ProtectOnce

Technical Brief

The complexities of API security

Modern API-driven applications are extremely complex and ever-changing. While APIs fuel growth and development speed, they expose the business logic of the application and make it a prime asset for attackers to target. Legacy WAFs are not enough, while modern solutions are too complex and expensive to run.

This problem is especially tricky for growing SaaS companies that are often short on security personnel and resources. These startups need a one-stop-shop solution that can secure their modern, API-driven web applications without endless configuration and fine tuning, and without huge costs. In today's API security market, this is a pretty big ask.

Why ProtectOnce

ProtectOnce sets out to radically simplify API and application security with an agentless solution that takes minutes to deploy, extremely easy to use and provides deep security visibility and threat detection immediately.

The solution reveals all of your APIs (REST, SOAP, GraphQL), including zombie and shadow ones, to provide customers with a complete inventory and posture of their APIs.

ProtectOnce then applies data analysis models to combine this deep visibility and inventory with behavioral data collected over time, to detect and expose attacks on your APIs as they are planned.

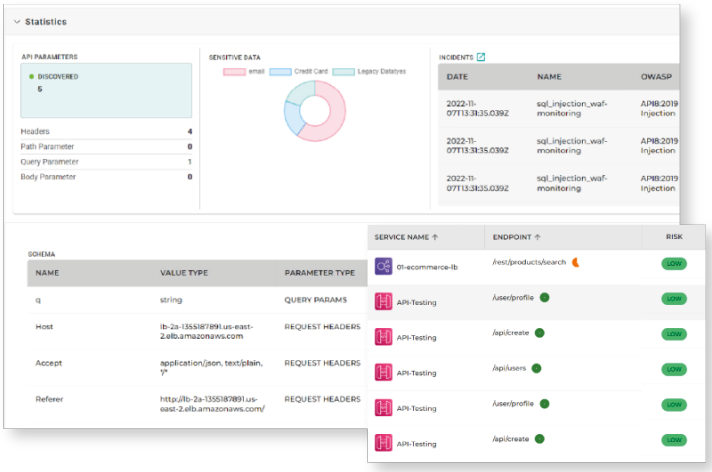
ProtectOnce allows you to expand your security to the workload level with the solution's in-app deployment. The in-app library helps customers to apply real time attack blocking only where they need it the most, without impacting their application's performance or creating overhead.

This allows you to enjoy both a no friction agentless discoverability and visibility, and a light-weight inline blocking where your crown jewels are.

Solution Highlights

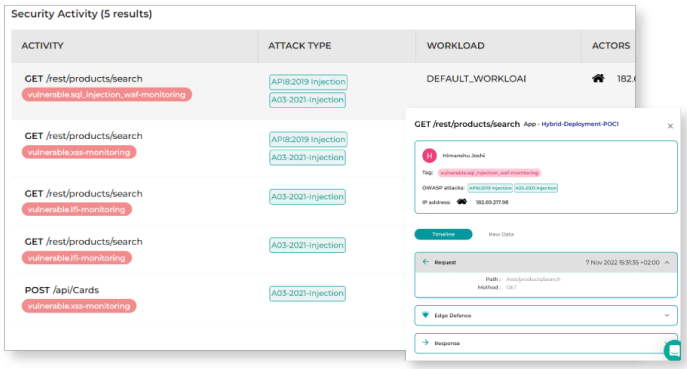
API Discovery & Posture

ProtectOnce automatically and continuously inventories your APIs, helping you keep track of all your public facing assets, detect any blindspots in your posture, and assess risks associated with your APIs.



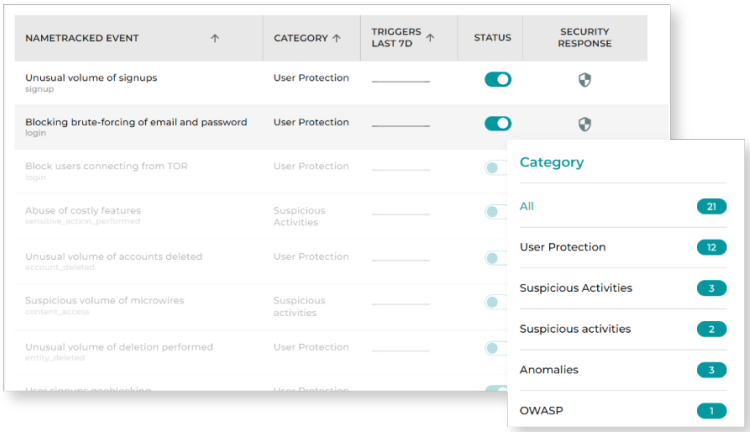
API Protection

Protect your APIs from sophisticated attacks by combining deep visibility with behavioral data and the context of your app, to detect and prevent attacks before they happen.



