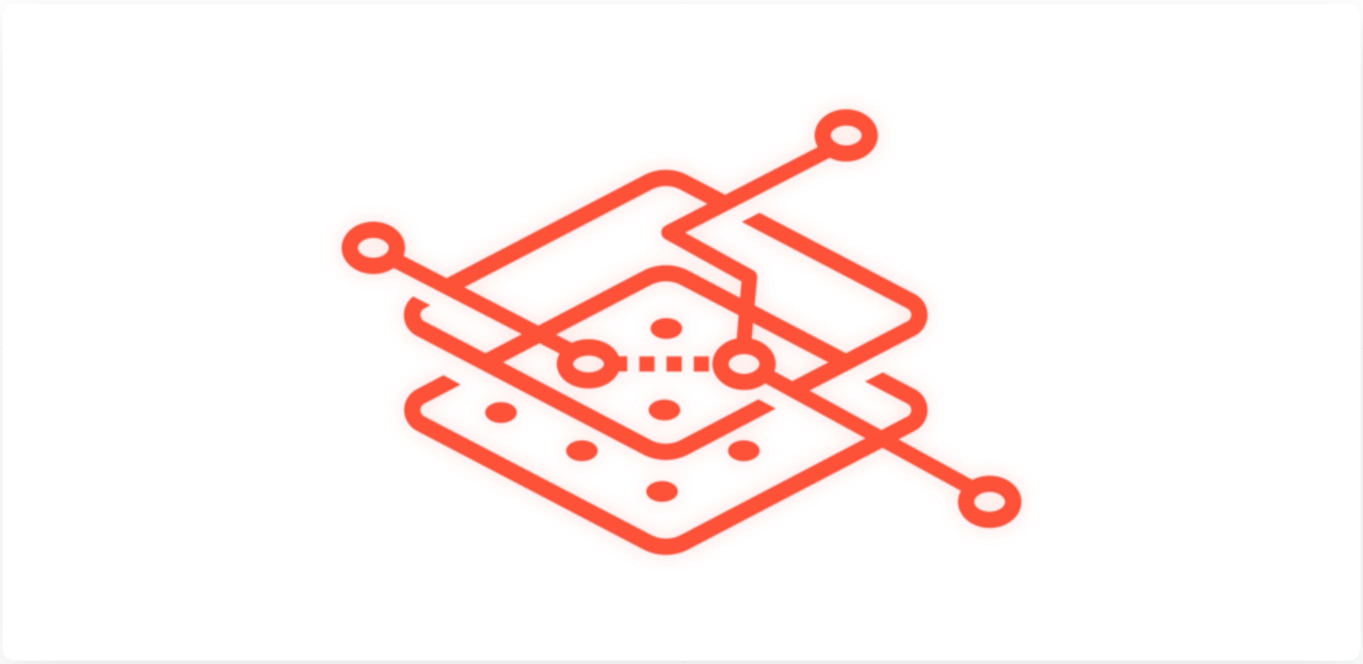
Sign Up

Sign In

technology

TraceAI : Machine Learning Driven Application and API Security

Ravi Guntur // Sanjay Nagaraj



API security

Modern applications are mobile first and are built around cloud native distributed microservices architectures. These architectures have become the basic building blocks for complex and reliable distributed web and mobile applications. Many of these distributed APIs expose the business logic directly over the web and hence the attack surface and attack vectors are very different for these next generation applications. These next generation applications demand next generation Application Security solutions that need to stitch together a distributed application's activity using techniques such as distributed tracing and then use next generation ML techniques to solve important application security problems.

Proactive anomaly detection

Anomaly detection approaches are a practical way to discover unseen strategies of attackers. However, these approaches are prone to very high false positive rates, thereby defeating their very purpose. Most anomaly detection systems flood analysts with false alarms resulting in a cognitive fatigue. Traditionally, the pitfalls of anomaly detection approaches have been addressed via false-positive workflows, better user interfaces and user experience tools. Analyst feedback is taken as a reinforcement on the algorithm and the algorithm tunes itself. However, the engineering cost of such an approach is very high and the product gets very complex leading to reliability and flexibility issues. Secondly, it still requires analysts in the loop to tune the back-end algorithms. Most practitioners think the issue lies with the anomaly detection algorithms, but in reality there is nothing much that the underlying mathematics powering these algorithms can really do in a cyber security setting. These very algorithms have been proven in domains other than cyber security. At Traceable, we believe that the problem lies elsewhere. Our understanding of the nature of data shows that we need to support anomaly detection algorithms with something else. We need to marry specific application security nuances with the mathematics powering these anomaly detection algorithms. This understanding led us to build TraceAI.

