





Anomaly Detection in the Noname API Security Platform



The goal of Noname Security's anomaly detection is to identify user anomalous behavior that indicates potentially malicious attempts to exploit the organization's APIs. By establishing a baseline of normal traffic, Noname Security's anomaly detection can compare incoming requests to the baseline and determine if it is likely to be conducted by an attacker.

Our anomaly detection algorithm identifies anomalous behavior, such as:

-  Using an unexpected field in the API request.
-  Pulling more data from the server than the regular user.
-  Trying to use other user/admin resources.
-  Calling the APIs in an unexpected order.

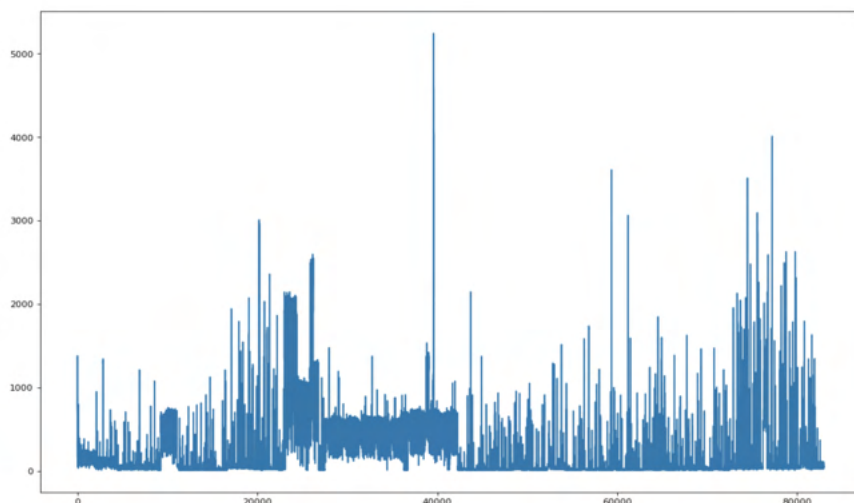
Our anomaly detection algorithm is based on an unsupervised online learning artificial intelligence & machine learning (AI/ML) model that learns the multi-features of the statistical behavior of the traffic and detects abnormal incidents after a fixed learning period. Our model is adaptive to changes in the traffic over time and to anomalies labeled as false positives by users.

During the learning phase, our system parses the customer's data and identifies the different APIs, authentication methods, users, data types, etc. As for each API, the model develops a list of features of the regular user traffic, including the number of API hits, the number of errors generated, the percentage of authenticated requests, the amount of data retrieved from the server, and more. Our algorithm detects user abnormalities by comparing the user's and API's characteristics with the results expected by the statistical model which our algorithm has learned.

Example

Noname Security's anomaly detection identifies users that excessively create more errors than other users. This allows us to identify attacks such as brute-force, path scanning, scraping, etc.

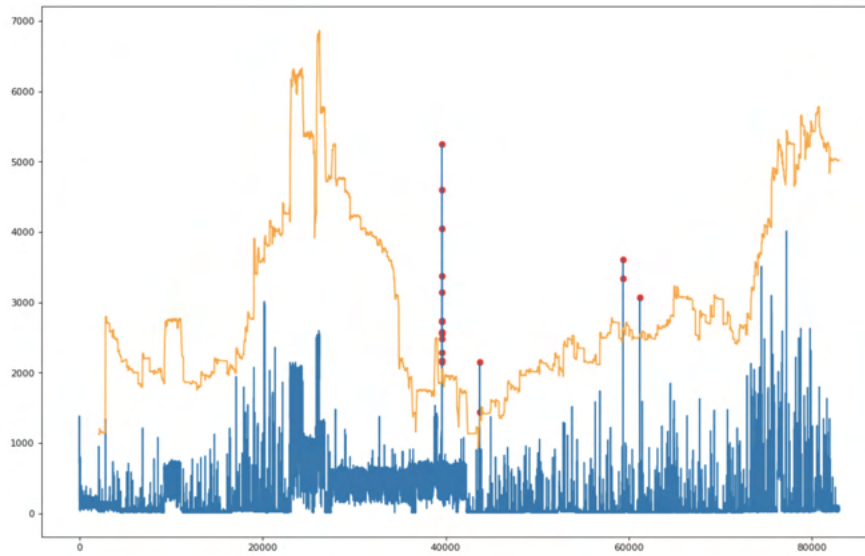
The following plot shows the maximum amount of user errors generated by a user every one-minute cycle in an environment.



