TX Protect

Managing Threats

Version 3.20, 2025-01-10

Table of Contents

Blocking 5
Rules 7

Matched Threats

The **Matched Threats** table provides data for each threat that matched the rule.

The table lists the other rules that were matched by the threat:

- Clicking a rule name in the **Rules** column displays that rule's activity page.
- Hovering over a rule in the **Rules** column also highlights all instances of the same rule in the other rows.
- Clicking a rule in the **Description** column displays the properties for that rule.
- The **Match Events** column shows the number of times traffic matched a rule within the selected time range and its change over time. A significant value could indicate a security problem.

You can drill into each threat to display its Entity Details page.

Activity

The **Activity** table lists each attack and the time it occurred. You can drill into each threat to display its Entity Details page.

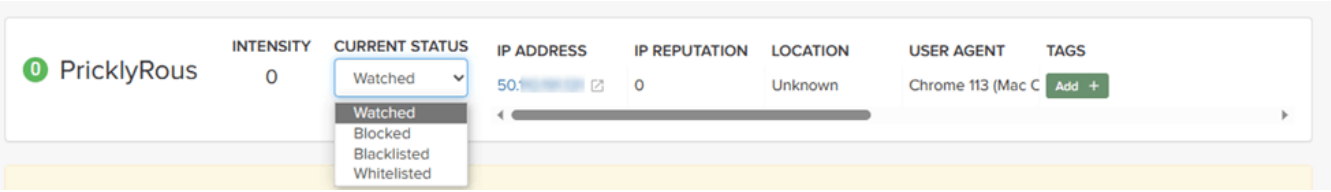
If you are unfamiliar with the Status icons, you can hover over the icon in the **Status** column to see its activity.

Managing threats

If your account has sufficient permissions, you can manually allow, block or deny entitie from the threat's Entity Details page, IWAF Settings page, or by using the ThreatX API.

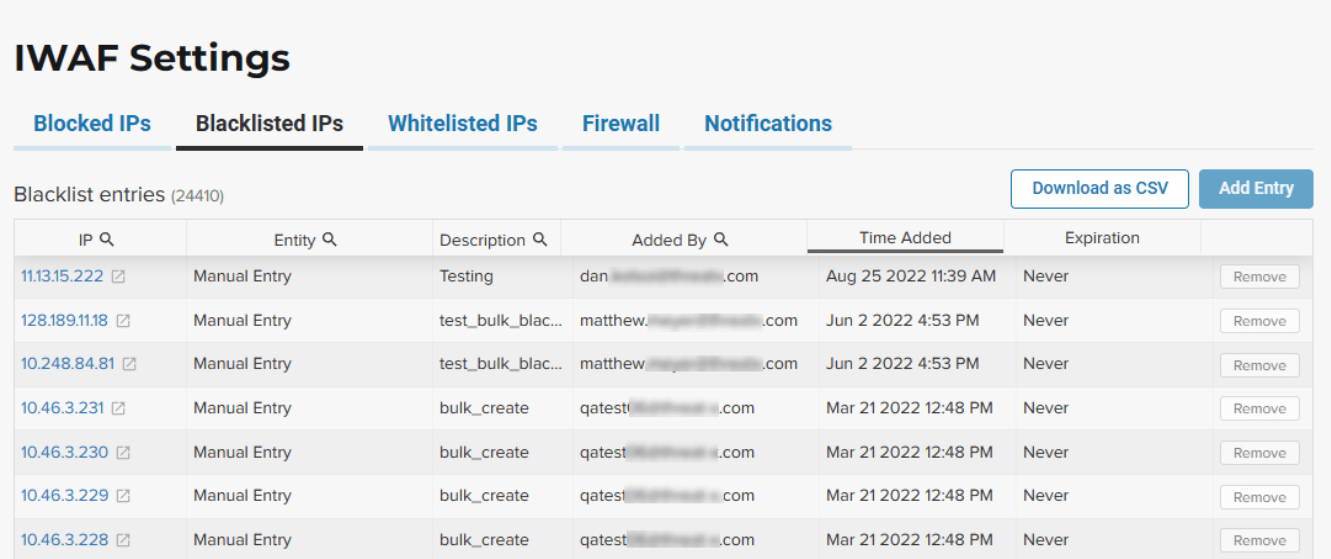
Entity Details

If the threat has interacted with your sites, you can add them to the list: . Click the threat in the Dashboard or other location to open its Entity Details page. . Click **[Current Status]** . Change it to the desired list.