TraceAI

TraceAI is the core engine that powers the application security functions of the Traceable Security Portfolio. The foundational principles behind TraceAI are:

- To be able to dissect the AppSec requirement into the constituent entities that are root causes for application vulnerability
- To be able to holistically observe the interaction behavior among these entities
- To be able to discover anomalous patterns in this behavior

Using these guiding principles TraceAI implements a complex behavioral anomaly detector. To power this behavioral anomaly detector, we have identified five entities that interact with one another to result in security vulnerabilities. They are:

- **APIs** – the basic building blocks of the distributed applications
- **API interaction** – how these APIs are wired in order to provide the intended functionality
- **Actors** – users, IP addresses or client-side applications such as mobile apps that use APIs to benefit from the intended functionality or in a malicious case, subvert the intended functionality. The actors may be assigned different roles such as admins, clients, managers etc…
- **API Data** – inputs from actors required for the APIs to come to life. Outputs from APIs in response to the input.
- **Resources** – back-end data or compute functionality requested by actors by calling APIs

TraceAI combines APIs, API interaction, API data, resources and actors holistically to build an actor-centric and API-centric behavioral anomaly product. As a result, common behaviors, outlier behaviors, automaton behaviors (bots), and attack behaviors are discovered from trace data. These behaviors are API-centric as well as Actor-centric.

False positives are eliminated by introducing additional security concepts commonly referred to as Trust-and-Risk models. TraceAI automatically identifies API risk and Actor-trust by observing data relative to a baseline which is again automatically built. The baseline in itself is not a piecemeal integration of various unconnected baselines, but is a holistic combination of various observations modeled using modern machine learning tools. We do not need teams of data scientists and security engineers to support the system as the underlying algorithms automatically learn from data. Once in production, the algorithms run in a self-drive mode, automatically building baselines and generating ML models.

Machine learning algorithms under the hood

The foundation of any machine learning is the depth and breadth of the dataset relative to the number of features used for detection purposes. Traceable agents, deployed inside your application, API gateways, and Kubernetes cluster, automatically collect all the necessary data and build distributed traces that form the data-layer foundation for TraceAI. Traces are continuously collected to become a learning set on logic and data flow from Edge APIs, to internal APIs to backends and third party services and thus generate the features that represent the application context. Having addressed the data challenge via the tracing platform our next challenge was to choose appropriate machine learning technology.

The biggest challenge for machine learning in cybersecurity is having to deal with high-dimensional categorical data of sequential nature, combined with the need for explainability, causality, and attribution.

Because the attackers' strategy constantly changes, supervised learning has its limitations. Most popular ML algorithms fail miserably while simultaneously dealing with these two requirements. Taking these factors into account, we have engineered ML-based solutions by building algorithms ground-up (instead of fully relying on libraries) and adopting a self-supervised learning framework. Our unique way of representing multidimensional categorical information allows us to perform ML-based detection even on certain forms of encrypted data as well.

A key component of the ML-based solution is the ability to discover the API endpoints, and to automatically generate the API spec via data analysis. As a result the expected request/response headers, body parameters and query parameters are not only identified, but are also characterized. This helps build the API spec, violations to which are flagged as anomalies. This completely automated process without the need for manually annotated data is feasible owing to the richness of data captured by the core distributed tracing platform.

Complementing the ability to detect API data anomalies, TraceAI also detects anomalies in API interaction, and anomalies in users accessing these APIs. The User Behavior and Business Logic representation modules capture, represent and build baselines for regular user behavior and expected business logic. Unsupervised ML techniques are employed because these techniques are the only scalable alternatives. Once the baseline models are built, business logic abuse and behavior deviations are captured and surfaced as alerts.

