterprise customers can now self-host the platform on RHEL and CentOS distributions.
- Credentials are now managed at the account level with per-organization access.
- The Rumble self-hosted CLI offers new features and a better user experience.
- Fingerprint updates.

v2.3.5
2021-07-04

- A bug that prevented explorer upgrades and scan stop requests from processing while a scan was active has been fixed.
- A bug that led to the subnet ping mode missing subnets during large scans cases has been fixed.
- A bug that led to an explorer showing as offline unexpectedly has been fixed.
- Fingerprint updates.

v2.3.4
2021-06-26

- A regression in the Screenshots inventory tab has been fixed.
- Fingerprint updates.

v2.3.3
2021-06-24

- Tabs, fonts, and styles have seen a light update.
- A number of small UX bugs were addressed.
- Fingerprint updates.

v2.3.2
2021-06-16
• The RFC 1918 coverage report now supports a starting date that can be used to exclude older scans.
• The Your team page is now searchable, sortable, and supports bulk user actions.
• Fingerprint updates.

v2.3.1

2021-06-09

• A bug that prevented single-organization users from viewing sites and tasks has been fixed.
• A bug that led to offline assets not actually being marked as offline has been fixed.
• A bug that could prevent full enumeration of Cisco Catalyst switches has been fixed.
• Fingerprint updates.

v2.3.0

2021-06-08

• The new RFC 1918 coverage report highlights unscanned address space and hinted ranges.
• SNMP v3 enumeration of Cisco Catalyst switches now handles per-vlan port mappings.
• Fingerprint updates.

v2.2.5

2021-05-29

• Tags are now always displayed with = instead of : to match the search engine syntax.
• The Subnet Ping and Host Ping modes are now more reliable on large scans.
• Fingerprint updates.

v2.2.4

2021-05-26

• A bug that led to a stack trace in the rumblectl command for self-hosted mode has been fixed.
• Fingerprint updates.
**v2.2.3**

2021-05-17

- The self-hosted platform now removes older scanner/explorer binaries during updates.
- The scan engine now pulls layer-2 information from Force-10 switches.
- The scan engine now ignores CheckPoint SMTP and SIP interception.
- The scan engine now extracts hostnames from Zyxel switches.
- An invalid fingerprint for Cisco IP phones has been fixed.
- Multiple notifications can now trigger from a single event.
- Agent fields are now included in the scan-started event.
- Fingerprint updates.

**v2.2.2**

2021-05-16

- The scan engine now extracts additional information from Zyxel switches.
- The Explorers page now supports sorting, searching, and tagging.
- Fingerprint updates.

**v2.2.1**

2021-05-14

- The scan confirmation dialog now warns when a mix of public and private IPs are in the scope.
- The SNMP v3 probe now supports sha224, sha256, sha384, and sha512 authentication.
- The SNMP v3 probe now supports aes192, aes256, aes192c, and aes256c encryption.
- The self-hosted platform now includes a CLI to manage user accounts.
- Fingerprint updates.

**v2.2.0**

2021-05-11

Rumble 2.2.0 is a roll-up of previous 2.1.x releases along with some additional changes and features.
Web console

- Enterprise customers can now export an HP iLO report which includes serial numbers, physical hardware information, and other fields useful for warranty tracking and server inventory.
- Virtual machines now indicate the virtualization vendor in the asset hardware field and can be searched and filtered by vendor.
- Virtual machines are now labeled with an icon in the asset view, and router icons are displayed with the other icons.
- Virtual machine and legacy MAC prefixes are no longer used for age calculation, as they gave inaccurate results.
- A new report for virtual machine vendors has been added.
- Vendors in the NDAA Section 889 report have been expanded to include Aztech and subsidiaries.
- The Name tag can be used to set a preferred hostname for any asset.
- Tags in inventory views can now be clicked to search the inventory.
- The alert notification templates can now include information about the name and internal IP address of the explorer which carried out a scan.
- The alert rules list now shows which rules are currently enabled.
- Confirmation dialogs now require a typed response for destructive actions.
- Page layout has been improved for browser window widths between 920 and 1200 pixels.
- If a scan results in too many changes to list in the task report, the report now explains this.
- Progress bars now use standard meter elements for smoother updating and better accessibility.
- Icons and screenshots are now lazy-loaded to speed up initial page rendering.
- Task duration is now rounded up to the nearest minute.

Explorer and scan engine

- A better hostname is chosen for each asset by default.
- VLANs are now tracked on each asset where possible.
- Minecraft servers are now identified on the network.
- HP iLO scans now return additional information.
- Virtual machine hardware is reported if a better fingerprint is not available.
- Pulse Secure VPN devices running newer firmware are now identified correctly.
- Additional CA roots can be set via the RUMBLE_TLS_ADDITIONAL_ROOTCA variable.
- Fingerprint updates.
Self-hosted platform

- Self-hosted installations now sync license changes during updates.
- Email validation is relaxed for the self-hosted platform.
- Install instructions now use curl instead of wget for improved robustness.

Bug fixes

- The last seen date for assets is now only updated when they have at least one open port and are therefore “alive”.
- The estimated scan times in recurring task exports are now accurate when default ports are used.
- A problem with password reset requests for accounts with no last name has been resolved.
- A possible scan stall issue caused by endpoint software on macOS has been resolved.
- A memory and CPU leak in the Rumble Explorer has been resolved.
- User name validation now works correctly when editing user preferences.
- An issue with user invitations for SSO accounts has been resolved.

v2.1.7

2021-05-03

- The web console now includes a new HP iLO CSV export for warranty tracking (Enterprise).
- Virtual machine and legacy MAC address prefixes are no longer used for age calculation.
- Self-hosted installations now sync license changes during updates.
- Virtual machines are now represented in the asset HW field.
- VLANs are now tracked on each asset, where possible.
- Tags are now clickable and lead to inventory searches.
- Fingerprint updates.

v2.1.6

2021-04-28

- Rumble Scans on macOS no longer stall when ICMP scans are blocked by endpoint software.
- The web console now better supports browser widths between 920 and 1200 pixels.
- An issue with user invitations from an SSO account has been resolved.
v2.1.5
2021-04-27
- The Rumble Explorer and Rumble Scanner now collect additional information from HP iLO nodes.
- Fingerprint updates.

v2.1.4
2021-04-26
- A memory and goroutine leak in the the Rumble Explorer and Rumble Scanner has been fixed.
- Fingerprint updates.

v2.1.3
2021-04-23
- The Rumble Explorer and Rumble Scanner now avoid probes on vendor-specific SunRPC services.
- The latest firmware of Pulse Secure VPNs is now fingerprinted correctly again.
- An issue with the confirmation dialogs on Chrome has been resolved.
- The dashboard now defines next steps for new organizations.
- The bundled nmap version has been upgraded to 1.31.
- Fingerprint updates.

v2.1.2
2021-04-19
- The Rumble Explorer and Rumble Scanner now detect the Minecraft service.
- Confirmation dialogs now require a typed response for permanent actions.
- Small bugs have been fixed in the default notification templates.
- Fingerprint updates.

v2.1.1
2021-04-14
- This update disables automatic nmap upgrades while we investigate a stall issue.
**Rumble User Guide**

v2.1.0

2021-04-13

Rumble 2.1.0 is a roll-up of previous 2.0.x releases along with some additional changes and features.

**Web console**

- Custom notification templates are now available and can be configured as Text, HTML, and JSON formats.
- Webhook notification channels can now include arbitrary HTTP headers for authentication.
- Tasks can now be exported as CSV and JSON from the Recurring and Search tabs of the task view.
- Tags be removed in bulk by specifying -tag in the inventory Tag dialog.
- Asset CSV bulk imports now ignore cases where there are extra fields.
- Specific TCP ports can now be excluded from the scan configuration.
- User invitation links can now be copied to the clipboard.
- UX tooltips are now easier to read across all platforms.
- Alert management is no longer organization-specific.
- Exports with complex queries are now much faster.

**Integrations**

- Device serial numbers can now be exported in Cisco Smart Net Total Care format for warranty checks.
- The Splunk add-on now supports proxy server configuration in version 1.0.11.

**Explorer**

- The Explorer now rejects scans tasks when there is not enough free disk space for the scan results.
- The Explorer now falls back to the install directory for temporary files if needed.
- The Explorer now tries to upgrade the npcap driver automatically on Windows.

**Scanner**

- AWS EC2 metadata is used to enrich scan results if the ec2:DescribeInstances permission is available.
- SNMP v2 enumeration of ports and vlan membership now uses community indexing automatically.
Rumble User Guide

- Chromium installations using Snap packages are no longer used for web screenshots.
- An overly aggressive mDNS fingerprint for LG webOS has been fixed.
- EC2 instances now report the instance type as the hardware field.
- Additional bogus SIP helper responses are now ignored.
- LPD fingerprinting is now limited to a status request.
- Fingerprint updates.

Events

- A single event will only trigger a single notification per unique notification channel.
- The agent-removed event is now generated for all web console agent removal actions.
- The offline-agent event no longer includes recently removed or forgotten agents.
- The task-failed event now includes the full organization and site name in the details.
- The task-failed event is now generated for agent restarts and timeouts.

