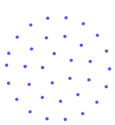
SALT

Salt Security in a DevSecOps Model

WHITE PAPER

Salt Security in a DevSecOps Model

TABLE OF CONTENTS



Securing APIs is made more difficult by the ramifications of today's DevOps processes. To help with the task of protecting API-based applications, organizations can embrace the tenets of DevSecOps to meet the goals of speed, efficiency, and security.

The following summaries capture the complications that rapid CI/CD can introduce into securing APIs as well as the ways in which the Salt Security API Protection Platform can mitigate those challenges:

Challenge	**Salt Security**
Keeping API documentation up to date at all times	Automated and continuous discovery to ensure complete API coverage and accurate documentation throughout the entire CI/CD pipeline
Ensuring APIs to be released to production meet security requirements	Ability to identify and provide detailed security insights into security gaps and risks while in the staging environment
Providing comprehensive protection of APIs in production	Automatic and comprehensive protection against all OWASP API Security Top 10 threats and other risks, leveraging our patented technology that is not dependent upon OAS, signatures, or configuration to detect and prevent API attacks
Streamlining communication between security and dev teams throughout the entire CI/CD pipeline	Ability to integrate with DevOps systems such as Jira to streamline communication about API security gaps
Integrating API security into existing automated processes and ecosystems	Using the Salt integrations to send alerts to incident response teams, block attackers using existing infrastructure, push important events to SIEMs, for example
Protecting all APIs across all environments with consistent tools and processes	Ability to protect both external and internal APIs by fitting seamlessly into every environment, including any cloud provider and on-prem environments, without dependencies on specific technology stacks
Ability to maintain compliance and privacy regulation requirements	Automated discovery of PII and sensitive data throughout the CI/CD pipeline to ensure an API change does not violate existing compliance and privacy regulation requirements, with no need for manual auditing

Overview

The goal of DevSecOps is to introduce security measures earlier in the Software Development Life Cycle (SDLC) while maintaining the high velocity of releases and rapid application innovation essential to business transformation today.

A great DevSecOps model allows fast and smooth deployment to production while meeting all security requirements, automates remediating security gaps as much as possible to ensure speed, and prevents deployment to production when security gaps exist while providing details on how to remediate the problem and complete the build.

Applying DevSecOps principles to the process of securing APIs is challenging. In the next section, we will cover these challenges and describe how Salt Security simplifies the process and helps ensure your organization will meet both its security and development goals.

DevSecOps Challenges in Securing APIs

Keeping API documentation up to date before and after deployment to production, despite frequent API changes as part of rapid CI/CD

- **Challenge**
  Accuracy of documentation such as OAS is critical to help security teams understand and assess the risk of APIs and exposure of data and ensure no unknown attack surface exists. Research conducted by Salt Security shows that a gap of 40% is common between a manually created OAS and what's really deployed, in terms of the number of APIs documented and the level of detail captured in the documentation vs. what the Salt platform reveals in the discovery phase.

- **Salt Security**
  The solution can be deployed in both staging and production environments to ensure OAS files are up to date at all times.

  In staging environments, Salt Security enables the ingestion of OAS files and provides a complete analysis of all missing elements such as missing API endpoints, missing parameters, discrepancies with parameter definition, and other factors.

  | *Note: see Appendix A for examples of this analysis.*

In production environments, Salt Security monitors all APIs in real time to detect any new APIs as well as changes to existing APIs. In the event an existing API is changed or a new API has been deployed without going through the "official" process, Salt will detect the change or addition and alert security teams. Such events often happen when urgent patches get deployed or old versions remain running alongside new versions.

Salt Security will also reduce the API attack surface by detecting any deprecated "Zombie APIs" that need to be removed from the production environment.

Ensuring that APIs about to be released to production meet security requirements

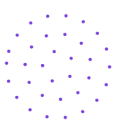
- **Challenge**
  To prevent the release of vulnerable APIs, organizations must analyze APIs deployed to staging and detect any security risks or gaps that need to be addressed prior to deploying to production. Many of these security risks and gaps require deep analysis of API traffic including how the APIs were implemented. Manual review is both time consuming and error prone and slows down the CI/CD process for every release version.