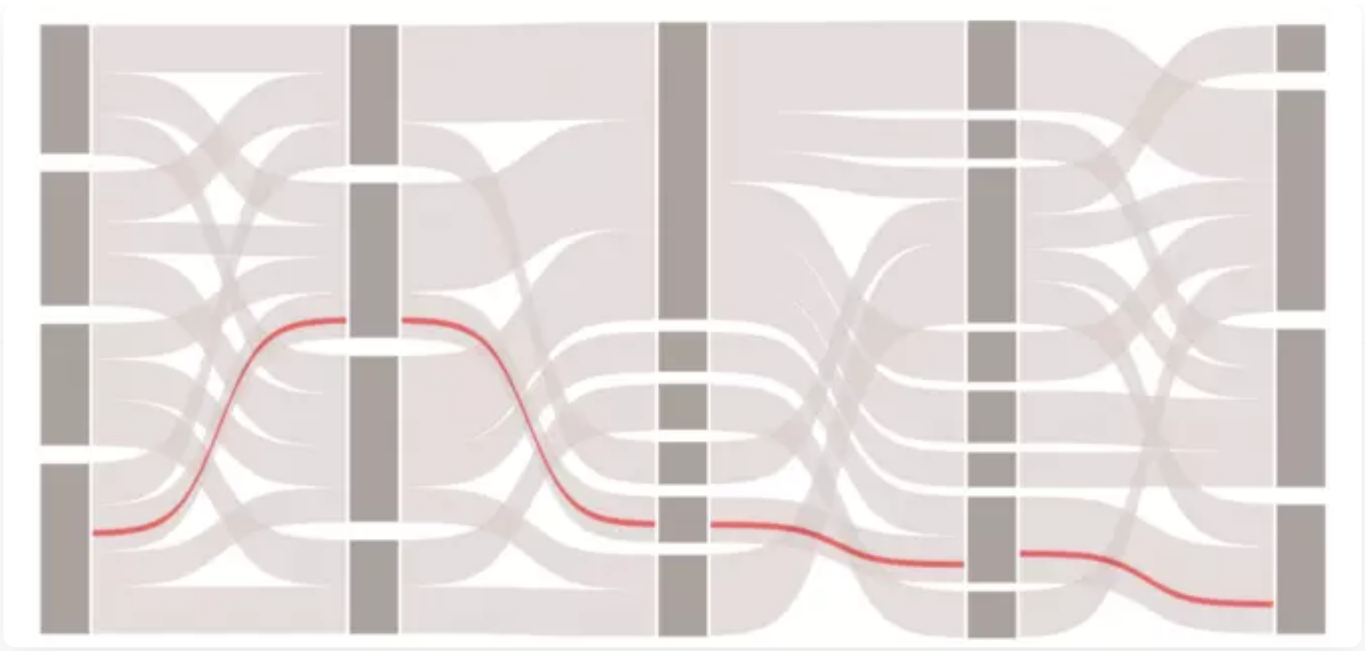
Extending behavioral anomaly detection to predictive threat detection, TraceAI introduces a novel exploratory learning paradigm for API security. This approach harnesses the fundamental principles behind reinforcement learning, adapted uniquely to application and user behavior representation. Using such an approach, we are able to generate potential anomalous patterns even before they occur. By comparing users' activities against these generated patterns, we can identify users who are beginning to tread on a dangerous path. Hence, we can flag these users even before a full-blown attack is launched. As a byproduct, we are also able to quantify known and risky behaviors without manual intervention. Some of the machine learning techniques that we have used in order to power TraceAI are MinHash based embeddings, hierarchical user behavior clustering using LSH and OLAP-style sub-clustering, Bi-Directional RNN variants for time-series anomaly detection, and use of random walks with evolutionary algorithms for explorative learning.

Anomalies in user behavior in the context of business logic

Understanding application behavior and user behavior helps in:

- Identifying violations to known user behaviors
- Discovering novel behaviors
- Grouping users based on their behaviors to create user cohorts
- Determining how exactly APIs are being used in order to build a web application