To remove a threat from a list, open **[Current Status]** and select a different list or Watched.

IWAF Settings



- In each tab, use **[Add Entry]** to add an IP address or CIDR range to the list. You are prompted to provide a reason for the action.
- Use the **[Remove]** in the entity's row to manually remove an entity.

ThreatX API

□□ Manage Lists with the `List` ThreatX API Endpoint

URL

`api.threatx.com/tx_api/v2/list`

List the IP addresses currently within the list

- `list_blacklist`
- `list_blocklist`
- `list_whitelist`

Return the details of a single IP entry

- `get_blacklist`
- `get_blocklist`
- `get_whitelist`

Add a single IP address or CIDR

- `new_blacklist`
- `new_blocklist`
- `new_whitelist`

Add one or more new entries

- `bulk_new_blacklist`
- `bulk_new_blocklist`
- `bulk_new_whitelist`

Delete a single entry

- `delete_blacklist`
- `delete_blocklist`

- `delete_whitelist`

Delete one or more new entries

- `bulk_delete_blacklist`
- `bulk_delete_blocklist`
- `bulk_delete_whitelist`

Remove an IP address from the block list

Request

```
$ curl https://api.threatx.com/tx_api/v2/lists \
  --header 'Content-Type: application/json' \
 --data @- <<EOF
{
  "command": "delete_blocklist",
  "token": "<api_token>",
  "customer_name": "<tenant_name>",
  "ip": "1.2.3.4"
}
EOF
```

Response

```
{"Ok": "Blocklist entry for IP 1.2.3.4 removed"}
```

🎓 Add an IP address to the Blacklist

Request

```
$ curl https://api.threatx.com/tx_api/v2/lists \
  --header 'Content-Type: application/json' \
  --data @- <<EOF
{
  "command":"new_blacklist",
  "token":"<api_token>",
  "customer_name":"<tenant_name>",
  "entry":{
     "ip":"1.2.3.4",
     "description":"Test Blacklist",
      "created":1
  }
}
```

```
EOF
```

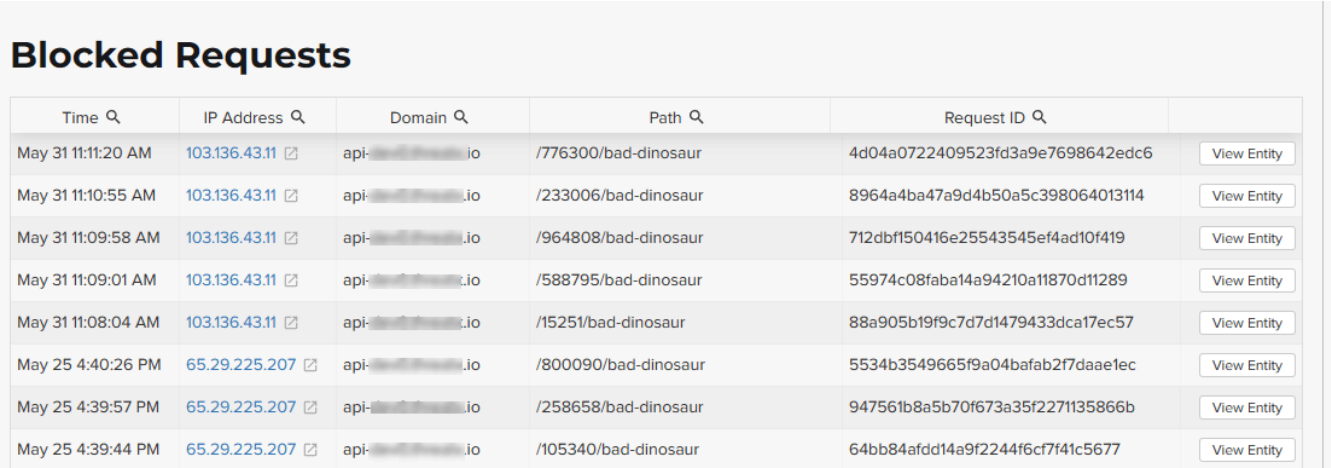
Response

```
{ "Ok": "Blacklist entry for ip 1.2.3.4 added" }
```

Blocking

Blocked Requests

The Blocked Requests page lists the requests that were blocked and relevant data about when and where the attack occurred that caused the threat to be blocked.