  Organizations need the ability to detect security risks and gaps (e.g., never expired tokens, overly detailed errors, exposure of sensitive data) that cannot be detected by OAS-based analysis or static analysis tools.

- **Salt Security**
  Security Posture Insights tickets are generated in staging environments for any potential risk or issue based on analyzing legitimate test traffic. Our big data and AI functions as the "API security expert" in staging, to ensure vulnerabilities in APIs are identified and remediated before deploying to production.

  Security Posture Insights include new sensitive data exposed in URLs, new endpoints exposing sensitive data, detailed errors that share sensitive information, never expired tokens, and more.

  | *example: a JWT or Authorization token is returned in a response where it should not be returned*



Comprehensive API Protection in production despite frequent API changes as part of rapid CI/CD

- **Challenge**
  Constant changes in APIs and applications make it difficult to provide accurate and comprehensive protection. As APIs change at a rapid pace, security tools can retain an "old" view of the API and gaps in protection, which can lead to either many false positives or inadequate protection. Any security solution that requires manual human verification of OAS files before and after deployment to production will not work at the speed required by DevSecOps.

  Security tools must continuously have an up-to-date view of APIs to provide proper protection.

- **Salt Security**
  Many customers deploy new API versions daily or weekly as part of their CI/CD process. The Salt Security solution automatically adapts its baseline of legitimate normal behavior for each new API and classifies any deviation or anomaly into clear attack cases to allow automated blocking of attackers early during their process of reconnaissance.

  Salt Security is the only company with patented technology to detect and prevent API attacks using Big Data and AI. With Salt Security, organizations avoid the need for any manual configuration of the security platform – instead, it leverages AI to automatically distinguish between legitimate frequent API changes and real malicious anomalies to ensure comprehensive API protection at all times.

Streamlined communications between security and development regarding API vulnerabilities in production

- **Challenge**
  Identifying and remediating vulnerabilities efficiently in production is just as critical as doing so while APIs are still in staging – in fact, one could argue it's more critical to ensure their security in production environments because that's when attackers can exploit their vulnerabilities. After a vulnerability is found, tickets must be routed to the appropriate team or developer with the right level of detailed insights to establish the proper priority and ensure quick remediation.

- **Salt Security**
  Use attacker activity as penetration testing efforts that yield insights into where vulnerabilities exist in APIs. After the Salt Security platform blocks an attacker, it sends insights learned from reconnaissance activity through ticketing systems, such as Jira or ServiceNow, to the appropriate development team for remediation. Details include where in the API a vulnerability exists, what normal behavior is for the API, how the attacker tried to manipulate the API, and how the application responded. These insights provide needed details to help development teams prioritize and efficiently eliminate vulnerabilities in the API.

Integrating API security into existing automated processes and ecosystem

- **Challenge**
  Organizations must balance the competing priorities of quick and efficient development and delivery of code with the need to release secure applications to production. Fitting into existing application delivery tools and workflows is essential to ensure that goals are met for both security and development teams without impacting the rate of application delivery.

  To fit into the DevOps workflows, security systems must integrate with existing tools such as ticketing systems and SIEMs.

- **Salt Security**
  A rich and flexible Integration Engine allows exporting any data or event to any other solution using webhooks and APIs. Organizations can use this integration to:

  - Block attackers with API gateways, load balancers, or other devices
  - Send alerts to any internal systems such as Splunk, QRadar, or Elastic
  - Report security insights to internal systems including PagerDuty, ServiceNow, Zendesk, or Splunk
  - Report on API vulnerabilities to any internal ticketing system such as Jira or ServiceNow
  - Report API discovery events such as new APIs, new PII exposure, and API changes to any internal systems

Protecting both external and internal APIs across environments and development frameworks without sacrificing speed or protection

- **Challenge**
  Development teams today run multiple development frameworks and a variety of tech stacks. It's not practical for organizations to deploy different security tools across all of those environments. Security solutions must straddle those different environments effectively and must not have dependencies on specific development frameworks to operate. They must also be able to protect legacy, current, and future APIs.

- **Salt Security**
  The platform's flexible architecture can collect data from any environment (cloud and on-premises) with a broad set of integrations. The Salt Security solution does not require any changes to application code and needs only a copy of API traffic, which ensures the solution is completely agnostic to software frameworks (e.g., different technologies and development kits). In addition, the solution works with APIs that do not have or can-not support OAS files such as non-RESTful APIs, SOAP APIs, traditional APIs (e.g., URL-Encoded based), and legacy applications with no proper documentation, for example.