Self-hosted platform

- SMTP configuration now supports additional TLS settings.
- SMTP errors are now logged to syslog correctly.
- Initial auto configuration is more complete.

v2.0.14
2021-04-06

- The Rumble Explorer and Rumble Scanner will now gather AWS EC2 instance metadata where possible.
- Fingerprint updates.

v2.0.13
2021-04-05

- The Rumble Explorer will now fallback to the install directory for temporary files.
- Fingerprint updates.

v2.0.12
2021-04-03
• Exports are now faster across the board, with major speedups for those using deep search queries.
• The Rumble Verifier has been updated and now supports 2.x macOS binaries.
• Various bugs related to the new tooltip implementation have been resolved.
• Chromium will no longer be used for screenshots when installed via Snap.
• Additional bogus SIP helper responses are now automatically ignored.
• Self-hosted installs now log SMTP-related errors.
• Fingerprint updates.

v2.0.11

2021-04-01

• Fingerprint updates.

v2.0.10

2021-03-30

• The Rumble Explorer and Rumble Scanner now collect additional Cisco-specific SNMP OIDs.
• The Cisco Serial Number export is now available for all licensed clients.
• Fingerprint updates.

v2.0.9

2021-03-30

• The Rumble Explorer and Rumble Scanner no longer stall in a rare case when enumerating SNMP endpoints.
• Fingerprint updates.

v2.0.8

2021-03-29

• The Rumble Explorer now automatically cleans temporary files left by interrupted scans.
• The Rumble Explorer now prepends “rumble-” to all temporary files.

v2.0.7

2021-03-28
• The Rumble Explorer out of disk space error has been improved for legibility.

v2.0.6

2021-03-28

• The Rumble Explorer now performs a disk space check before running a new scan.
• The Rumble Explorer now reports disk related errors more reliably.
• Asset CSV imports are now accepted where the records have extra fields.
• Fingerprint updates.

v2.0.5

2021-03-26

• The Rumble Explorer and Rumble Scanner now better support enumeration of Catalyst switches.
• Administrators can copy the invite link for direct sharing with team members.
• Issues with image links on the self-hosted version have been resolved.

v2.0.4

2021-03-26

• The Rumble Explorer and Rumble Scanner now handle the LPD protocol more carefully.
• Fingerprint updates.

v2.0.3

2021-03-23

• The Rumble Explorer no longer reports an intermittent “no child processes” message on installation.
• The Rumble Explorer and Rumble Scanner now always return wireless results when iwlist is present.
• The web console now generates agent-removed events for each agent, including for bulk actions.
• The web console now generates task-failed events for agent restarts and timeouts.
v2.0.2

2021-03-21

- The Rumble Explorer now logs connectivity issues with the websocket protocol in more detail.
- The self-hosted Rumble platform now supports better automatic configuration.
- Fingerprint updates.

v2.0.1

2021-03-20

- A bug that resulted in deleted event rules being processed has been resolved.
- A bug that could lead to scan stalls in Subnet Ping mode has been resolved.
- Fingerprint updates.

v2.0.0

2021-03-16

Rumble 2.0 is a roll-up of the 1.16.x releases, along with the following changes:

Web console

- The new Rules Engine supports advanced alerts and automated asset updates.
- Organization-level Alerts are now managed at the global level.
- The Explore menu item has been renamed to Reports.
- The interface received light cosmetic updates.

Deployment

- Rumble Agents are now Rumble Explorers to better indicate their function.
- Rumble Explorers and Rumble Scanners moved to the Deploy menu.
- Self-hosting of the full platform is now available.

REST API

- The Account API now provides organization, user, and event management.
- Support for ServiceNow® ITOM integration via three new API endpoints.
Scan engine

- Microsoft Exchange and Outlook Web Access detection has improved.
- A stall bug in the subnet and host ping modes has been fixed.
- The number of bogus results ignored is now reported.
- The npcap driver has been upgraded to v1.20.
- Fingerprint updates.

v1.16.6
2021-03-06

- Layer-2 topology reports now display the best matching port and not all ports where an asset was seen.
- The Rumble Agent and Rumble Scanner now handle subnet and host ping modes faster and more accurately.
- The Rumble Scanner now processes gzip-compressed imports faster.
- Fingerprint updates.

v1.16.5
2021-02-27

- Assets that were previously identified with through a TCP RST, but otherwise had no services, have been removed from the platform.
- The Rumble Agent and Rumble Scanner now collect more data about exposed SSH authentication methods.
- A regression related to asset tracking based on the TCP/IP fingerprint has been resolved.
- Fingerprint updates.

v1.16.4
2021-02-26

- The Rumble scan engine now ignores assets where all TCP ports are closed and no other services are available. This reduces the reporting of bogus hosts when scanning through certain firewalls.
- The task summary and task details now report how many assets were ignored due to having no valid services. This highlights how much network interference may be present.
- The macOS binaries have moved to a new code signing process. Agents for macOS that have been offline for some time may require a manual update.
The live asset count and project asset count was calculated incorrectly for users with deleted organizations. This has been corrected.

Search queries that contained string matches against <\% and \%> now work as expected.

Fingerprint updates.

**v1.16.3**

2021-02-25

Fingerprint updates.

**v1.16.2**

2021-02-24

Rumble Agents can now have tags applied in the per-agent settings page.

Events are now regularly generated for offline agents.

Fingerprint updates.

**v1.16.1**

2021-02-21

The Rumble scan engine now supports a maximum TTL for all scan traffic.

The Rumble scan engine now supports subnet ping and host ping modes.

The Rumble scan engine now distributes scan traffic more evenly across subnets.

The Rumble scan engine now reports and tracks closed TCP ports.

The Rumble scan engine now reports additional ICMP fields.

The Rumble scan engine now auto-scales the group size.

Apple macOS is now partially supported on ARM systems.

Fingerprint updates.

**v1.16.0**

2021-02-09

This includes a roll-up of all 1.15.x point releases.

VMware ESXi versions are now reported correctly.

Fingerprint updates.

**v1.15.6**

2021-01-31
• The Inventory Search, Exports, and Reports are now significantly faster for large organizations.
• Fingerprint updates.

v1.15.5

2021-01-28

• The Agents page will now flag any Windows Agents with an obsolete version of Npcap installed.
• Fingerprint updates.

v1.15.4

2021-01-26

• The JARM probe in the Rumble Scanner and Rumble Agent has been upgraded.
• Fingerprint updates.

v1.15.3

2021-01-23

• The Rumble Scanner and Rumble Agent now gather serial numbers from SNMP devices.
• The 169.254.0.0/16 subnet is no longer ignored when processing scan results.
• The Rumble Scanner and Rumble Agent now detect the TeamViewer protocol.
• Partial site scans now consider ARP cache data from the entire site.
• The Rumble Scanner now supports importing gzip-compressed scan data.
• The Rumble Scanner and Rumble Agent now detect the CheckMK service.
• Fingerprint updates.

v1.15.2

2021-01-19

• The dashboard now links to the top 5,000 results for asset types and service details.
• Rumble-provided queries can now be saved as per-account copies and modified.
• Partial site scans will now use ARP cache entries from the rest of the site.
• Fingerprint updates.
v1.15.1

2021-01-16

- This resolves an issue with the Crestron probe that could cause concurrent scans on the same agent to hang.
- Fingerprint updates.

v1.15.0

2021-01-12

- This is a roll-up of all 1.14.x point releases.
- Fingerprint updates.

v1.14.9

2021-01-10

- The Rumble Agent and Rumble Scanner now support the Crestron discovery protocol.
- The Rumble Agent and Rumble Scanner now capture TLS fields from PostgreSQL services.
- Fingerprint updates.

v1.14.8

2021-01-06

- Rumble Agent proxy usage has been improved to handle additional corner cases.
- Fingerprint updates.

v1.14.7

2021-01-05

- Rumble Agent and Rumble Scanner updates now use any proxies configured in the environment.
- Web screenshots now ignore any proxies configured in the environment.
- Fingerprint updates.
v1.14.6
2021-01-04

- A minor memory leak in the Rumble Agent and Rumble Scanner’s HTTP2 probe has been resolved.
- Web screenshots now ignore any proxy configured for the Rumble Agent communication.
- Web screenshots no longer leave zombies in additional environments.
- Fingerprint updates.

v1.14.5
2020-12-28

- A memory leak in the Rumble Agent and Rumble Scanner has been resolved.
- Web screenshots no longer leave zombies in environments without init.
- Fingerprint updates.

v1.14.4
2020-12-24

- Services with empty virtual hosts will be consolidated into the first non-empty virtual host service where applicable.
- Subtasks created by a recurring scan will now carry the “defaults” parameters forward.
- Fingerprint updates.

v1.14.3
2020-12-19

- A memory leak in the Rumble Agent and Rumble Scanner has been resolved.

v1.14.2
2020-12-17

- The Rumble Agent and Rumble Scanner runtime has been upgraded.
- The Rumble Agent and Rumble Scanner now use npcap 1.10.
- The site scan API now handles custom probe configurations.
- The task stop API documentation has been updated.
• Fingerprint updates.

v1.14.1
2020-12-14
• Agents can now be automatically assigned to their connected sites from the Manage menu.
• Scan tasks configured through the API now handle the probes parameter correctly.
• PAN-OS virtual MACs are now ignored for asset correlation.
• Scan task parameters are now consistently normalized.
• Fingerprint updates.

v1.14.0
2020-12-08
• SNMP System Description fingerprints now take precedence over SSH-based OS matches.
• Fingerprint updates.

v1.13.11
2020-12-02
• A bug that caused certain API calls to return a 500 instead of a 400 error when called with a non-JSON content type has been resolved.
• Bogus services caused by captive portals, honeypots, and certain firewalls are now automatically ignored.
• Fingerprint updates.

v1.13.10
2020-12-01
• Agent to cloud communication is now limited to the console.rumble.run hostname and two IPv4 addresses (13.248.161.247, 76.223.34.198).
• The breadcrumbs are now navigable across the product user interface.
• Fingerprint updates.

v1.13.9
2020-11-25
• Inventory search boxes now have autocompletion for search keywords (name:, hardware: and so on). Completion can be triggered via keyboard (tab, enter) or mouse.
• The documentation for search queries has been updated and cleaned up.
• Tag editing dialogs now autocomplete from your top 50 most used tags.
• Some search keywords have been updated to be more consistent.

v1.13.8

2020-11-23
• The Rumble Agent and Rumble Scanner now collect JARM TLS hashes
• The Explore menu now offers additional reports, including JARM.
• Fingerprint updates.

v1.13.7

2020-11-20
• The fingerprinting engine was updated to support more precise device detection.
• Support for new mDNS fingerprints, including M1-based Apple devices.
• Various security fixes to address findings from an ongoing audit.
• Fingerprint updates.

v1.13.6

2020-11-19
• The task details page now shows the scan speed in the upper left section.
• Fingerprint updates.

v1.13.5

2020-11-13
• The Rumble Agent and Scanner have been updated to work with macOS Big Sur.

v1.13.4

2020-11-12
• The Rumble Agent, Scanner, and Console now use the latest Go runtime.
• Fingerprint updates.
v1.13.3
2020-11-10

- Assets with both SMB v1 and v2 enabled are now fingerprinted more consistently.
- The BACnet probe now supports multiple ports per scan.
- Fingerprint updates.

v1.13.2
2020-11-08

- Session and login timeouts can now be configured from the Account Settings page.
- The Subnet Grid report is now faster and supports RTT, TTL, and Age color modes.
- Fingerprint updates.

v1.13.1
2020-11-05

- The new Account Settings page allows MFA to be enforced and provides the ability to block Support access.
- Fingerprint updates.

v1.13.0
2020-11-03

- This is a roll-up of all 1.12.x point releases.
- Fingerprint updates.