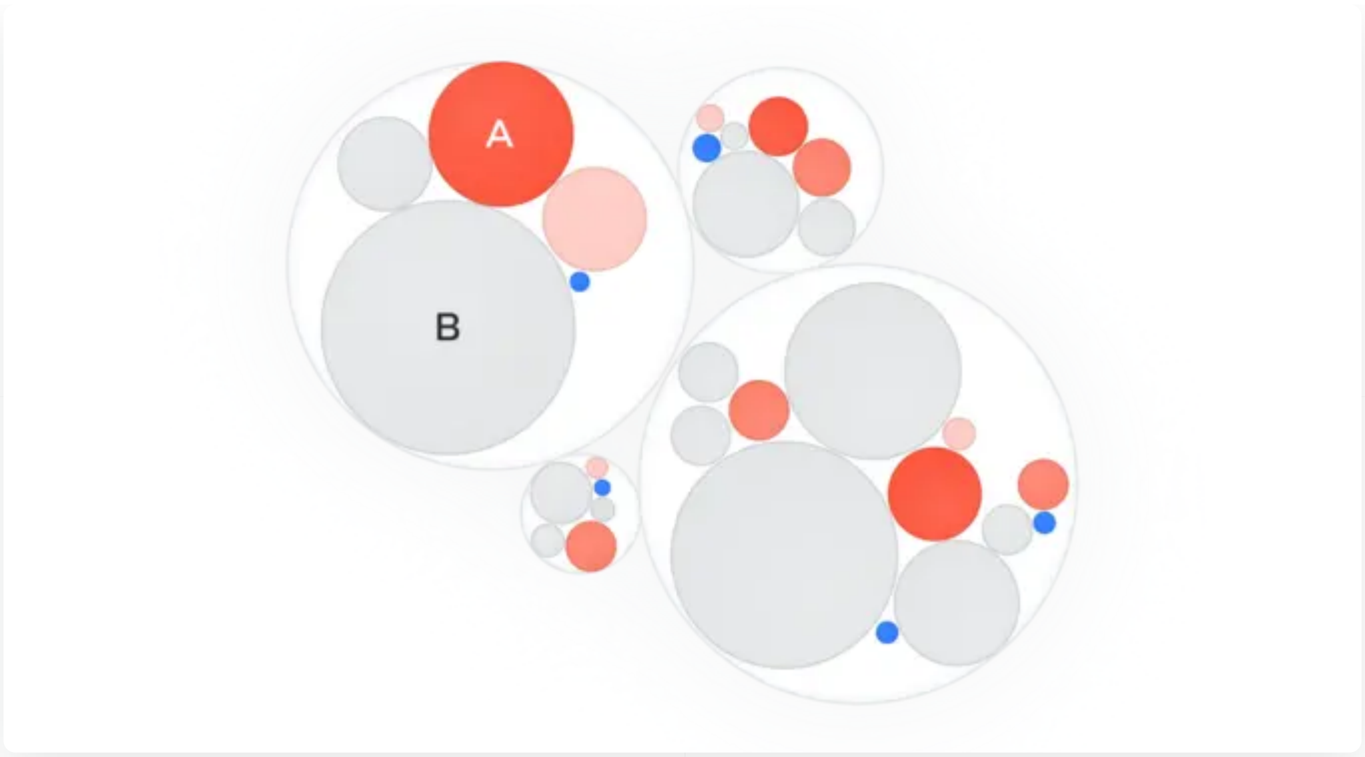
One can represent user cohort behavior as a summarized flow-graph as shown below.



Sample application flow.

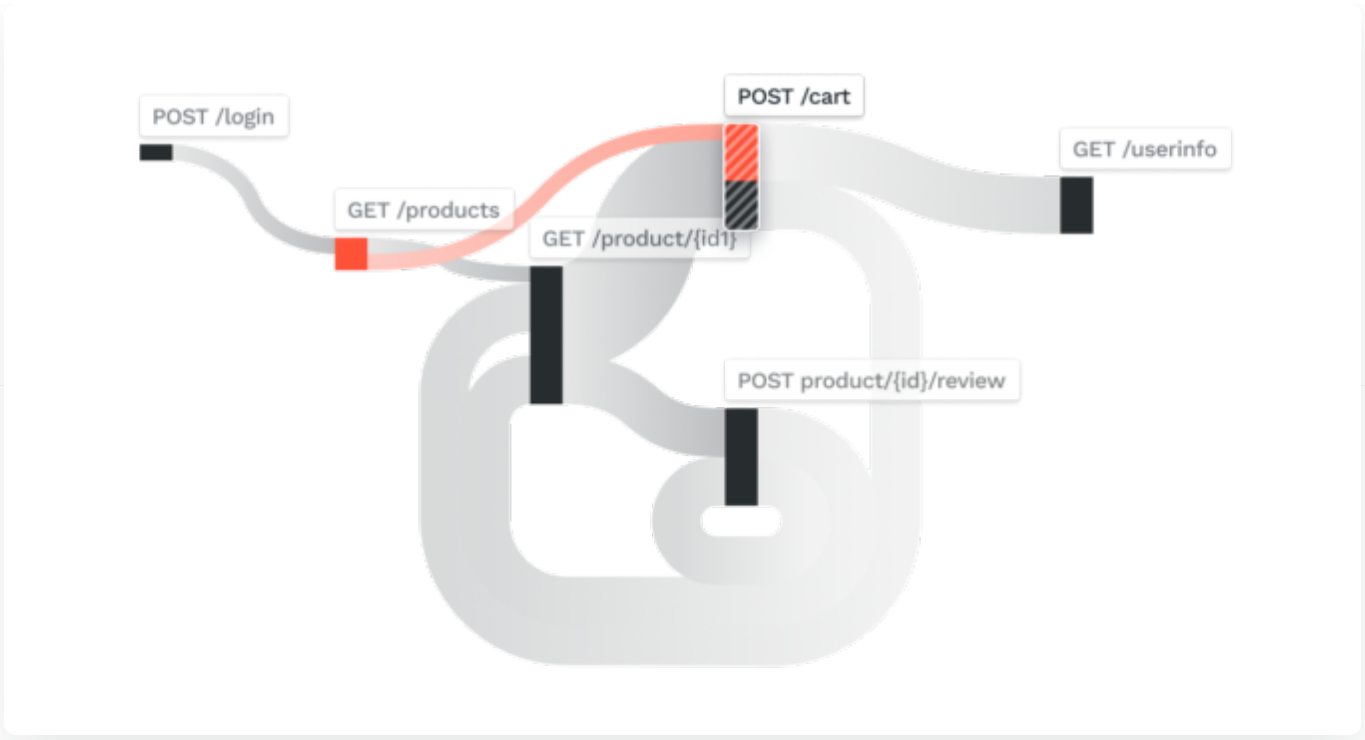
Each node in the graph represents an API that was called during an active user's session. Two nodes are connected if there is a transition from one API to another, and the flow between two nodes represents various statistics such as how many users exhibit such a transition or how often such a transition occurs across user sessions and so on.