Blocked Requests

Time	IP Address	Domain	Path	Request ID	
May 31 11:11:20 AM	103.136.43.11	api-[redacted].io	/776300/bad-dinosaur	4d04a0722409523fd3a9e7698642edc6	View Entity
May 31 11:10:55 AM	103.136.43.11	api-[redacted].io	/233006/bad-dinosaur	8964a4ba47a9d4b50a5c398064013114	View Entity
May 31 11:09:58 AM	103.136.43.11	api-[redacted].io	/964808/bad-dinosaur	712dbf150416e25543545ef4ad10f419	View Entity
May 31 11:09:01 AM	103.136.43.11	api-[redacted].io	/588795/bad-dinosaur	55974c08faba14a94210a11870d11289	View Entity
May 31 11:08:04 AM	103.136.43.11	api-[redacted].io	/15251/bad-dinosaur	88a905b19f9c7d7d1479433dca17ec57	View Entity
May 25 4:40:26 PM	65.29.225.207	api-[redacted].io	/800090/bad-dinosaur	5534b3549665f9a04bafab2f7daae1ec	View Entity
May 25 4:39:57 PM	65.29.225.207	api-[redacted].io	/258658/bad-dinosaur	947561b8a5b70f673a35f2271135866b	View Entity
May 25 4:39:44 PM	65.29.225.207	api-[redacted].io	/105340/bad-dinosaur	64bb84afdd14a9f2244f6cf7f41c5677	View Entity

The **Request ID** is a random string generated to help identify every request that passes through your ThreatX sensors. This request ID is visible on every allowed request in the response header, and also is presented in the 403 message of every blocked request.

Request IDs are useful for investigating issues or blocked requests, and can be given to the ThreatX SOC if more assistance is needed. ThreatX SOC retains the logs of all suspicious and malicious requests for 90 days, and IDs for those requests remain searchable during that time.

Click **[View Entity]** to be taken to that entity's Entity Details page, where you see the full details of the request that was blocked.

Managing listed IP addresses

Dashboard	**Settings › IWAF › Firewall**
ThreatX API	`api.threatx.com/tx_api/v2/lists`

Lists are used to **block**, **deny**, or **allow** IP addresses.

List	Entries
Block List	Temporary *(automatic removal)*
Black List	Permanent *(manual removal)*
White List	Permanet *(manual removal)*

The lists are often managed by your analysts and, therefore, the procedures to manage the lists are provided in the ThreatX Managed API and Application Protection Platform Analyst Guide.

Managing Risk-Based Blocking

Dashboard	Navigate menu:[Settings]
ThreatX API	`api.threatx.com/tx_api/v2/customer`

Risk-Based Blocking Settings

Risk-Based Blocking Timeout

Length of time a threat is blocked. Applies only to those threats that are blocked automatically. Default is 30 minutes.

Risk-Based Blocking Threshold

Rsk Level score

Any threat that meets or exceeds the score is blocked automatically. lock Embargoed Countries

Block TOR Exit nodes

When enabled, all incoming traffic from a TOR Exit node is not allowed. Tor Exit Nodes are the gateways where encrypted Tor traffic hits the Internet.

Rules

Managing rules

ThreatX rules can specify firewall behavior required for your business's individual needs, such as restricting certain resources to company IP addresses or limiting the number of failed login attempts to an application developed in-house.

A ThreatX *rule* is a set of Boolean conditions that, when true, implement the rule's defined action and risk level. ThreatX rules can watch, temporarily block, permanently block, interrogate, or tarpit suspicious traffic. The action is implemented by the sensor.