v1.12.9
2020-10-25

- The Rumble Scanner and Rumble Agent now decompress non-negotiated gzip responses from HTTP services.
- Fingerprint updates.
v1.12.8
2020-10-25

- The Subnet Utilization report now sorts networks by density more accurately.
- The Subnet Grid report is now slightly faster with large networks.
- Asset correlation now ignores bogus replies for SMB and RDP.
- Fingerprint updates.

v1.12.7
2020-10-24

- The Team page now contains a tab showing which users have access to the active organization.
- Users with per-organization admin roles are now able to manage user accounts.
- Fingerprint updates.

v1.12.6
2020-10-22

- Users with the “No Access” permission are now able to manage their own account settings.
- Users can now have a Billing role that is limited to license and entity management.

v1.12.5
2020-10-21

- Inventory searches with the haspublic keyword now handle multi-homed systems correctly.
- Inventory searches for full and partial IPv4 addresses are now treated as host queries.
- The Subnet Utilization percentage is now calculated correctly for sites with non-default masks.
- Asset tags set from the Inventory page are now additive and will be merged into existing tags.
- Asset tags are now shown on the overview page.
- Small improvements to the Task Search page.
v1.12.4
2020-10-21

- The Rumble Scanner and Rumble Agent now handle an even wider range of SNMP devices when polling the ARP cache.
- Fingerprint updates.

v1.12.3
2020-10-20

- The Rumble Scanner and Rumble Agent now handle a wider range of SNMP devices when polling the ARP cache.
- Fingerprint updates.

v1.12.2
2020-10-17

- The Rumble Scanner terminal UI has been cleaned up and generally improved.
- Fingerprint updates.

v1.12.1
2020-10-15

- The Rumble Scanner and Rumble Agent now deduplicate overlapping target network ranges.
- Fingerprint updates.

v1.12.0
2020-10-13

- This is a roll-up of all 1.11.x point releases.

v1.11.9
2020-10-11

- The Rumble Scanner can now automatically update when run with the upgrade argument.
The Rumble Agent can be installed via a static MSI wrapper.

**v1.11.8**

2020-10-04

- The scan engine is now more consistent through the addition of UDP retries and by pre-warming the ARP cache for each target group.
- Fingerprint updates.

**v1.11.7**

2020-09-28

- The Rumble Agent and Rumble Scanner now include npcap version 1.0.
- Fingerprint updates.

**v1.11.6**

2020-09-23

- Users with the the Viewer role are now restricted to the Dashboard, Inventory, Explore, and Agents screens.
- Fingerprint updates.

**v1.11.5**

2020-09-21

- A regression in the license expiration tracking of the Rumble Scanner has been resolved.
- Fingerprint updates.

**v1.11.4**

2020-09-18

- Fingerprint updates.

**v1.11.3**

2020-09-13


• Fingerprint updates.

**v1.11.2**

2020-09-11

• Organizations can now be converted to Projects from the settings page.  
  • The top-level organization navigation has been updated.

**v1.11.1**

2020-09-09

• Accounts with a default organization role of “user”, but a per-organization role of “none” were inadvertently prevented from accessing certain features.  
  • An intermittent crash in the Rumble Agent was identified and resolved.

**v1.11.0**

2020-09-09

• This is a roll-up of all 1.10.x point releases.

**v1.10.8**

2020-09-08

• Projects are now available as temporary, self-deleting organizations. These are useful for one-off scans and exploring historical data.  
  • Web screenshots now try additional Chrome locations on the Windows platform.  
  • Over 10,000 new SNMP fingerprints have been added.  
  • Small bugfixes and cosmetic improvements.  
  • Additional fingerprint updates.

**v1.10.7**

2020-09-01

• Web screenshots now retry on timeouts and choose the best quality image automatically.  
  • Web screenshots now use more concurrent Chrome processes on x86 systems, based on available memory.
The Rumble Agent and Rumble Scanner now track CPU and memory usage across the life of a scan.
Fingerprint updates.

v1.10.6
2020-08-22

- The Rumble Agent, Rumble Scanner, and Rumble Console now compress raw scan by default. The scan.rumble output from the scanner has been renamed to scan.rumble.gz. The web console and API can import both compressed and uncompressed versions of this data. Existing scan data will be migrated to the compressed form automatically. This change helps with bandwidth usage by agents and speeds up large imports over the network.
- The Tasks view now links to the inventory search for each associated site.
- The status of agent-run scans is now updated more frequently.

v1.10.5
2020-08-21

- The completed task list now shows the task runtime in the information column.
- The task views now also link to the inventory view of each site.
- Fingerprint updates.

v1.10.4
2020-08-17

- A regression in the Rumble Scanner that prevented API uploads from succeeding has been resolved.
- Fingerprint updates.

v1.10.3
2020-08-14

- Small bug fixes and dependency updates across the platform.
- Fingerprint updates.

v1.10.2
2020-08-12
• Site exports and imports now include the registered subnets.
• Bulk asset updates are now possible by importing a modified CSV export from the Inventory screen.

v1.10.1

2020-08-11

• A bug that made it difficult to query subnet tags with multiple subnets per tag has been resolved.
• Fingerprint updates.

v1.10.0

2020-08-04

• The console user interface received a light update around colors and styles.
• Event logs are now available in the console.
• Fingerprint updates.

v1.9.10

2020-08-03

• A bug that prevented Scan Tags from being shown in the Scan Configuration form has been resolved.
• The DNS and mDNS probes now always report the protocol, even for error responses.
• Fingerprint updates.

v1.9.9

2020-08-02

• Site scopes now automatically convert CIDR input into registered subnets.
• Scan tasks can now have their Scope and Excludes pinned to their associated site using the string "defaults".
• Scan tasks can now be pinned to the default TCP service list using the string "defaults".
• A bug that caused non-Windows SMB-enabled services to be identified as Windows has been resolved.
• A bug that caused SMB v1 to be reported incorrectly as been resolved.
• Fingerprint updates.
v1.9.8

2020-07-29

- A bug that prevented the Delete and Merge buttons in the Service Inventory toolbar from working has been resolved.
- A bug that led to the wrong title being shown in the FTP Service Attribute report has been resolved.
- Fingerprint updates.

v1.9.7

2020-07-22

- A bug that prevented the Inventory Import action from recognizing valid scan data has been resolved.
- The Rumble Agent and Rumble Scanner are now much more reliable for lossy network environments.

v1.9.6

2020-07-21

- The TCP probes have been updated to be less bursty. This resolves an issue where scans consisting of mostly HTTP services can timeout and lose valid responses.
- The TCP fingerprinter now handles unexpected termination more gracefully. This improves the reliability of AWS ELB scans and should help with a reliability across a range of services.

v1.9.5

2020-07-20

- All paid plans now support Continuous recurring scans. These scans will run back-to-back and can simplify continuous monitoring. An agent running continuous scans will not run additional scans unless the Concurrency setting is increased beyond 1.
- Out-of-date agents will be upgraded prior to new scans being run. For the few agents where upgrades are impossible (read-only partitions, network filters, etc), this can delay each scheduled scan by up to five minutes.
v1.9.4

2020-07-18

- The Rumble Agent and Rumble Scanner now include a TLS CA root bundle to work around connectivity issues on older platforms. Bundle selection can be controlled via the environment
- Fingerprint updates.

v1.9.3

2020-07-16

- The web screenshot features now tries even harder to prevent orphaned Chrome.exe processes.
- The Rumble Agent now removes all agent-related files on uninstall.
- Rumble Agents can now be reassigned to other organizations.

v1.9.2

2020-07-12

- The Export API now supports an optional fields parameter that determines which fields are exported in JSON/JSONL exports. The fields parameter is supported for Assets, Services, Wireless, and Sites.

v1.9.1

2020-07-09

- A bug in the scan engine that could cause scans to hang when probing unresponsive SSH daemons has been resolved.
- A bug in the scan engine that could result in SMBv1 being reported erroneously on some NAS devices has been resolved.

v1.9.0

2020-07-06

- A bug in the TFTP probe that could lead to missing results in some cases has been fixed.
- The SNMP probe now gathers the route table from many types of switches and routers.
• TCP SYN scans of non-local targets now try harder when there is congestion.
• Fingerprint matches that include a hardware version are now given priority.
• Fingerprint matches for SSH daemons now support more platforms.
• The permanent organization and permanent site can now be deleted/recreated.
• The Scan Configuration page now shows a notice when input validation failed.
• The Scan Configuration now shows SNMP parameters at the top of the form.
• The Network Bridges report now links all external IPs to an internet cloud.
• The Network Bridges report now uses subnet masks from Sites.
• The Subnet Utilization report now provides a Scan link for each network.
• The Subnet Utilization report now uses subnet masks from Sites.
• The Subnet Grid report now handles errors more gracefully.
• A bug that prevented some users from logging in has been resolved.
• Search queries are now slightly faster across assets and services.

v1.8.14

2020-07-02

• Tasks are now searchable and sortable via the Search tab.
• A regression in numerical search queries has been resolved.

v1.8.13

2020-07-01

• The Scan Configuration page now provides an estimated runtime through a confirmation dialog.
• Trial accounts are now longer limited to scanning a /16 and may now scan a full /8.
• The Rumble Agent now supports log configuration using the environment. See the documentation for details.
• The Rumble Agent and Rumble Scanner now collect SSH pre-auth banners and host keys.
• Bogus service responses from Fortigate helpers on ports 80 and 8008 are now ignored.
• Fingerprint updates.

v1.8.12

2020-06-24

• The Rumble Agent and Rumble Scanner now handle a wider range of ppp-based link types on Linux and macOS.
• Bogus service responses from Fortigate helpers on ports 21, 25, 80, 110, 143, 8008, 8010, and 8020 are now ignored.
- Fingerprint updates.

**v1.8.11**

2020-06-22

- Bogus service responses from Cisco H.323 helpers on port 1720 are now ignored.
- The Rumble Agent now stores additional diagnostics in the raw task data.
- Fingerprint updates.

**v1.8.10**

2020-06-21

- Bogus service responses from Fortigate SIP ALG helpers on ports 2000 and 5060 are now ignored.
- A regression in HTTP handling with redirects and TLS+HTTP headers has been resolved.
- Fingerprint updates.

**v1.8.9**

2020-06-20

- The Rumble Agent and Rumble Scanner now handle malformed HTTP responses and redirects better.
- Fingerprint updates.

**v1.8.8**

2020-06-18

- ICMP Echo probes now record the IP header information from the response (useful for Ripple20/Treck detection).
- The Rumble favicon.ico MD5 fingerprint database has been contributed to the Recog project.