API Threat Response

Easily respond to security events and automatically configure compensating controls.



How it works

Instrumentation Options

ProtectOnce is available to customers through two instrumentation options:

Agentless: A zero friction connector to your cloud environment

In-app: A lightweight library that requires no fine-tuning or configuration

Agentless vs In-app tradeoffs

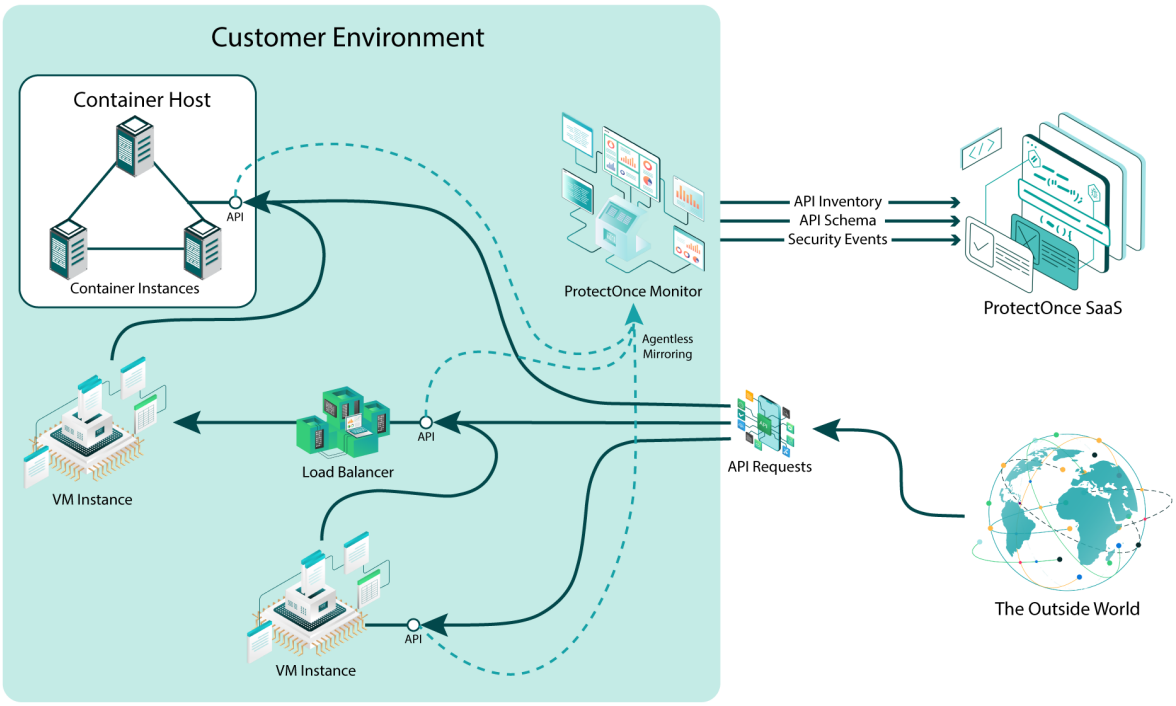
The ProtectOnce in-app and agentless deployment options come with some tradeoffs and benefits. With the in-app instrumentation ProtectOnce is able to provide capabilities which are not offered with the agentless deployments, and vice versa. These are:

	In-app	Agentless	Hybrid
API Discovery	✓	✓	✓
API Schema	✓	✓	✓
Managed API Gateways	✗	✓	✓
Vulnerability Detection	✓	Partial	✓
Attack Detection	✓	✓	✓
In-line Attack Blocking	✓	✗	✓
Application Self-Protection	✓	✗	✓
Application Flow Tracking	✓	Partial	✓

Customers can choose to work with one instrumentation model or the other to best suit their security needs and development processes.

Combining both models often provides the best security: customers can onboard environments with the agentless connector, quickly providing visibility and detection across their entire environment, and then add the in-app library to workloads with direct access to sensitive data, to allow real-time attack blocking.

How it works: Agentless



There are two types of agentless deployments - a one-click installation or a manual process.

One-click Configuration

With one-click configuration, you are redirected to the AWS cloud formation create stack page. On this page, the ProtectOnce agentless cloud formation template is already uploaded with pre-populated fields. The stack name is generated and the ProtectOnce token is pre-populated.

This token is essential for the deployed resources to communicate with the ProtectOnce backend. Once the stack is successfully created, multiple cloud resources are created including an EC2 instance which has the ProtectOnce Collector and AWS recommended Suricata installed on it. These components are responsible for storing and processing traffic mirroring data.