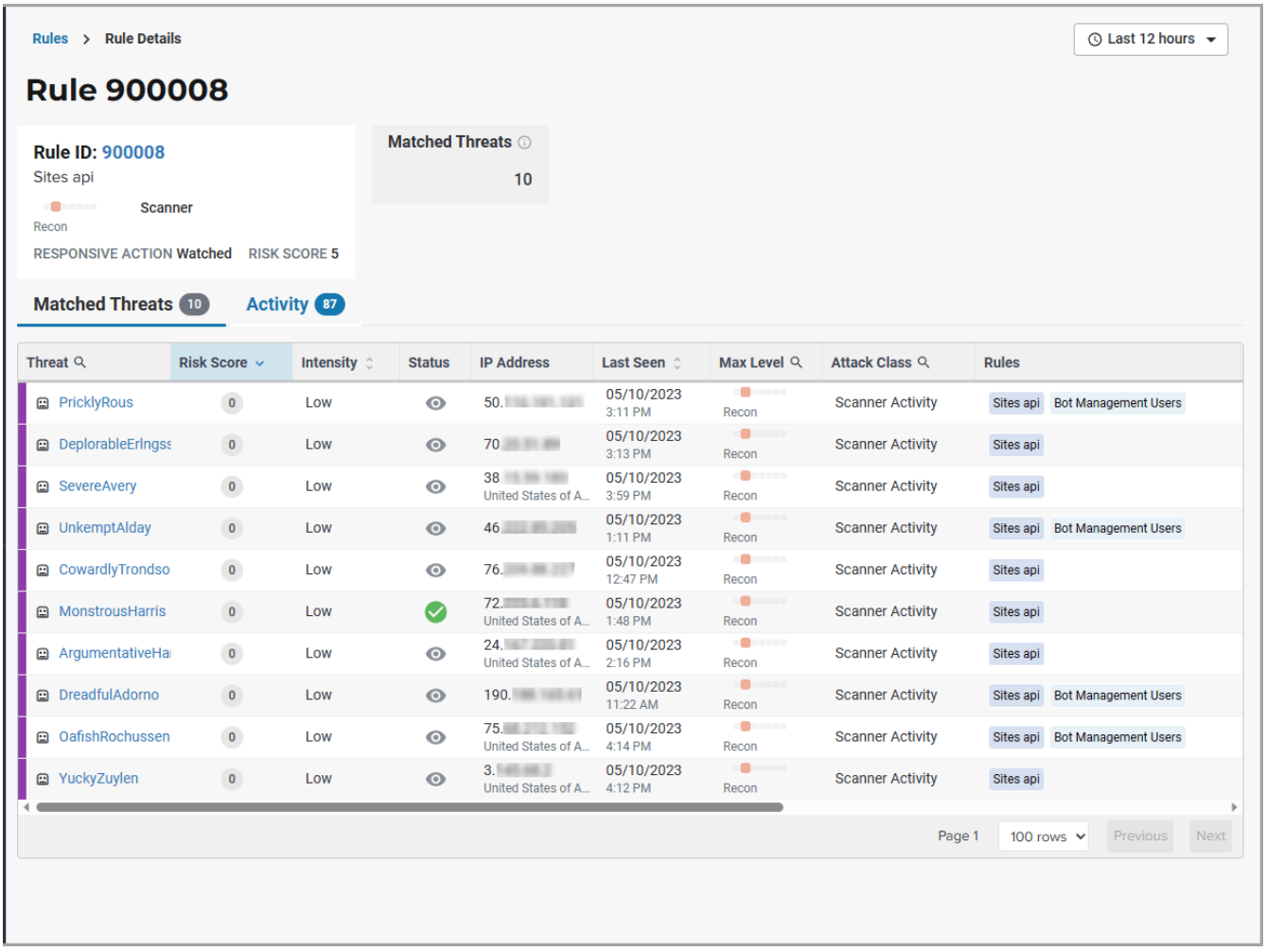
You can add, modify, and delete rules, and view's rule's activity to determine its effectiveness.

To access rules in the ThreatX user interface, navigate to menu:Settings[Rules*. You can also manage custom rules using the ThreatX API api.threatx.com/tx_api/v2/rules endpoint.

Rules can be complex. Creating or modifying a rule could have unintended consequences. You can request the ThreatX SOC group to create rules or modify any rule in the ThreatX platform to meet the specific needs and behavior of your environment.

Rule Activity

The **Rule Activity** page, shown as Rule Details, provides data about the threats that matched the rule. This page is accessible from other pages by clicking a rule name in the **Rules** column.