In the above figure, there are around 26 different nodes, each representing an API, and the flows connecting these nodes indicate paths of API access based on monitoring users' API call graphs. For a human analyst, this map is very difficult to comprehend and more so to detect that a particular user's behavior has changed compared to other users' behavior or to detect that a particular user's behavior has changed compared to a baseline. Only machine learning algorithms can automatically interpret these kinds of representations and that too at the scale that we have to deal with.



Behavioral clusters

The figure above shows how TraceAI was able to create user behavior groups based on the pattern of API access. In essence TraceAI has peered into the complex flow graph shown above and then identified groups of users based on their respective API call flow behavior. The gray circles represent users who fall under commonly occurring behavior groups. These groups are low priority for an analyst as they are population characteristics. The red circle-A represents a user whose behavior is very similar to the gray circle-B. This means the user in circle-A is accessing the same APIs as users in circle-B but managed to attack one or more APIs. This further implies that the user in circle-A is following the same business logic as many other users but has managed to compromise one or more APIs. This is interesting information for an analyst, because one can now understand what led to the compromise in addition to knowing what was compromised.



Tracking user behavior to detect API attacks

Here, the figure above shows the user's behavior and where exactly the attacks took place. There were authorization bypasses detected in two of the APIs (nodes) colored in red. The analyst now knows where the attack took place and how they got to this stage. The APIs leading up to the attacked API can now be analyzed for security vulnerabilities and code design flaws. In essence TraceAI has isolated and surfaced interesting behaviors from many possible behaviors so that security analysts can investigate further. By cutting down on possible candidates, TraceAI has reduced cognitive load and investigation fatigue by many orders of magnitude. TraceAI can also detect outlier behavior to flag groups of users who are not explicitly responsible for a known attacks, but are exhibiting suspicious behavior and not being like the rest of the users. Analysts can also dig into the outlier groups shown as "blue" circles, analyze few of these users, and then as a batch put all these users under a watch list. By working at the level of a group, multiple users can be tagged/flagged /watched without having to deal with each user separately.

Summary

Traceable combines the power of Distributed Tracing and TraceAI technologies to protect next generation attacks on Web Applications and APIs. By choosing the right machine learning technology, and focusing on addressing API security issues Traceable is pushing the boundaries of API security.

We would love to talk to you further and demonstrate how our technology addresses API security issues in your environments.

See product

view demo

Meet with sales

book meeting

Recommended reads.

Introducing Traceable

Application Security for APIs and Cloud-Native Apps.

Modern Application Security – Good and Bad News

Part I: What are Modern Applications

product	personas
platform	security executives
API Protection	security operations
API Catalog	engineering executives

API Analytics

developers

why traceable

how it works

pricing

learn

resources

blog

webinars

customer stories

compare

glossary

documentation

API Security Academy

company

our story

team

press

compliance

t-shirt shop

careers

contact us

meet with traceable

- log in
- support
- status

meet with traceable