**v1.8.7**

2020-06-16

- The Rumble Agent and Rumble Scanner now support “cooked” interface types (ppp-based VPNs).
• The scan engine now extracts additional information from Netgear routers.
• Fingerprint coverage for Netgear routers has been improved.

v1.8.6

2020-06-15

• Fingerprint updates.

v1.8.5

2020-06-15

• Fingerprint updates.

v1.8.4

2020-06-15

• Fingerprint updates.

v1.8.3

2020-06-14

• Asset Inventory and Search Inventory performance has been improved.
• The bundled npcap driver in the Rumble Agent and Rumble Scanner for Windows has been upgraded to version 0.9994.
• Fingerprint updates.

v1.8.2

2020-06-09

• The Rumble Scanner CSV output now includes populated UUID values.
• The Rumble Scanner now creates a standalone bridges.json file for third-party processing.
• Fingerprint updates.

v1.8.1

2020-06-09
• A bug that could cause agent uninstalls to crash on BSD platforms has been resolved.

v1.8.0

2020-06-09

• This release is a roll-up of the 1.7.x changes listed below.

v1.7.13

2020-06-08

• Sites now support registered subnets. Assets can be queried via the associated Site subnet tags.
• Tags can be set with empty values and queried more precisely through the Inventory search.
• Asset fingerprinting via favicon.ico hashes has been implemented.
• The Rumble Scanner now creates a standalone topology.json file for third-party processing.
• Assets now store the MAC-to-IP relationship in the hidden _macs.ipmap attribute.
• The Rumble Agent and Rumble Scanner now support OpenBSD on x86 (64-bit).
• Fingerprint updates.

v1.7.12

2020-06-05

• This release fixes a bug that prevents the Rumble Agent from restarting automatically after an update on certain Debian-based distributions.
• Fingerprint updates.

v1.7.11

2020-06-04

• A reliability bug in the Rumble Agent and Rumble Scanner for BSD-based platforms (macOS, FreeBSD, NetBSD, DragonFly BSD) has been resolved. This bug would manifest as missing scan results in the TCP SYN and ARP probe responses.
• The bundled npcap driver in the Rumble Agent and Rumble Scanner for Windows has been upgraded to version 0.9992.
v1.7.10

2020-06-02

- The Rumble Agent and Rumble Scanner now support FreeBSD, NetBSD, and DragonFly BSD. FreeBSD and NetBSD support cover the following architectures: x86 (64-bit, 32-bit), ARM v5, ARM v6, and ARM v7. DragonFly BSD is supported on x86 (64-bit).
- The Rumble Agent and Rumble Scanner now support additional Linux architectures. These include x86 (64-bit, 32-bit), ARM v5, ARM v6, ARM v7, ARM 64-bit (aarch64), MIPS (BE/LE), MIPS64 (BE/LE), PowerPC64 (LE), and s390x (IBM Z).
- The Rumble Agent now runs in standalone mode when no supported services backend is detected.
- The Rumble Agent now supports automatic updates in standalone mode on non-Windows platforms.
- The Rumble Agent binary now supports command-line flags (-h, -v, -l) and displays usage.

v1.7.9

2020-05-27

- MAC address fingerprints are now live. The initial set includes fingerprints for devices manufactured by Amazon, Google, Honeywell, August, SimpliSafe, TRENDNet, FLIR, Microsoft, Belkin, Meross, LG, Logitech, Hunter, Lutron, Orbit, Arlo, Panasonic, Sony, Vizio, Chameleon, iRobot, SharkNinja, Netatmo, Nintendo, HP, Intel, Lenovo, Dell, and PC Engines. MAC fingerprints are used as a fallback when more precise fingerprinting is not available.
- Microsoft SQL Server versions obtained from the network are now mapped to specific releases and patch levels, enabling queries that look for end-of-life versions and missing patches.
- Chromecast devices now return additional service attributes, including information about the wireless network that they are connected to. Fingerprinting of older Chromecast models (Gen 1) has been improved. MAC addresses and additional IP addresses from the Chromecast web endpoint is now applied to the asset.
- MySQL and MariaDB version detection now also applies the appropriate OS fingerprint, if known.
- HTTP services that return JSON responses now camelCase the attribute names and support a wider range of data types. This impacts JSON-based HTTP interfaces such as ElasticSearch and Riak HTTP.
- OS and Hardware matching is more precise after adjustments were made to the weighting and priorities. The most precise and most confident fingerprint should
always be chosen going forward.

- The confidence of the OS match is now reported as the asset-level match.score attribute. This may be renamed to match.os.score in the future as we accommodate more granular hardware weights.
- NTLMSSP-based OS matching now disqualifies systems that are obviously not Windows (BSD-based stacks, etc).
- Brother printers now use distinct hardware and firmware (OS) fingerprints. This should address cases where the firmware version overrode the hardware model by mistake.
- Release notes are now consolidated across the Platform, Agent, and Scanner.
- Versioning is now shared across the Platform, Agent, and Scanner.

**Archived release notes**

Prior to version 1.7.9, release notes and versions were split between the Platform, Agent, and Scanner. You can find these archived release notes at the links below.

- Archived Release Notes: Rumble Platform
- Archived Release Notes: Rumble Agent
- Archived Release Notes: Rumble Scanner
Older platform release notes

Platform release notes

Starting with version 1.7.9 all release notes have been consolidated into one page.

v1.7.8
2020-05-23

- The pre-login style has been updated.
- The account registration flow has been updated to provide a smoother activation experience.

v1.7.7
2020-05-17

- Restricted user accounts may now be assigned roles within multiple organizations.

v1.7.6
2020-05-14

- Asset and Service attributes have been normalized. All keys are now camelCase and most service attributes are now prefixed by the protocol name.

v1.7.5
2020-05-13

- SMB v2 OS detection now applies more weight to Windows 10 versus Server 2019 in most situations.
- SMB v1 OS detection has been improved for non-Windows platforms.

v1.7.4
2020-05-08

- SSO now supports a default role of No Access. Accounts created through the SSO login process with this default role must be manually promoted to a usable...
Users now support a No Access role for both All Orgs and specific organizations.

A bug preventing access to the per-account Demo Organization has been resolved. Agents and Scanner downloads for the Demo Organization have been explicitly disabled. Any agents with the Demo Organization will automatically uninstall.

All Rumble Console confirmation dialogs now list the specific item they are confirming. This can prevent misclicks from removing the wrong item.

A fingerprinting regression related to non-Windows machines running SMB services has been resolved. This would unintentionally classify some NAS devices as Microsoft Windows.

v1.7.3

2020-05-08

- ToneLoc export support in the Subnet Grid Report.

v1.7.2

2020-05-07

- Cisco HSRP MAC addresses are now ignored for the purposes of asset correlation.

v1.7.1

2020-05-06

- Agent diagnostic information is now visible from the agent details page. This requires agent version 1.7.1 or later.

v1.7.0

2020-05-04

- The Subnet Grid View report is now available (via the Subnet Report).

- The ts, tls.notBeforeTS, and tls.notAfterTS service fields can now be queried using time comparison operators.
v1.6.10
2020-05-03

- Scan schedules now show the day of the week for first, last, and next timestamps.
- A bug that obscured the validation error for invalid TCP ports has been resolved.
- Alert Rules can now be limited to specific Sites or All Sites.
- Dependency upgrade across the backend and frontend.

v1.6.7
2020-04-26

- A bug in the recurring scan limits of the Starter Edition has been resolved.

v1.6.6
2020-04-23

- A bug in the asset export that resulted in missing service details has been resolved.

v1.6.5
2020-04-22

- Rumble now supports a free tier via the Starter Edition.

v1.6.4
2020-04-19

- The dashboard now shows statistics for live assets only and not all assets.
- The license view now ignores demo organizations and sites in the statistics.
- The task view now represents scans pending processing more accurately.

v1.6.3
2020-04-18

- The Asset and Service exports now include the service.address, service.transport, service.port, and service.vhost fields (if present) to make them more consistent with the Rumble Scanner assets.jsonl exports.
v1.6.2

2020-04-12

- The Analysis Reports section has been added, including the new Domain Membership and Service Attribute reports.
- The Query Library has been added, including a set of predefined common queries.
- Inventory search queries can now be saved to the Query Library.
- Inventory search action buttons have been slimmed down.
- The Explore tab provides easy access to the Analysis Reports and Query Library.

v1.6.1

2020-04-07

- Accounts that have been temporarily locked due to failed logins may be manually unlocked by superusers from the Team page.

v1.6.0

2020-04-06

- Data retention controls are now available on the Organization Settings page. These can be used to specify data retention periods and remove stale offline assets after a fixed period of time.

- Inventory, Site, and Organization search now support exact matches with the = prefix. An exact hostname match can be specified as name:=SERVER01.

- Inventory, Site, and Organization search now support matching empty fields using the = prefix. An empty hostname match can be specified as name:=

- Inventory searches now support the mac-vendor_count, macvendor_count, and vendor_count terms.

- Inventory Asset and Service search now support the haspublic, hasprivate, hasipv6, and haslinklocal terms.

- Inventory Asset and Service search now support % as a wildcard for address matching. The query address:%.0.1 will find all assets with an address ending in .0.1.
A bug that prevented searches for less than one hostname has been fixed (name_count:0 and name_count:<1).

- Active and Failed scan tasks now support the Copy action. Copied scans scheduled in the past will default to the current time to start.
- Single Sign On users can now specify the default organization role of SSO-enrolled users.
- Account lockouts are now enforced after 10 failed password-based login attempts.

**v1.5.6**

2020-04-01

- The Screenshot Inventory now sorts by address instead of port number by default.

**v1.5.5**

2020-03-27

- The Asset Inventory now allows attributes to be queried even if they conflict with a keyword by using the prefix _asset.
- The Service Inventory now allows attributes to be queried even if they conflict with a keyword by using the prefix _service.

**v1.5.4**

2020-03-24

- Fingerprinting improvements for HTTP, X509, Telnet, SSH, FTP, and SIP.

**v1.5.3**

2020-03-22

- Scan tasks now have a configurable grace period. Setting this to zero or a negative value will result in the scheduling retrying indefinitely until an agent becomes available.

**v1.5.2**

2020-03-15
- Web services with multiple asset icons now show each icon correctly.
- A bug that prevented Sites from being imported has been resolved.
- A bug that prevented superusers from logging in directly when SSO was in required mode has been resolved.

**v1.5.1**

2020-03-09

- Asset icons are now lazy loaded as they scroll into the viewport.
- Asset icons URLs are now more cache friendly.
- Page meta tags have received small tweaks.