You can use the data to determine the effectiveness of the rule and if a change is needed. For example:

- Does a threat match too many rules?
- Does the rule catch the expected threats?

Metrics

The **Rule ID** tile provides the ID of the rule, description and the following data:

□□ Rule ID Tile Data

State that the rule assigns to a threat
: The state is shown as a bar with text underneath. The state displays in various pages as the Max Level. In the previous figure, the state is Recon.

Classification that the rule assigns to a threat
: The classification displays in various pages as the attack class. In the previous figure, the classification is Scanner.

Responsive action
: Action that the rule performs when responding to a threat. The action displays in various pages as the status.

Risk Score
: Score that the rule assigns to a threat.

Matched Threats
: Shows the total number of threats that matched the rule in the selected time frame.

Rule details

To view a rule's details, navigate to **Settings** › **Rules** then click **[Edit Rule Details]** for a specific rule.

Description
: Text that describes the intended behavior or logic a rule match is intended to indicate. This information is displayed in the ThreatX user interface when your custom rule is matched.

Tag Name
: Text that identifies a rule when a description is long

Classification
: Describes the kind of attack or behavior it is meant to detect. See Rule Classifications table.

State
: Assumed objective. Maps the intent to a stage on ThreatX Web Application Kill Chain. Rule States table.

Risk
: Assigned risk level (0 to 100) at which the entity triggers a rule. The higher the rule's risk, the fewer hits it takes to block a given entity. The biggest factor in determining entity risk is the total risk assigned by rules they trigger, which you can see in Risk Assignment table.

Action
: Action for the sensor to perform. See the Rule Actions table.

Visual
Tab that displays the rule in a graphical format.

JSON tab
Tab that displays the rule in a JSON format.

Beta
If selected, the platform **does not process matches** to rhe rule.

Table 1. Rule Classifications

Classification	Description
`Undefined`	Unknown attack type.
`SqlInjection`	SQL injection attack. Attempt to exploit input form or un-sanitized input vector to the SQL backend.
`XSS`	Cross Site Scripting. Attempt to execute unauthorized code in the user's context.
`RFI`	Remote File Inclusion. Attempt to have the application server evaluate or include unauthorized 3rd party content or code.
`SessionHijacking`	Attempted unauthorized takeover or co-opting an existing authenticated session.
`DirTraversal`	Directory traversal. Attempt to have the application server evaluate or include unexpected and potentially sensitive content
`Evasion`	Attempt to evade detection of malicious commands or code with various encoding tricks.
`TrojanActivity`	Indications of known malicious software.
`InfoDisclosure`	Information disclosure. Attempt to inappropriately disclose sensitive information about a server, application, or other.
`ExecutableCode`	Indications of an attempt to upload or execute executable code in a malicious context.
`PasswordGuessing`	Attempted word list or online brute-force to gain access to known application accounts.
`PasswordSpraying`	Attempted use of known default, weak, or compromised passwords to gain unauthorized access.
`CredentialStuffing`	Attempted discovery or unauthorized use of compromised user credentials username and password.
`FormSpam`	Abuse user-generated content such as response forms, comments, and reviews for unauthorized promotional purposes.
`OSDetection`	Operating System detection. Attempt to fingerprint server operating system for use in targeting future attacks.
`ContentEnumeration`	Enumerate site pages or content for abusive or malicious purposes.
`PluginEnumeration`	Enumerate content-management-system plugins, software components, and more for use in targeting future attacks.
`UsernameEnumeration`	Attempt to collect authorized users for future malicious purposes.
`ResourceExhaustion`	Attempt to exhaust server CPU and memory resources to negatively impact legitimate services.
`TrafficFlood`	Attempt to exhaust server bandwidth resources to negatively impact legitimate services.