  As a result, organizations can apply the Salt Security platform to any environment and to any API.

Maintaining compliance and privacy regulations (GDPR, CCPA) despite frequent API changes as part of rapid CI/CD

- **Challenge**
  Maintaining compliance and data privacy (e.g., GDPR, CCPA) is challenging in CI/CD environments, because every API deployment, update, or change can require a change in the organization's privacy policy. For example, a developer can add a parameter of a user's private email address in a version release without the compliance team knowing it.

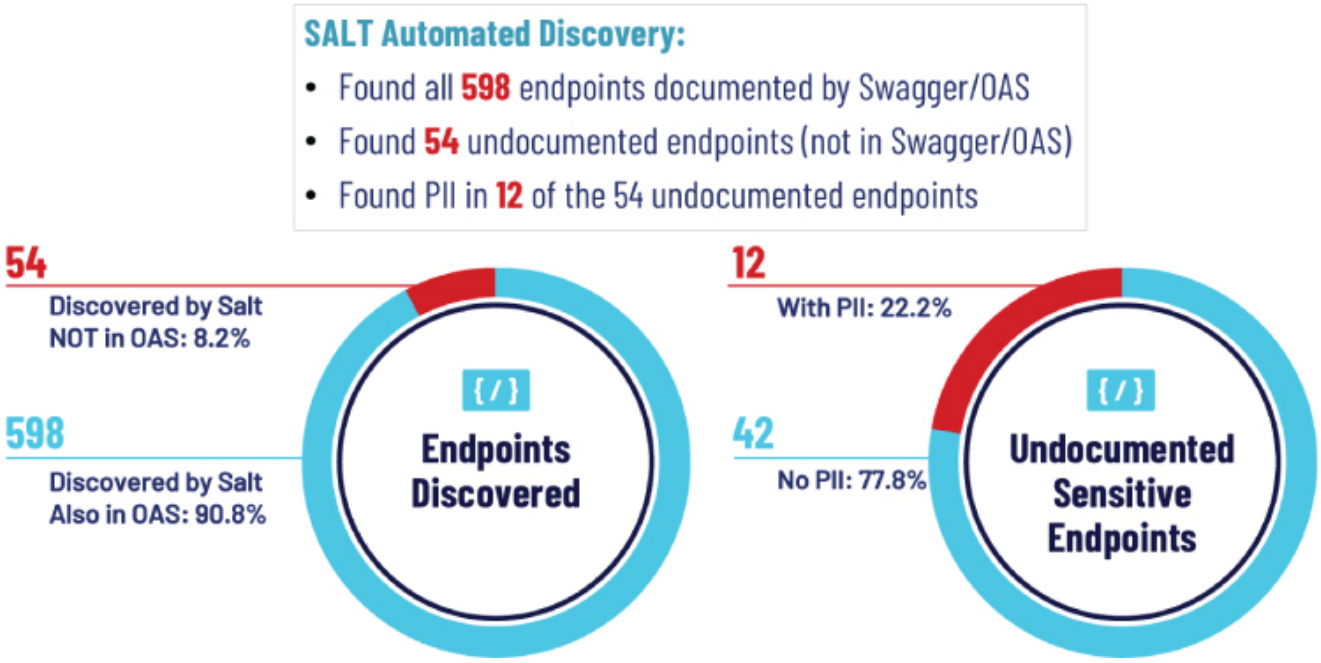
  Compliance and privacy regulations demand that organizations map all sensitive data (e.g., private data or PII) within the environment and deploy proper controls to delete data as necessary. In addition, any sensitive data organizations collect or share with third-party vendors should be part of the privacy documentation.

  Organizations need a mechanism to detect any changes or new sensitive data and PII exposure as early as staging, before any API is deployed to production.

In addition, organizations also need the ability to monitor for new sensitive data in production.

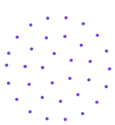
- **Salt Security**
  All sensitive data and PII is automatically mapped to enable detection of any new sensitive data being introduced in environments, from staging to production. This approach ensures and validates that API changes are not collecting new sensitive data from consumers, exposing new information, or sharing sensitive data with third-party vendors without the security team's full awareness.