**v1.5.0**

2020-03-04

- This release includes major updates to the Rumble Agent and Rumble Scanner.
- The asset inventory now displays any HTTP and UPnP icons acquired during the scan.
- The asset inventory now displays an icon for assets with available screenshots.
- The inventory search no longer triggers an error on some grouped query corner cases.
- The inventory search now supports /32 masks for the net and cidr keywords.
- The inventory search now treats the type keyword as an exact match.
- The inventory search now supports filtering based on service counts.
- The inventory search now supports filtering based on the RTT and TTL fields.
- The inventory search for services now supports filtering based on port ranges.
- The asset detail page now describes upstream and downstream layer 2 links.
- The asset detail page now lists all acquired icons in the top summary.
- The asset detail page now has a Scan action available from the menu.
- The new scan page now supports applying a list of tags to all identified assets.
- The new scan page agent selection now defaults to the first site-specific agent.
- The tasks page now allows recurring scans to be paused an unpaused.
- The console now always shows the live agent count on the navigation menu.
- The console now applies styling to the print view.
- Hostnames consisting of leading digits are now ignored when correlating assets.
- A bug that resulted in the wrong timezone being used after SSO login has been resolved.
- Scans triggered via API now default to the site scope if no targets were specified.
- Scans triggered via API can now specify a list of tags to apply to all identified assets.
- Import tasks with the wrong data supplied format are rejected before task creation.
v1.4.5

2020-02-19

- Completed tasks are now always shown in the order they finished (versus were created).
- A Cisco IOS fallback fingerprint was leading to false positives and has been disabled.

v1.4.4

2020-02-15

- The License page now lists invoices and allows payment methods to be updated.
- The SMB Server GUID attribute is now used to correlate results to assets.
- The SNMP sysName and sysObjectID attributes are now used to unmatch assets that have changed IPs or were mistakenly matched through another attribute (shared bogus MAC addresses or similar).

v1.4.3

2020-02-13

- A bug that could treat active agents as offline has been resolved.
- Windows agents are now limited to a single concurrent scan to avoid resource contention issues.

v1.4.2

2020-02-09

- Changes in source TTLs on external segments no longer disqualify an asset match during correlation.
- Cosmetic updates to email templates and task displays.

v1.4.1

2020-02-05

- Updates to various javascript and backend dependencies.
**v1.4.0**

2020-02-04

- Version 1.4.0 is a rollup of post-1.3.0 point release work.

**v1.3.5**

2020-02-02

- Scans now support specifying the SNMP v3 Context for devices that require this (CatOS).
- Scans now support the “Fast” option for the ARP probe, which is on by default in AWS VPC environments.
- Agents now support concurrent scans, with the limit configurable per-agent.
- Asset correlation avoids accidentally grouping when a forged MAC address is present in routed protocol responses (NetBIOS, SNMP, etc).
- Change reports now ignore differences in reverse DNS, hostnames, domain names, and the top-level service count.
- Recurring, scheduled, and queued tasks without an available agent will mark the task as failed after 4 hours. Recurring tasks will attempt to run scans normally during the next scheduled period.
- Scanner binaries for OEM customers now behave according to the license agreement.
- Long task names are now truncated automatically in the task list.

**v1.3.4**

2020-01-29

- The failed tasks tab now shows a longer history by default.

**v1.3.3**

2020-01-29

- Minor dependency upgrades (javascript and backend).

**v1.3.2**

2020-01-28
Search queries can now use parenthesis to group terms, for example: os:linux AND (port:22 OR port:80) AND NOT port:3306.

v1.3.1
2020-01-15
- The Export API now supports an asset sync backend with support for checkpointing based on created_at or updated_at fields.

v1.3.0
2020-01-07
- Version 1.3.0 is a rollup of post-1.2.0 point release work.

v1.2.9
2020-01-04
- The Organization API is now available.

v1.2.8
2019-12-29
- The Teams view is now split between members (access to all organizations) and restricted users (access to single organizations).
- Team member invitations now have a Reply-To header set to the inviting user.
- Team member invitations can now specify custom subject lines and messages.
- Active tasks are now always sorted by their start time, not their last update time.
- Tasks now always show the created by field (for tasks created or updated after v1.2.2).
- The search warnings field is now cleared after each new query.
- The search timestamp fields now default to less-than (<) if no operator is specified.

v1.2.7
2019-12-28
- The Active Tasks tab now offers a Clear Queue button to remove all queued tasks.
v1.2.6

2019-12-27

- Sites and Organizations can now be sorted and searched.

v1.2.5

2019-12-26

- The Beta discount is officially retired.
- End of year discount is valid through 2019

v1.2.4

2019-12-19

- Support for BACnet is now available in new scans.
- Leading and trailing spaces in Site and Organization names are now removed.

v1.2.3

2019-12-13

- Many new fingerprints were added for HTTP and SIP endpoints. These may trigger asset change notifications due to improved device and service recognition.
- Network scans of segments where a device responds to all ARP requests with the same MAC address will now be handled appropriately.
- The SIP protocol is now reported properly on UDP SIP responses.

v1.2.2

2019-12-10

- Tasks now track which user created or last updated them.

v1.2.1

2019-12-05

- SSO configurations with multiple IdP x509 certificates will now validate against any certificate in the list, not just the first one.
v1.2.0
2019-12-01
- Version 1.2.0 is a rollup of post-1.1.0 point release work.

v1.1.9
2019-12-01
- Initial support for the Wireless Inventory using the new wlan-list probe.

v1.1.8
2019-11-27
- An encoding issue in the Inventory display of certain columns has been resolved.

v1.1.7
2019-11-24
- The Scan Max Group Size is now set to a reasonable default when older scans are copied.
- The Copy action will no longer create names with more than one “Copy of” prefix by default.

v1.1.6
2019-11-19
- SNMP v3 credentials may now be specified in the scan configuration.

v1.1.5
2019-11-19
- The Switch Topology report is now available from the Inventory reports menu.
- Assets that respond to SNMP enumeration of the CAM/MAC table are now automatically classified as switches.
**v1.1.4**

2019-11-18

- A bug in the email notifier has been resolved that prevented delivery in rare circumstances.

**v1.1.3**

2019-11-14

- The Network Bridges report has been optimized for improved readability.

**v1.1.2**

2019-11-11

- The agent inventory search term has been added. This accepts a UUID as well as the agent name.

**v1.1.1**

2019-11-08

- Email notifications for expiring, deactivating, and unused trial accounts.

**v1.1.0**

2019-11-05

- Version 1.1.0 is a rollup of post-1.0.0 point release work.

**v1.0.20**

2019-11-04

- OS detection now prefers the most granular fingerprint in the case of multiple matches.
- Stopped tasks now show in the Failed list and not the Completed list.
- Scans now support the Max Group Size option to limit the number of concurrent scan targets.
- The site inventory search term now accepts a UUID as well as the site name.
The task inventory search term now shows all assets that were last updated by a given task ID.
- The lowest_ttl field is no longer considered as a significant change in asset change notifications.
- A bug that led to some assets being stored with only IPv6 link-local addresses has been resolved.
- A bug that prevented OS version numbers from being tracked properly has been resolved.
- A bug that allowed hostname-based OS matching to override more reliable fingerprints has been resolved.

v1.0.19

2019-11-01

- Trial accounts and associated data are now automatically removed two weeks after the license period expires.

v1.0.18

2019-10-31

- The Network Bridges report now supports filtering using the same syntax as the Asset Inventory search.
- The Network Bridges report is now much faster, with the caveat that nodes are no longer draggable.
- The Network Bridges report now supports up to 1,000 multi-homed nodes at a time.
- The Completed Tasks view no longer shows tasks in a Cancelled or Error state (these are now listed under Failed Tasks).
- The Asset Inventory view now allows merging multiple assets into a single asset.

v1.0.17

2019-10-26

- Failed tasks can now be cleared via the Clear action button.

v1.0.16

2019-10-25

- The Export API now treats the Organization ID in the URL as optional.
• The Asset and Services inventory now support the `id` search term to find assets by ID.

**v1.0.15**

2019-10-25

• The Export API now offers a list of sites in three formats (CSV, JSON, JSONL).
• The change summary titles in the task view now link to their respective sections in the body.

**v1.0.14**

2019-10-24

• A small number of typos and cosmetic improvements resolved with the web console.

**v1.0.13**

2019-10-23

• The SYN probe now retries twice if no RST is received. This improves reliability at the cost of a small increase in scan times. This can be changed by the `syn-max-retries` advanced parameter for the SYN probe in the new scan configuration screen.

**v1.0.8**

2019-10-18

• Single Sign On support is now available for all users (SAML2). This is available the SSO Settings button on the My Team page.

**v1.0.7**

2019-10-07

• A handful of small cosmetic issues have been resolved in the web interface.

**v1.0.6**

2019-10-08
• The asset correlation algorithm now accepts looser matches for TCP/IP stack fingerprints when matching an asset by IP address. This reduces the chance of asset churn when one or more services change between scan runs.

v1.0.5

2019-10-07

• Scan tasks now have Name and Description fields. These can be set during task creation and updated in Recurring and Scheduled tasks. These fields are shown in the main task view right after the task type.
• Existing scan tasks can now be copied to a new scan configuration (Copy action in the task list). This simplifies the process of running a one-off scan from an existing recurring scan definition.
• Less common scan options have been moved to the Advanced Scan Options section, which is now visible by default in the scan configuration page.
• The Team Manage menu now include an option to reset the security tokens of a team member (available to superusers only).
• The Sites listing now links the site name to the inventory search query. To edit the Site definition, use the Update Site action from the Modify menu.

v1.0.3

2019-10-06

• A race condition was fixed that could lead to some scan tasks failing with the error agent failed to queue task.
• Minor cosmetic improvements were applied to the console.

v1.0.2

2019-10-03

• A bug was fixed that caused some Scanner downloads to have a license expiration set in the past.
• The task scheduler is now more tolerant of temporary network errors when queueing scans.

v1.0.0

2019-10-01

• Rumble Network Discovery is out of Beta with version 1.0.0!
Older explorer release notes

Explorer release notes (agents)

Starting with version 1.7.9 all release notes have been consolidated into one page.

v1.7.8
2020-05-23
  • Fingerprint updates.

v1.7.7
2020-05-22
  • Fingerprint updates.

v1.7.6
2020-05-14
  • Corrects inconsistent use of the new service attributes when processing the dynamic MAC address filter.