A traffic mirroring session is automatically created to mirror your API traffic to the ProtectOnce Collector. This session is pre-configured with the traffic source (this is the cloud resource you want to protect) and destination resource (the instance that ProtectOnce creates after successful stack creation). API traffic is mirrored with zero overhead or impact to your running application, and all API traffic remains within your account.

Manual Deployment

The manual deployment requires you to download the cloud formation template, and upload it to AWS CloudFormation to create the stack. This will launch an EC2 instance with the ProtectOnce collector and Suricata installed on it.

When the stack is created, a traffic mirroring session is also created which has the traffic source (this is the cloud resource you want to protect) and the target resource which is the instance created by ProtectOnce.

This model functions in the same way as the One-click deployment, but gives you more control over how, when and where to deploy the stack, and is suitable for more complex cloud management environments.

Resource Creation

When the AWS Cloud Formation stack is created, a SNS (Simple Notification Service) and a child stack is deployed. The child stack consists of a Cloud Inventory Management Lambda, which creates all the required cloud resources to initiate ProtectOnce's agentless security.

Resource	Resource Type	Reason
LambdaRole	AWS::IAM::Role	This role gives the lambda access to scan and create new resources.
CloudInventoryManagementLambda	AWS::Lambda::Function	This will create instances with POCollector and Suricata deployed on it, depending on the customer application's network configuration.
POCollector	AWS::EC2::Instance	The ProtectOnce agentless solution and Suricata will be deployed on this instance.
TargetENI	AWS::EC2::NetworkInterface	This is an ENI device which is connected to the instance on which the POCollector is deployed. This will receive all the mirrored traffic.

Resource	Resource Type	Reason
SGtarget	AWS::EC2::SecurityGroup	Security group attached to TargetENI in order to filter out only the mirrored traffic.
NetworkInterfaceTarget	AWS::EC2::TrafficMirrorTarget	This will make the instance with POCollector in it as a target resource for mirrored traffic.
TrafficMirrorFilter	AWS::EC2::TrafficMirrorFilter	This is a filter for traffic mirroring.
TrafficMirrorFilterRule AllOutbound	AWS::EC2::TrafficMirrorFilterRule	An egress outbound rule to allow all the TCP traffic to target the resource.
TrafficMirrorFilterRule SSHInbound1	AWS::EC2::TrafficMirrorFilterRule	An egress inbound rule which will reject all the TCP packets on the port 22 of target resource.
TrafficMirrorFilterRule SSHInbound2	AWS::EC2::TrafficMirrorFilterRule	An egress inbound rule which will reject all the TCP packets on the port 22 of source resource.
TrafficMirrorFilterRule SSHOutbound1	AWS::EC2::TrafficMirrorFilterRule	An egress outbound rule which will reject all the TCP packets on the port 22 of target resource.
TrafficMirrorFilterRule SSHOutbound2	AWS::EC2::TrafficMirrorFilterRule	An egress outbound rule which will reject all the TCP packets on the port 22 of source resource.
TrafficMirroringSession	AWS:EC2::TrafficMirrorSession	1 traffic mirroring session will be created for each instance on which POCollector is deployed. This will help mirror the traffic and route it to the target instance.

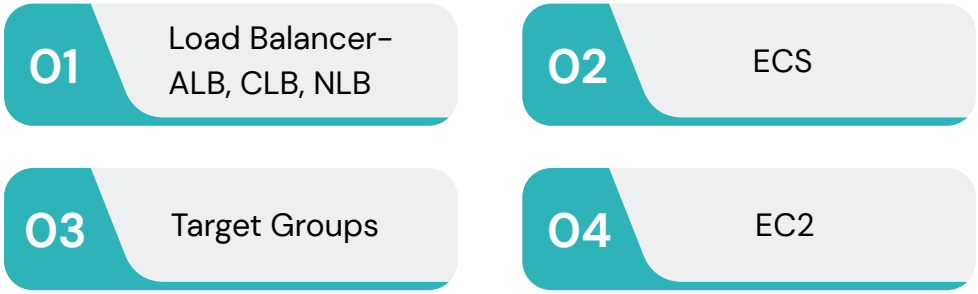
Supported Environments

ProtectOnce Agentless Mirroring currently supports the following environments:

- AWS: Full support for APIs in the AWS cloud platform.