  If no new sensitive data or PII exposure is detected, the Salt Security platform will enable the release to progress to production, keeping the deployment process smooth while also keeping legal teams and auditors happy.



Summary

The significant increase in the use of APIs, along with the special risks they pose, create new challenges for security teams, challenges compounded by the speed of DevOps practices. By embracing the tenets of DevSecOps and integrating security throughout all steps of the API lifecycle, organizations meet security requirements without hindering the speed and efficiency of application development that DevOps enables. Visit https://salt.security/demo/ to see how the Salt Security API Protection Platform can help you innovate faster with APIs while still protecting your data and services.

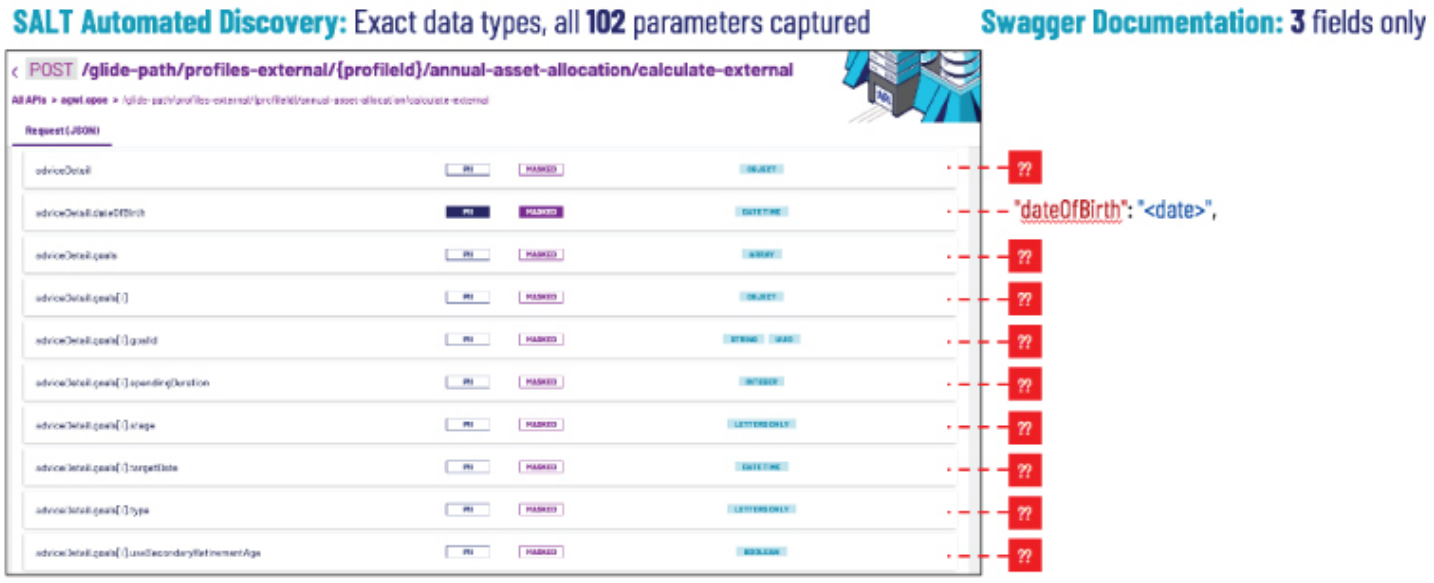


Appendix A

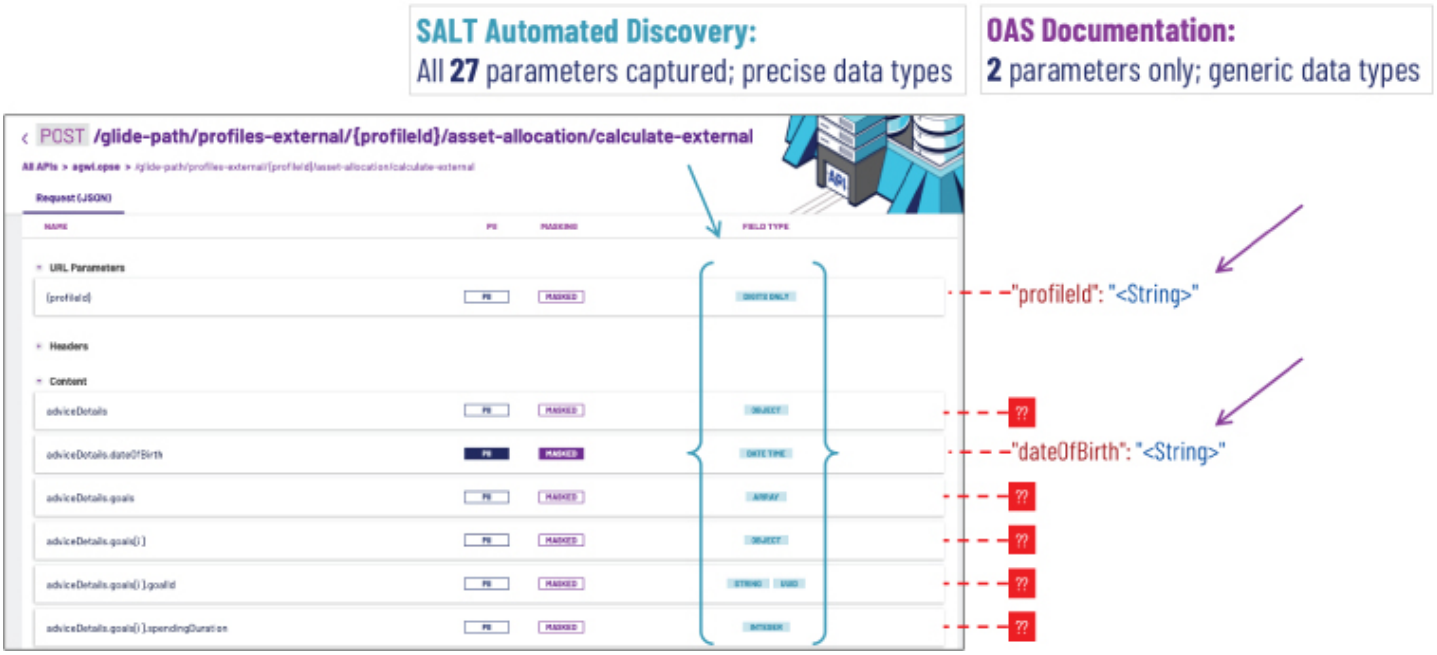
Accuracy of documentation such as OAS is critical to help security teams understand and assess the risk of APIs and exposure of data. Accurate documentation also helps to ensure there are no shadow APIs, introducing unrealized attack surface. Research conducted by Salt Security shows a common gap of up to 40% between the manually created OAS vs. what's actually deployed in the APIs. These gaps fall into the following three categories:

1. **Shadow API Endpoints** – API endpoints that are missing from the OAS. In the following example, Salt Security research found an additional 54 endpoints that were not included in the Swager/OAS documentation, and 12 of those undocumented endpoints were exposing sensitive PII data.