v1.7.5
2020-05-14
  • Asset and Service attributes have been normalized. All keys are now camelCase and most service attributes are now prefixed by the protocol name.

v1.7.4
2020-05-13
  • Support has been improved for the following database protocols: Memcached (TCP), CouchDB, Cassandra, Redis, ElasticSearch, Riak (TCP/HTTP), MySQL, PostgreSQL, MongoDB, MSSQL, and Oracle.
v1.7.3
2020-05-07

- Cisco HSRP MAC addresses are now ignored for the purposes of asset correlation.
- Updated Ethernet fingerprints.

v1.7.2
2020-05-06

- A bug in the mDNS probe that could lead to a hung scan on certain platforms has been resolved.

v1.7.1
2020-05-06

- Diagnostic information is now returned in the “systeminfo” response.
- Updated Ethernet fingerprints.

v1.7.0
2020-05-04

- The bundled npcap driver has been upgraded to version 0.9991.
- The TLS probe now reports tls.notBeforeTS and tls.notAfterTS fields as unix timestamps.
- Updated Ethernet fingerprints.

v1.6.10
2020-05-03

- Updated Ethernet fingerprints.

v1.6.9
2020-05-01
• The Rumble Agent will verify its own binary and exit on startup if corrupted.
• Updated Ethernet fingerprints.

v1.6.8
2020-04-23
• Updated Ethernet fingerprints.

v1.6.7
2020-04-23
• The scan engine can now identify TCP services on the scanning system across all platforms.

v1.6.6
2020-04-22
• An issue that could lead to the scan engine hanging with misbehaving HTTP services has been resolved.

v1.6.5
2020-04-22
• Updated Ethernet and BACnet fingerprints.

v1.6.4
2020-04-17
• Devices that relay mDNS from other networks (ex: Ubiquiti USG) are no longer associated with the relayed asset information.

• Additional Google Chrome paths are considered for screenshot collection. Snap packages of Chromium are no longer used.

v1.6.3
2020-04-14
• An issue that could lead to scans hanging while processing HTTP services has been resolved.

**v1.6.2**

2020-04-13

• The Linux agent now restarts properly on Ubuntu 14.04, CentOS 6.10, and other Upstart-based platforms.
• Fingerprint updates for Crestron, ELAN, MAC addresses, and BACnet.

**v1.6.1**

2020-04-08

• The Windows agent will now try harder to work around temporary issues while installing an update.
• The Windows agent will now clear any stale chrome.exe processes running as LocalSystem during the update process.

**v1.6.0**

2020-04-06

• Screenshots will now limit the number of concurrent Chrome processes based on core count, available RAM, and architecture.
• The bundled npcap build has been updated to version 0.9990.

**v1.5.6**

2020-04-04

• The RDP probe now collects the full NTLMSSP response for more platforms.
• The HTTP probe now collects information about web forms and their inputs.

**v1.5.5**

2020-03-27

• The SNMP probe no longer reports invalid MAC addresses found in ARP caches or MAC tables.
v1.5.4

2020-03-26

- The TCP probe now handles a wider variety of RDP responses.

v1.5.3

2020-03-26

- The SMB probe now reports subprotocols (smb1, smb2, and smb3) consistently.
- The SMB probe now collects hashing, encryption, and compression methods from SMBv3 servers.
- The SMB probe now reports the server-allocated Session ID for smb2 and smb3.
- The TCP probe now collects NTLM information from Remote Desktop endpoints and reports the protocol as rdp.
- The HTTP probe now collects additional information from VMware SOAP endpoints.

v1.5.2

2020-03-20

- A bug that could lead to the HTTP/2 probe stalling during TLS negotiation has been resolved.

v1.5.1

2020-03-14

- A full system disk no longer results in a deadlocked scan.
- Fingerprints have been updated for Ethernet MAC addresses, BACnet vendors, and Enterprise IDs.
- HTTP/1 probes now explicitly disable HTTP/2 upgrades even when advertised. HTTP/2 is handled separately.
- Generic protocol negotiation is no longer attempted on NDMP ports (10,000/30,000). A future release will support improved NDMP detection and negotiation.
- A potential deadlock in the runtime library has been resolved by reverting to an older runtime version.
v1.5.0

2020-03-04

- A NTP probe has been added that reports the clock skew compared to the scanning instance.
- A TFTP discovery probe has been added that requests a non-existent file and stores the response. The TFTP probe supports port ranges.
- An OpenVPN probe has been added that can detect remote instances across multiple ports.
- A dTLS discovery probe has been added that handles both bare dTLS and CAPWAP-encoded variants.
- Microsoft Remote Desktop Gateway instances are now fingerprinted through dTLS and HTTP, reporting the rdg.Transport service key.
- The protocol handlers for NATPMP, WS-Discovery, and UPnP Device XML now parse out specific subfields for easier matches and future fingerprinting efforts.
- The UPnP Device XML parser now triggers a request to download and report the device icon.
- The SYN scanner has been updated to improve reliability and report more accurate progress.
- The HTTP probe now identifies and reports web site icons as base64-encoded images along with their MD5 hashes.
- The HTTP probe now extracts the generator meta tag from HTML responses.
- The HTTP probe now extracts splunkd versions from HTML responses.
- The RPCBind probe now sends a null call to every UDP service and probes the NFS daemon directly.
- VMware ESXi detection has improved and will be used as a fallback in more cases.
- TCP protocol fingerprinting will retry more often on temporary network errors.
- Empty fields in the result structure within the JSON output are now omitted.
- Linux on ARM 64-bit (aarch64) is now a supported platform.
- Improved detection and early rejection of invalid CIDRs.

v1.4.5

2020-02-19

- The SMB probe now records the NTLMSSP response from a wider range of operating systems.
- The HTTP probe now stores the response to GET / and the response after any redirects are followed. Key names for the redirect responses are prefixed by last, such as last.http.code.
- The HTTP probe now handles compression and chunked transfer encoding properly, storing the normalized HTTP body.
• The HTTP probe now reports a banner consisting of the raw HTTP response.
• The HTTP probe now supports collecting environment data from LANDesk Management Agents.
• HTTP screenshots are now only collected when a 2XX HTTP response code is seen.
• HTTP screenshot processing is now more reliable.

v1.4.4

2020-02-16

• The SMB Server GUID attribute is now used to correlate results to assets.
• The SNMP sysName and sysObjectID attributes are now used to unmatch assets that have changed IPs or were mistakenly matched through another attribute (shared bogus MAC addresses or similar).
• Interfaces with no global unicast addresses (including RFC1918) are no longer considered by the ARP and SYN scanners.
• VLAN-tagged frames are now ignored by the SYN scanner resolving an issue where packets could be sent on the wrong interface by mistake.
• SYN scans now have a mandatory delay between retry attempts, which improves reliability and decreases change churn when small network ranges are scanned.

v1.4.3

2020-02-13

• Concurrent scans now use less resources and provide more accurate results.
• A bug that caused some HTTP requests to be sent with an empty Host header has been fixed.
• Version 1.4.1 and 1.4.2 were internal test releases and not deployed.

v1.4.0

2020-02-04

• Version 1.4.0 is a rollup of post-1.3.0 point release work.

v1.3.2

2020-02-02

• Agents now support concurrent scans, configured via the cloud console.
• Agents now write log files and rotate these log files automatically.
• Agents now scan faster on local segments within AWS VPCs.
- Agents now support SNMP v3 Context values in the scan configuration.
- Agents now try harder to recover from error cases during installation.

v1.3.1
2020-01-26
- A race condition was resolved that could leave abandoned chrome.exe processes after a scan.

v1.3.0
2020-01-07
- Version 1.3.0 is a rollup of post-1.2.0 point release work.

v1.2.3
2019-12-19
- The agent now tracks virtual hosts better for reported HTTP and TLS services.

v1.2.2
2019-12-19
- Support for the BACnet protocol has been added.

v1.2.1
2019-12-13
- The protocol detection engine has received a number of small improvements (mongod recognition among others).

v1.2.0
2019-12-01
- Version 1.2.0 is a rollup of post-1.1.0 point release work.

v1.1.15
2019-12-01
• Improved normalization of wireless network fields for the wlan-list probe.

**v1.1.14**

2019-11-27

• Additional bug fixes for SNMP processing.
• Initial support for the wlan-list probe module.

**v1.1.13**

2019-11-26

• Better support for truncated HTTP responses.

**v1.1.12**

2019-11-24

• Invalid SNMP responses are now handled more efficiently.

**v1.1.11**

2019-11-24

• A bug that could lead to memory exhaustion when Max Group Size was set to zero has been resolved.

**v1.1.10**

2019-11-23

• A bug in the SNMP probe that could result in the scan missing the last round of enumeration results has been fixed.
• Scans that result in excessive memory usage will now automatically upload a heap profile to enable support diagnostics.

**v1.1.9**

2019-11-22

• Improved error handling and logging, minor performance increase.
v1.1.8

2019-11-22

- Reduced memory usage on scan reply deduplication.

v1.1.7

2019-11-19

- Cisco-specific MIBs are now enumerated for CAM/MAC table enumeration.
- SNMP v2 is now queried two ways by the SNMP probe to improve device compatibility.
- SNMP v3 authenticated enumeration is now available.

v1.1.6

2019-11-19

- A number of small bugs in the SNMP probe have been resolved.

v1.1.5

2019-11-18

- The SNMP probe will now try to obtain the full interface and MAC address list from each asset.

v1.1.4

2019-11-14

- Miscellaneous fingerprinting improvements.

v1.1.3

2019-11-07

- Improved protocol detection for the Click Modular Router daemon.

v1.1.2

2019-11-05
• A bug that could lead to an agent being stuck in “stopping” status when a scan is stopped has been resolved.

v1.0.15

2019-11-04

• Scans now support the Max Group Size option to limit the number of concurrent scan targets.
• The MAC address prefix database and various other dependencies and fingerprints were updated.

v1.0.9

2019-10-24

• The SYN probe now sends retries using the same source port and sequence number to minimize duplicate responses. MAC address fingerprints have been updated.

v1.0.8

2019-10-23

• The SYN probe now retries twice if no RST is received. This improves reliability at the cost of a small increase in scan times. This can be changed by the syn-max-retries parameter in the console.

v1.0.7

2019-10-21

• Scanner performance is no longer reduced when the ARP probe is enabled for non-local scan targets.

v1.0.3

2019-10-06

• The macOS agent now supports additional interface types (loopback and tunnel adapters).
v1.0.2