Supported Cloud Resource Types:

For AWS environments, you can use any of the below traffic mirroring sources to configure ProtectOnce's agentless security monitoring:

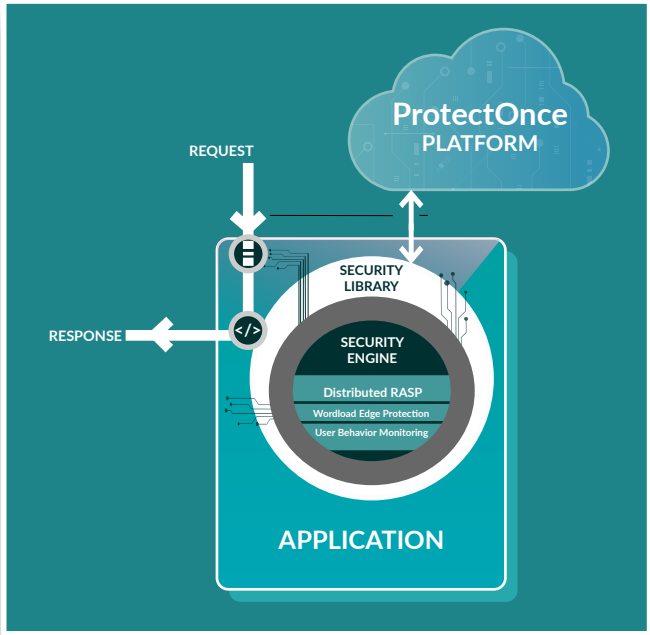


Data Privacy

ProtectOnce recognizes the sensitivity of your security metadata. Our first line of defense is ensuring that sensitive PII does not flow from your system back to the ProtectOnce cloud. Additionally, all customer data is encrypted both in-transit and at rest. All customer data is carefully tagged with ownership, and ProtectOnce implements a strict audit process to ensure that customer data is only ever accessible to an authenticated and authorized member of your team.

The ProtectOnce backend runs in secure AWS data centers, and employs state-of-the-art security defense, including ProtectOnce itself, to ensure your data is protected at all times. Critical data is backed up using native secure cloud backup mechanisms to ensure that in the event of a catastrophic failure, your data will not be lost.

How it works: In-app



Library Installation

ProtectOnce security libraries can be installed by developers with a single line of code. Simply install the protectonce package, and add a require or import in the main application module. No other manual instrumentation is necessary.

Automatic Instrumentation

The ProtectOnce library will automatically detect the environment and frameworks being used and apply transparent instrumentation of security verifications where needed. This includes communications frameworks such as express for nodejs environments or flask for python environments. The library also instruments specific APIs to allow inspection of API calls before they are executed.

Language	Supported Frameworks
NodeJS	Express 4.0 and greater, hapi 13 greater
Python	Flask 0.10 and greater
Java	Comming Soon

Language	Supported Libraries
NodeJS	Express, Hapi, SQLite3
Python	Django, Flask, SQLite3, PostgreSQL
Java	Coming Soon

Data Privacy

All security verification happens within the workloads. The security library only sends metadata to the backend. Parameter values, body values and data files are not transmitted outside the workload.

Security Library Flows

There are several key activities that the security library handles. These can be organized under the following application flows:

Flow	Security Library Actions
Startup	◉ The library collects basic information about the workload and notifies the backend that it has been launched. This information includes the application static bill-on-materials. ◉ The backend provides updates to the security configuration and rules to be implemented. ◉ The library applies all necessary instrumentation hooks to the application.
API Route Definition	◉ The library detects new API routes and reports them to the backend[1].
Module Loading	◉ The library detects the usage of a module and reports this to the backend[2] .
Incoming Requests	◉ Incoming requests are first inspected by the In-App WAF for security risks. ◉ Requests then flow as usual, with the RASP engine inspecting calls to the filesystem, network, databases, and the operating system. ◉ The library reports basic statistics for successful requests as part of the heartbeat. ◉ The library reports detailed information for any detected security risks, along with flow and stack trace information[3]. ◉ The library reports security overhead (in milliseconds) to the backend.

Flow	Security Library Actions
Outgoing Communication	◉ The library instruments outgoing requests with trace path information headers to allow downstream libraries to detect the path a request took.

1. Reports are cached and transmitted during the next heartbeat interaction with the backend. API Route reports are used for mapping and visualizing APIs.

2. Reports are cached and transmitted during the next heartbeat interaction with the backend. Module loading is tracked to provide better risk analysis for vulnerable libraries. A vulnerable library that is detected in the application BOM but is not actually loaded can be deprioritized.

3. Flows indicate how requests traveled between microservices and help developers understand the path an attack took, and where the best place to prevent it is. Stack traces provide developers with a clear indication to where in the code the risk occurred and how the request arrived at that point.

About ProtectOnce

As a group of SaaS developers and cyber security entrepreneurs, application & API security has always been a pain point.

In today's cloud-native world, developing and deploying web applications is easier than ever, however properly securing them has never been more complex.

This is why we built ProtectOnce. We believe that quality API security should be accessible to all companies, big and small, without having to invest endless resources and manpower.
With ProtectOnce, growing SaaS companies can get API visibility, discoverability and protection for free, with an automatic, out-of-the-box approach.

Start for free:

app.protectonce.com/signup

ProtectOnce