2. **Shadow Parameters** – API endpoints known to exist but whose documentation is missing many parameters. As a result, the documentation does not cover the majority of the attack surface – in this research, documentation listed three parameters, but the Salt Security platform identified 102 parameters.



3. **Parameter Definition Discrepancies** – in addition to many missing parameters, data types that lack needed details such as “String” instead of “UUID” or “DateTime” will make APIs vulnerable, since any input will be processed.

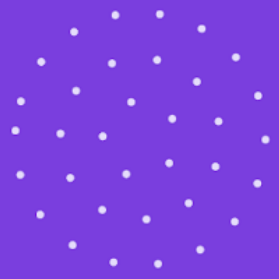


Salt Security – Securing your innovation

Salt Security protects the APIs that form the core of every modern application. Its patented API Protection Platform is the only API security solution that combines the power of cloud-scale big data and time-tested ML/AI to detect and prevent API attacks. By correlating activities across millions of APIs and users over time, Salt delivers deep context with real-time analysis and continuous insights for API discovery, attack prevention, and shift-left practices. Deployed in minutes and seamlessly integrated within existing systems, the Salt platform gives customers immediate value and protection, so they can innovate with confidence and accelerate their digital transformation initiatives.

Request a Demo today!
info@salt.security
www.salt.security

WP-243-092622



SALT

Securing your Innovation.