2019-10-02

- A race condition was fixed that could cause an agent to crash mid-scan. Affected scans would have an error status of task lost to agent restart.

v1.0.1

2019-10-01

- Rumble Network Discovery is out of Beta with version 1.0.0!
Older scanner release notes

Scanner release notes

Starting with version 1.7.9 all release notes have been consolidated into one page.

v1.7.8

2020-05-23

• Fingerprint updates.

v1.7.7

2020-05-22

• Fingerprint updates.

v1.7.6

2020-05-14

• Corrects inconsistent use of the new service attributes when processing the dynamic MAC address filter.

v1.7.5

2020-05-14

• Asset and Service attributes have been normalized. All keys are now camelCase and most service attributes are now prefixed by the protocol name.

v1.7.4

2020-05-13

• Support has been improved for the following database protocols: Memcached (TCP), CouchDB, Cassandra, Redis, ElasticSearch, Riak (TCP/HTTP), MySQL, PostgreSQL, MongoDB, MSSQL, and Oracle.
v1.7.3
2020-05-07

- Cisco HSRP MAC addresses are now ignored for the purposes of asset correlation.
- Updated Ethernet fingerprints.

v1.7.2
2020-05-06

- A bug in the mDNS probe that could lead to a hung scan on certain platforms has been resolved.

v1.7.1
2020-05-06

- Updated Ethernet fingerprints.

v1.7.0
2020-05-04

- The bundled npcap driver has been upgraded to version 0.9991.
- The TLS probe now reports tls.notBeforeTS and tls.notAfterTS fields as unix timestamps.
- Updated Ethernet fingerprints.

v1.6.10
2020-05-03

- Updated Ethernet fingerprints.

v1.6.9
2020-05-01

- Support for Recog development with --fingerprints and --fingerprints-debug options.
• The Switch Topology and Network Bridges reports are now available for the CLI scanner.
• Updated Ethernet fingerprints.

**v1.6.8**

2020-04-23

• Updated Ethernet fingerprints.

**v1.6.7**

2020-04-23

• The scan engine can now identify TCP services on the scanning system across all platforms.

**v1.6.6**

2020-04-22

• An issue that could lead to the scan engine hanging with misbehaving HTTP services has been resolved.

**v1.6.5**

2020-04-22

• Support for the Rumble Starter Edition.
• Updated Ethernet and BACnet fingerprints.

**v1.6.4**

2020-04-17

• Devices that relay mDNS from other networks (ex: Ubiquiti USG) are no longer associated with the relayed asset information.
• Additional Google Chrome paths are considered for screenshot collection. Snap packages of Chromium are no longer used.

**v1.6.3**

2020-04-14
• An issue that could lead to scans hanging while processing HTTP services has been resolved.

v1.6.2
2020-04-13
• Fingerprint updates for Crestron, ELAN, MAC addresses, and BACnet.

v1.6.1
2020-04-08
• The MAC address database fingerprints have been updated.

v1.6.0
2020-04-06
• Screenshots will now limit the number of concurrent Chrome processes based on core count, available RAM, and architecture.
  • The bundled npcap build has been updated to version 0.9990.

v1.5.6
2020-04-04
• The RDP probe now collects the full NTLMSSP response for more platforms.
  • The HTTP probe now collects information about web forms and their inputs.

v1.5.5
2020-03-27
• The SNMP probe no longer reports invalid MAC addresses found in ARP caches or MAC tables.

v1.5.4
2020-03-26
• The TCP probe now handles a wider variety of RDP responses.
v1.5.3

2020-03-26

- The SMB probe now reports subprotocols (smb1, smb2, and smb3) consistently.
- The SMB probe now collects hashing, encryption, and compression methods from SMBv3 servers.
- The SMB probe now reports the server-allocated Session ID for smb2 and smb3.
- The TCP probe now collects NTLM information from Remote Desktop endpoints and reports the protocol as rdp.
- The HTTP probe now collects additional information from VMware SOAP endpoints.
- A race condition in --nopcap mode that led to inconsistent results has been resolved.

v1.5.2

2020-03-20

- A bug that could lead to the HTTP/2 probe stalling during TLS negotiation has been resolved.

v1.5.1

2020-03-14

- Fingerprints have been updated for Ethernet MAC addresses, BACnet vendors, and Enterprise IDs.
- HTTP/1 probes now explicitly disable HTTP/2 upgrades even when advertised. HTTP/2 is handled separately.
- Generic protocol negotiation is no longer attempted on NDMP ports (10,000/30,000). A future release will support improved NDMP detection and negotiation.
- A potential deadlock in the runtime library has been resolved by reverting to an older runtime version.

v1.5.0

2020-03-04

- A NTP probe has been added that reports the clock skew compared to the scanning instance.
• A TFTP discovery probe has been added that requests a non-existent file and stores the response. The TFTP probe supports port ranges.
• An OpenVPN probe has been added that can detect remote instances across multiple ports.
• A dTLS discovery probe has been added that handles both bare dTLS and CAPWAP-encoded variants.
• Microsoft Remote Desktop Gateway instances are now fingerprinted through dTLS and HTTP, reporting the rdg.Transport service key.
• The protocol handlers for NATPMP, WS-Discovery, and UPnP Device XML now parse out specific subfields for easier matches and future fingerprinting efforts.
• The UPnP Device XML parser now triggers a request to download and report the device icon.
• The SYN scanner has been updated to improve reliability and report more accurate progress.
• The HTTP probe now identifies and reports web site icons as base64-encoded images along with their MD5 hashes.
• The HTTP probe now extracts the generator meta tag from HTML responses.
• The HTTP probe now extracts splunkd versions from HTML responses.
• The RPCBind probe now sends a null call to every UDP service and probes the NFS daemon directly.
• VMware ESXi detection has improved and will be used as a fallback in more cases.
• TCP protocol fingerprinting will retry more often on temporary network errors.
• Empty fields in the result structure within the JSON output are now omitted.
• Linux on ARM 64-bit (aarch64) is now a supported platform.
• Improved detection and early rejection of invalid CIDRs.

v1.4.5

2020-02-19

• The SMB probe now records the NTLMSSP response from a wider range of operating systems.
• The HTTP probe now stores the response to GET / and the response after any redirects are followed. Key names for the redirect responses are prefixed by last, such as last.http.code.
• The HTTP probe now handles compression and chunked transfer encoding properly, storing the normalized HTTP body.
• The HTTP probe now reports a banner consisting of the raw HTTP response.
• The HTTP probe now supports collecting environment data from LANDesk Management Agents.
• HTTP screenshots are now only collected when a 2XX HTTP response code is seen.
• HTTP screenshot processing is now more reliable.
v1.4.4

2020-02-16

- The SMB Server GUID attribute is now used to correlate results to assets.
- The SNMP sysName and sysObjectID attributes are now used to unmatch assets that have changed IPs or were mistakenly matched through another attribute (shared bogus MAC addresses or similar).
- Interfaces with no global unicast addresses (including RFC1918) are no longer considered by the ARP and SYN scanners.
- VLAN-tagged frames are now ignored by the SYN scanner resolving an issue where packets could be sent on the wrong interface by mistake.
- SYN scans now have a mandatory delay between retry attempts, which improves reliability and decreases change churn when small network ranges are scanned.

v1.4.3

2020-02-13

- A bug that caused some HTTP requests to be sent with an empty Host header has been fixed.

v1.4.2

2020-02-05

- Version 1.4.2 improves fingerprinting and type classification of Windows operating systems.

v1.4.1

2020-02-04

- Version 1.4.1 resolves a regression where --input-targets was being ignored.

v1.4.0

2020-02-04

- Version 1.4.0 is a rollup of post-1.3.0 point release work.
v1.3.2
2020-02-02

- Support for the new --arp-fast option to send ARP probes at the configured --rate, without additional delays. This option is on by default in AWS VPC environments.
- Support for the new --snmp-v3-context option to specify the SNMP v3 Context for queries.

v1.3.1
2020-01-26

- Support for the --baseline option (-b) to load a previous assets.jsonl and use existing asset IDs in the new assets.jsonl output.
- Support for multiple --import files. This allows multiple scans to be recombined into a single output.
- Support for the --upload option, which uses --api-key, --upload-site, and other API-related options to upload scan data to the Rumble Console.
- A race condition was resolved that could leave abandoned chrome.exe processes after a scan.
- Prevent use or display of a blank virtual host for HTTP servers

v1.3.0
2020-01-07

- Version 1.3.0 is a rollup of post-1.2.0 point release work.

v1.2.3
2019-12-19

- The CLI scanner now generates a protocols.csv file with a simplified list of services and their URLs.

v1.2.2
2019-12-19

- Support for the BACnet protocol has been added.
**v1.2.1**

*2019-12-13*

- Many new fingerprints were added for HTTP and SIP endpoints.
- Asset correlation now occurs after the scan completes, not as it runs, for more consistent results.
- The protocol detection engine has received a number of small improvements (mongod recognition among others).
- Network scans of segments where a device responds to all ARP requests with the same MAC address will now be handled appropriately.

**v1.2.0**

*2019-12-01*

- Version 1.2.0 is a rollup of post-1.1.0 point release work.

**v1.1.15**

*2019-12-01*

- Automatic generation of wireless.jsonl and wireless.csv reports when the wlan-list probe returns results.
- Improved normalization of wireless network fields for the wlan-list probe.

**v1.1.14**

*2019-11-27*

- Additional bug fixes for SNMP processing.
- Initial support for the wlan-list probe module.

**v1.1.13**

*2019-11-26*

- Better support for truncated HTTP responses.

**v1.1.12**

*2019-11-24*
• Invalid SNMP responses are now handled more efficiently.

**v1.1.11**

2019-11-24

• A bug that could lead to memory exhaustion when Max Group Size was set to zero has been resolved.

**v1.1.10**

2019-11-23

• A bug in the SNMP probe that could result in the scan missing the last round of enumeration results has been fixed.

**v1.1.9**

2019-11-22

• Improved error handling and logging, minor performance increase.

**v1.1.8**

2019-11-22

• Reduced memory usage on scan reply deduplication.

**v1.1.7**

2019-11-19

• Cisco-specific MIBs are now enumerated for CAM/MAC table enumeration.
• SNMP v2 is now queried two ways by the SNMP probe to improve device compatibility.
• SNMP v3 authenticated enumeration is now available.