Classification	Description
HighVolume	High request volume. Suspicious or maliciously high volume of requests, bandwidth used, or other volume with the intent to negatively impact legitimate service.
ErrorRate	Elevated error rate. Indication that an offending entity might be performing malicious actions as evidenced by an increase in HTTP errors returned by the server.
KnownVulnerability	Attempt to exploit a known vulnerability in the application.
CSRF	Cross Site Request Forgery. Attempt to abuse a user or user-agent context to perform unauthorized actions on behalf of logged-in user.
EscalationOfPrivilege	Attempt to gain unauthorized access or gain permissions otherwise not expected or permitted for a given user.
WebShell	Indicators of malicious code intended to aid in unauthorized access to a web application or server.
BadBot	Known malicious or undesirable web bots, spiders, scrapers, or other entities.
CommandInjection	Attempt to trigger server-side execution of unauthorized commands through a web form or application.
CryptoMining	Cryptocurrency mining. Attempt to use server resources for unauthorized cryptocurrency related activities.
Toolkit	Hacker toolkit. Indicators of known security or hacker toolkit attempting access to the web application.
BotnetActivity	Indicators of known botnet or infected hosts attempting access to the web application.
BusinessLogicAbuse	Abuse of custom business logic or application workflow to commit various fraudulent or unauthorized activity.
LFI	Local File Inclusion. Attempt to have the application server evaluate or include local, potentially sensitive, content.
MaliciousInclude	Attempt to introduce known malicious code for execution in user or user-agent context.
SoftwareDetection	Attempt to fingerprint application technology and frameworks for future malicious use.
ProgrammaticAccess	Indicators of programmatic or automated access attempts for the web application.
CustomerRule	Custom rules to enforce business logic which might not fit in another rule category.

Table 2. Rule States

State	Description
Reconnaissance	Basic data collection
Scanning	Scanning for content and known vulnerabilities
Web Application Mapping	Find possible weak points
Brute Force Attack	Gain unauthorized access
Denial of Service	Disrupt application availability

State	Description
`Exploitation`	Exploit application weaknesses
`Malware Communication`	Consolidate position on a compromised server

Table 3. Rule Risk Assignments

Range	Description
`[0-10] (Low)`	**Best used to track interesting, but not notably suspicious requests.** Rules with this risk level never result in a block unless combined with a higher risk rule.
`[11-90] (Medium)`	**Should be used for most rules.** Multiple matches are required before blocking an entity. This reduces the likelihood of blocking a benign entity (which sent a few odd-looking requests).
`[91-100] (High)`	**Indicates a known vulnerability or probable malicious threat.** A request triggering a risk 91+ rule quickly increases the entity's risk score and results in a block.

Table 4. Rule Sensor Actions

Action	Description
`Track`	Begin or continue tracking a risk score for the offending entity, based on the risk assigned to this rule and other factors. This is the default and recommended action for most custom rules.
`Block`	Immediately block the request and track a risk score for the offending entity. Blocking rules are best used to stop known malicious behavior, "virtually patch" known vulnerabilities, etc.
`Tarpit`	Limit the speed at which the offending entity receives responses and tracks a risk score for the entity. Tarpit actions are best used to discourage scanning or scraping behavior without immediately blocking the traffic.
`Interrogate`	Challenge an offending entity with a cookie and attempt to fingerprint the user-agent. Interrogation allows a custom rule to explicitly invoke anti-bot mitigations for an entity.

Rule format

A rule must define at least one **criteria**: a boolean expressions that consist of an **attribute** and a **supplied value**.

Some criteria have an operator to determine how the value is compared. However, if there is **no operator available**, the criteria can still be met if that **attribute value** is **equal** to the **checked value**.

Criteria are contained within a **group**: a boolean expression derived from comparing the results of all those criteria. The group can have multiple nested levels to support complex conditions.

Rule Evaluation Operators

or Rule is matched if any of the criteria are true.