There are multiple challenges in identifying anomalies in this scenario:

- ① The model needs to account for data drift when calculating the threshold.
- ② We will want to avoid learning abnormalities during the learning period of the model.
- ③ The learning is conducted in stream, meaning the model never sees the entire data and needs to adjust every time step.
- ④ Alerts must be in real-time, therefore our algorithm cannot rely on future data to predict an anomaly.
- ⑤ To avoid spamming the user, our model needs to learn a statistically guaranteed threshold on the data.

In the plot below, we can see how our model satisfies those requirements by adjusting thresholds according to incoming data.



The orange line shows the threshold function calculated by the model, and the red dots show the anomalies it detected based on that function.

Frequently Asked Questions

What is the necessary learning period for Noname's anomaly detection algorithm?



Most of our algorithms require a learning period of 2 - 7 days. In addition, the algorithm's learning period is also affected by how many different user behaviors were observed during the learning period.

When an anomalous behavior is detected, how long does it take for the alert to be generated?

Our algorithm will create a relevant alert for the client within 30 to 60 seconds, in most cases, from the moment it receives the anomalous traffic.







What are the different types of anomalies detected by Noname?

Noname detects two types of anomalies:

-  Pattern-based anomalies – anomalies that are based on identifying malicious patterns in the traffic such as web exploitation techniques and known malicious user agents such as Command Injection, Path Traversal, and Suspicious User-Agent.
-  Behavioral-based anomalies – anomalies that are based on the learning behavior of users, and identifying abnormal users, such as Excessive API Usage, Range Violation, and Broken Level Object Authorization.

Does the algorithm use a supervised or unsupervised model?

Our algorithms are based on multi-features engineered by conducting a statistical analysis of the traffic, such as:

-  the number of different users that use an API
-  if the API is authenticated
-  the server's response code
-  the amount of data that is pulled by the user
-  the user's IP geolocation
-  the user's user-agent, etc.

Can the user control the sensitivity of the algorithm?

Yes, the user can control the sensitivity of each anomaly by modifying the relevant policy sensitivity. The policy sensitivity is a number between **1(low)** and **5 (high)**; the highest value makes the system the most sensitive) that can be configured for each anomaly policy in the Noname API Security Platform. Our algorithm takes this parameter into account as part of the model.

Can the user mark an issue alerted by Noname as a false positive, and how will it affect the algorithm?






Yes, for improving our anomaly detection, our users can mark relevant issues as "False Positive". When an issue is marked as a false positive, our algorithm takes this into account and adjusts the model according to the input from the user.

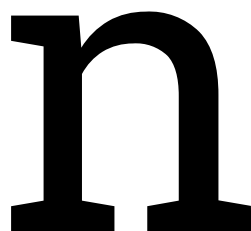
How does Noname avoid “spamming” the client with a user that keeps sending the same attack scenario?

Our algorithm will identify similar issues that keep being triggered on the same user and API. In this case, our algorithm will ignore similar issues for a constant period of time.

How does Noname handle drift/seasonality in the data?

Being an online learning algorithm, Noname Security’s solution needs to address a variety of challenges, such as

-  new APIs
-  new field(s) in existing API(s)
-  change of value type/range in a field
-  server availability issues
-  bugs in APIs that may cause errors (404, 500, etc.) and other challenges to decide which are to be learned and which are not. Noname takes precautions to not learn these abnormalities by requiring a combination of minimal user count, time period, and persistence in order to trigger learning.



About Noname Security

Noname Security is the only company taking a complete, proactive approach to API Security. Noname works with 20% of the Fortune 500 and covers the entire API security scope across four pillars — Discovery, Posture Management, Runtime Security, and API Security Testing. Noname Security is privately held, remote-first with headquarters in Silicon Valley, California, and offices in London.