**v1.1.6**

2019-11-19

• A number of small bugs in the SNMP probe have been resolved.
v1.1.5
2019-11-18

- The SNMP probe will now try to obtain the full interface and MAC address list from each asset.

v1.1.4
2019-11-14

- Network topology links are now reported in the `_links.*` asset attributes.
- Miscellaneous fingerprinting improvements.

v1.1.3
2019-11-07

- Additional SSH fingerprints, covering Debian, Ubuntu, Raspbian, and FreeBSD.
- Improved protocol detection for the Click Modular Router daemon.

v1.1.2
2019-11-05

- A bug that could lead to a scan engine hang when the scan is interrupted has been resolved.

v1.0.15
2019-11-04

- The scanner now supports the `--max-group-size` option to limit the host working set (default is 4096).
- A race condition in the `--text` mode output that could rarely lead to a crash has been resolved.
- The MAC address prefix database and various other dependencies and fingerprints were updated.

v1.0.10
2019-10-25
- The scanner now supports the `--overwrite` option to allow reuse of an existing output directory

**v1.0.9**

2019-10-24

- The SYN probe now sends retries using the same source port and sequence number to minimize duplicate responses. MAC address fingerprints have been updated.

**v1.0.8**

2019-10-23

- The SYN probe now retries twice if no RST is received. This improves reliability at the cost of a small increase in scan times. This can be changed by the `--syn-max-retries` parameter.

**v1.0.7**

2019-10-21

- Scanner performance is no longer reduced when the ARP probe is enabled for non-local scan targets.

**v1.0.3**

2019-10-06

- The macOS scanner now supports additional interface types (loopback and tunnel adapters).
- The macOS scanner no longer prints warnings about unusable interfaces unless the verbose flag is set.
- A bug was fixed that led to the wrong result count being reported when using the `--text` interface of the scanner.

**v1.0.2**

2019-10-02

- A race condition was fixed that could the scanner to crash mid-scan.
• The macOS scanner no longer crashes if an unusable interface is found and the scanner is not run as root.

v1.0.1

2019-10-01

• Rumble Network Discovery is out of Beta with version 1.0.0!
Index

2
2FA: 48

A
Account API: 49 188
Account Configuration: 1
Action: 117
Activating your account: 1
Adaptive Cards: 119
Address ranges: 61
Administrators: 47
Alert messages: 119
Alerts: 1
Alerts: 111 115 116
Alive: 71
Analysis reports: 13
Annotators: 47
API: 188
API key: 188
ARP: 70
Asset data: 5
Assets: 66 69
Automated installers: 27
Automatic queries: 4
AWS: 77
Azure AD: 52

B
Billing: 48
Billing information: 130
Binary signature: 43
Binoculars: 82
Block Kit: 119

C
Caret: 129
Censys Search: 184
Censys Universal Internet Dataset: 185
Change report: 196
Channel: 117
Channels: 115
Cisco ASA: 197
CMDB: 152
Connect: 67
Container: 14
Coverage reports: 80
Credentials: 74
CSV: 72
Curly brackets: 119

D
Dashboard: 1
Data formats: 189
Data retention: 73
Department: 58
Deploy: 14
Detected by: 71
Device type: 70
DHCP: 66
Discovery scan: 60
DNS: 70
Docker: 14
Domain membership: 13
Download: 14
Duplicate assets: 59

E
EC2: 77
Email: 115
Embedded devices: 14
ETL: 152
Events: 116
Explorer: 14
Explorer: 21 60
Export API: 188
Export token: 188
External network discovery: 14

F
Filter: 83
Fingerprint results: 66
Firewall: 197
<table>
<thead>
<tr>
<th>Category</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat networks</td>
<td>59</td>
</tr>
<tr>
<td>G</td>
<td></td>
</tr>
<tr>
<td>Global credential</td>
<td>74</td>
</tr>
<tr>
<td>Google Chrome</td>
<td>198</td>
</tr>
<tr>
<td>Grace period</td>
<td>3</td>
</tr>
<tr>
<td>H</td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td>70</td>
</tr>
<tr>
<td>Host ping</td>
<td>64</td>
</tr>
<tr>
<td>Hostnames</td>
<td>61</td>
</tr>
<tr>
<td>Hostnames</td>
<td>66 69 70</td>
</tr>
<tr>
<td>HP iLO</td>
<td>75</td>
</tr>
<tr>
<td>I</td>
<td></td>
</tr>
<tr>
<td>IAM</td>
<td>77</td>
</tr>
<tr>
<td>ICMP</td>
<td>70</td>
</tr>
<tr>
<td>Insights</td>
<td>1</td>
</tr>
<tr>
<td>Installation</td>
<td>1</td>
</tr>
<tr>
<td>Inventory</td>
<td>66</td>
</tr>
<tr>
<td>Inventory Views</td>
<td>1</td>
</tr>
<tr>
<td>Invoices</td>
<td>131</td>
</tr>
<tr>
<td>IP addresses</td>
<td>66 69</td>
</tr>
<tr>
<td>IPv4</td>
<td>80</td>
</tr>
<tr>
<td>J</td>
<td></td>
</tr>
<tr>
<td>JSON</td>
<td>72</td>
</tr>
<tr>
<td>JSONL</td>
<td>189</td>
</tr>
<tr>
<td>L</td>
<td></td>
</tr>
<tr>
<td>Licensing</td>
<td>130</td>
</tr>
<tr>
<td>Licensing options</td>
<td>131</td>
</tr>
<tr>
<td>Limits</td>
<td>4</td>
</tr>
<tr>
<td>Live assets</td>
<td>130</td>
</tr>
<tr>
<td>M</td>
<td></td>
</tr>
<tr>
<td>MAC addresses</td>
<td>66 69</td>
</tr>
<tr>
<td>Mattermost</td>
<td>115</td>
</tr>
<tr>
<td>Max group size</td>
<td>62</td>
</tr>
<tr>
<td>Max group size</td>
<td>197</td>
</tr>
<tr>
<td>MDM</td>
<td>67</td>
</tr>
<tr>
<td>MFA</td>
<td>48</td>
</tr>
</tbody>
</table>
Microsoft Teams: 119
Missing subnets: 80
MSI package: 27
Multi-factor authentication: 48
Mustache: 119

NameID: 46
NAT: 197
Network bridges: 12
No access: 48
Npcap: 23

Offline: 69
Offline asset expiration: 3
Okta: 55
On-premises: 28
Operating system: 66 70
Organization: 194
Organization API: 188
Organizations: 58 60
Outliers: 12

Packet rate: 61
Password rules: 7
Ports: 62 200
Prebuilt searches: 114
Prescan modes: 64
Private IP addresses: 80
Probes: 62
Profile images: 8
Project: 58
Project assets: 130
Protocols: 199

Queries: 114
Query language: 114
Query library: 114
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raspberry Pi</td>
<td>24</td>
</tr>
<tr>
<td>Recurring scan</td>
<td>65</td>
</tr>
<tr>
<td>Red border</td>
<td>81</td>
</tr>
<tr>
<td>Reinstall</td>
<td>22</td>
</tr>
<tr>
<td>Renew</td>
<td>131</td>
</tr>
<tr>
<td>Reports</td>
<td>1</td>
</tr>
<tr>
<td>REST</td>
<td>188</td>
</tr>
<tr>
<td>Restrictions</td>
<td>134</td>
</tr>
<tr>
<td>RFC 1918</td>
<td>12</td>
</tr>
<tr>
<td>RFC1918</td>
<td>59</td>
</tr>
<tr>
<td>Roles</td>
<td>46</td>
</tr>
<tr>
<td>Router</td>
<td>197</td>
</tr>
<tr>
<td>RTT</td>
<td>70</td>
</tr>
<tr>
<td>Rules</td>
<td>115</td>
</tr>
<tr>
<td>Rules tab</td>
<td>118</td>
</tr>
<tr>
<td>Sample Queries</td>
<td>10</td>
</tr>
<tr>
<td>Sample rate</td>
<td>64</td>
</tr>
<tr>
<td>Scan completed</td>
<td>124</td>
</tr>
<tr>
<td>Scan data</td>
<td>5</td>
</tr>
<tr>
<td>Scan data expiration</td>
<td>3</td>
</tr>
<tr>
<td>Scan result</td>
<td>192</td>
</tr>
<tr>
<td>Scan scope</td>
<td>69</td>
</tr>
<tr>
<td>Scan speed</td>
<td>2</td>
</tr>
<tr>
<td>Scan speed</td>
<td>198</td>
</tr>
<tr>
<td>Scan stats</td>
<td>191</td>
</tr>
<tr>
<td>Scanner</td>
<td>133</td>
</tr>
<tr>
<td>Scans</td>
<td>65</td>
</tr>
<tr>
<td>Scheduled scans</td>
<td>3</td>
</tr>
<tr>
<td>Scope</td>
<td>61</td>
</tr>
<tr>
<td>Screenshot</td>
<td>21</td>
</tr>
<tr>
<td>Screenshots</td>
<td>1534</td>
</tr>
<tr>
<td>Search</td>
<td>66</td>
</tr>
<tr>
<td>Search query rule</td>
<td>124</td>
</tr>
<tr>
<td>Search syntax</td>
<td>83</td>
</tr>
<tr>
<td>Secondary addresses</td>
<td>69</td>
</tr>
<tr>
<td>Self-hosted</td>
<td>28</td>
</tr>
<tr>
<td>Sendgrid</td>
<td>115</td>
</tr>
<tr>
<td>ServiceNow</td>
<td>152</td>
</tr>
<tr>
<td>Services</td>
<td>663</td>
</tr>
<tr>
<td>Sign up</td>
<td>7</td>
</tr>
<tr>
<td>SIP</td>
<td>197</td>
</tr>
<tr>
<td>Site</td>
<td>6669</td>
</tr>
</tbody>
</table>
Site: 194
Sites: 58 60
Sites: 2
Slack: 115 119 129
Splunk: 164
SSO: 46 55
Stale asset expiration: 3
Starter Edition: 134
Stateful firewalls: 60
Statistics: 82
Subnet analysis: 81
Subnet sampling: 64
Subnet size: 64
Subnet utilization: 12
Subnets: 66 80
Superuser: 47
Switch topology: 11 13
System events: 111
System requirements: 1

T
Tasks: 65
Tasks page: 65
TCP services: 70
Team member: 46
Templates: 119
Time To Live: 62
Two-factor authentication: 48

U
UAC: 14
UDP services: 70
Unmapped: 11
Upgrade: 22
UUID: 194

V
Verifier: 43
Viewers: 47
Virtualized system: 14
VMware: 14
VPN gateways: 60

W
Web screenshots: 1
Webhook: 115
Webhooks: 119
WiFi: 68
Wireless networks: 3