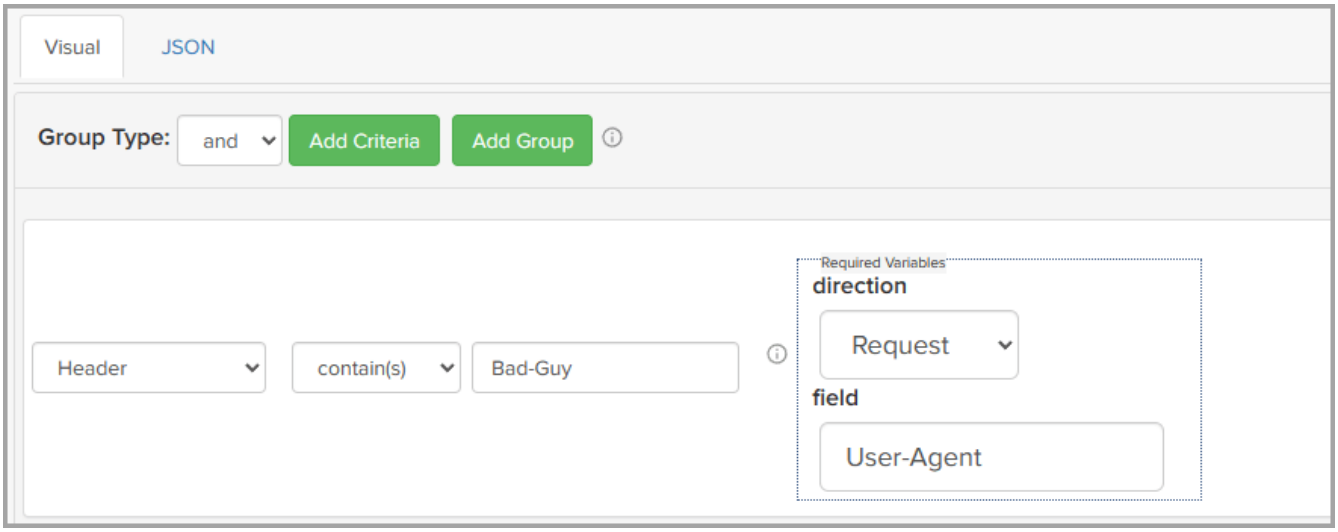
and Rule is matched if all the criteria are true.

not Rule is matched if none of the criteria are true.

A `true` state is also known as a **match**.

When a rule is matched, a `classification`, `state`, and `risk level` is assigned to the threat. The sensor also then performs the configured `action`.

The following figure shows the **Visual** tab with the **Group Type** operator set to **and**, and one criteria entry with **Header** as the attribute. The `Header` attribute has two required variables: `direction` and `field` The direction determines that headers in requests only are checked, and that the header name is `User-Agent`. For this entry to be true, the header name must contain `Bad-Guy`.



Additional Operators

Some criteria attributes have additional operators available:

contain(s) Expression is true if the value includes the provided value.

equal(s) Expression is true if the value is equal to the provided value.

Starts with Expression is true if the value begins with the provided value.

Regex Expression is true if the value equals the provided regular expression.

Group Allows you to add a group within the criteria.

Types of Criteria

There are three types of criteria: * Entity * Request * Response

Table 5. Entity Criteria

Attributes	Description	Example
`Source IP`	Checks if the entity's IP address matches at least one of the provided list of IPv4 addresses or CIDR networks.	127.0.0.1/24, 127.0.1.1, 127.54.3.64/26
`Countries`	Uses Internet geolocation to check if the entity's IP address resolves to at least one of the provided countries. The criteria take a comma-separated list of two-letter country codes (ISO alpha2).	PR,RU,UA

Table 6. (Incoming) Request Criteria

Attributes	Description	Example
`Hostname`	Checks if the Host header sent in a HTTP request matches the provided name.	`example.com`
`URI`	Checks if the “path” portion of URI sent in HTTP request matches the provided path.	`/wp-login.php`
`Arguments`	Checks if the “URL query” or form-encoded POST data sent in HTTP request matches the provided argument.	`wp-submit=Log+In`

Attributes	Description	Example
`Named Argument`	Checks if a specific "URL query" or form-encoded POST data key + value pair sent in HTTP request matches the provided argument. Requires an argument name.	`Log+In`, `name:wp-submit`
`Method`	Checks if the HTTP method used in the request matches the selected method.	`POST`
`Header`	Checks if a specific HTTP header value matches the provided header. The **direction** must be **Request** and the **field** must contain the header name.	`Mozilla/5.0 (Chrome) direction:Request header-name:User-Agent`

Table 7. Response Criteria

Attributes	Description	Example
`Response Code`	Check the HTTP response code/status code returned by the application.	`401`
`Header`	Check if a specific HTTP header value matches. The **direction** must be **Response** and the **field** must contain the header name.	`JSESSIONID= direction:Response header-name:Set-Cookie`

Rule matching

For a rule to be matched, the condition set by the operator of the group must be true. For example, some of the criteria are matched while others are not. If the group operator is set to **or**, the rule is matched since at least one criterion is matched. If the operator is **and**, the rule would not be matched.