The Evolution of Enterprise Application Security

Why enterprises need runtime application self-protection
Abstract

Enterprise information security encompasses a broad set of disciplines and technologies, but at the highest level it can be broken down into three main categories: network security, endpoint security and application security. Network security and endpoint security have advanced greatly in the last few years and enterprises have invested appropriately.

Hackers, meanwhile, have switched their focus to a softer target: applications. Enterprise data (and most importantly their customer information) is the pot of gold at the end of the rainbow. And applications are the way in. This is why Gartner estimates that more than 70% of all hacks happened at the application layer in 2013 – making it the new attack surface of choice.

The industry initially responded with the introduction of web application firewalls (WAFs) in the network, but this approach proved to be limited at best. It’s becoming clear that the best practice for organizations is to find ways to build security directly into the applications themselves, which explains the recent boom in application security testing tools to identify vulnerabilities. However, these tools don’t actually fix anything, so many challenges remain -- not the least of which are the significant backlogs of known vulnerabilities in running production applications.

It’s time for applications to protect themselves.

This paper outlines why Runtime Application Self Protection (RASP) is an important emerging application security category and how Prevoty’s RASP technology allows robust security to be easily built into applications. It also discusses how Prevoty’s new approach solves organizations’ security challenges across their entire application portfolio:

Past: Instant remediation of existing application vulnerability backlogs

Present: Quicker time to market for applications without introducing new vulnerabilities

Future: Protection from zero-day attacks
Hackers’ new favorite target: the application

For most organizations, applications are at the heart of their business today and application downtime due to a breach has a very direct impact on a business. More importantly, the customer data processed by these applications is an organization’s lifeblood. Ultimately, customer data and intellectual property is precisely what the hackers are after, however they choose to launch their attacks.

Modern network security and endpoint security have made notable strides in recent years. “Why do things the hard way if there’s an easier alternative?” Application security has simply not kept pace and sophisticated hackers have found that it is almost always easier to exploit application vulnerabilities. With enterprise data as the prize, it’s no wonder that the majority of hacks now take place at the application layer.

First response: the WAF

The industry developed what was essentially an extension of the network security model - web application firewalls (WAFs) - as a first attempt at defense. WAF’s see all network traffic destined for the application and so have the ability to filter traffic based on a set of rules and can be effective in preventing certain types of attacks, such as DDoS, directory traversal attacks, etc.

However, since WAFs sit at the perimeter they have no context about the applications themselves, and, without the combination of content and context, they have proven to be ineffective at thwarting application security attacks.

In a recent Gartner research paper¹, Joseph Feiman, Gartner Fellow and Vice President, eloquently articulates the problem:

"Infrastructure and perimeter protection technologies inherently lack insight into application logic and configuration, event and data flow, executed instructions and data processing. Thus, they lack the necessary means to ensure accurate detection of application vulnerabilities and protection against application-level attacks. Perimeter protection technologies cannot protect against behind-the-perimeter insider attacks, which are as devastating as outsider attacks.

1. Source: Gartner - “Maverick Research, Stop Protecting Your Apps; It’s Time for Apps to Protect Themselves”, published September 25th, 2014
The Evolution of Enterprise Application Security

We used to think of an application as a binary or single host deployment accessed by PCs (or terminals!) that always reached the server-side of the application through a firewall. This simplistic model has evolved to something much larger and much more complex. Applications today are dynamic and distributed, using cloud computing and storage, partner applications, RSS feeds, and API calls to external services.

Perimeter protection technologies cannot protect against behind-the-perimeter insider attacks, which are as devastating as outsider attacks.

Perimeter protection technologies cannot protect what ceases to exist — the perimeter, which dissipates in the mobile, consumer-oriented and cloud-oriented world.”
So where exactly is the perimeter? There are too many paths into a modern application that we can no longer accept the anachronistic notion that perimeter security is sufficient.

And what of the traffic that does go through the traditional perimeter? Most WAFs claim “active prevention” or “active defense” capabilities -- not just detecting threats but actively rejecting the users, inputs or queries that may be harmful to the application before they get there. Let’s consider this.

Firstly, active prevention performed outside the context of the application is essentially guesswork. When an application receives input, it transforms and manipulates the data within the context of application logic. No external environment can possibly understand what the application is going to do with that data.

Sophisticated hackers construct attacks that pass cleanly through perimeter defenses and become malicious only when reconstructed by the application. Therefore, a WAF’s “active prevention” function unknowingly rejects genuine users, content, and queries -- resulting in a spate of errors commonly known as “false positives”.

WAFs also rely on past definitions from known hacks, meaning that they can offer no protection against previously unknown threats, a.k.a. zero-day attacks.

The high false positive rate and consequent lack of effectiveness against sophisticated
attacks has led many enterprises to turn off “Active Mode” in their WAFs entirely. Others resort to lowering the acceptance thresholds in their WAF configuration, exposing the application to more attacks.

To sum up, WAF’s can help an enterprise significantly with compliance and in preventing certain types of attacks. But they are not able to offer true application security.

So if an enterprise cannot trust perimeter security to stop application layer attacks, what other options does it have?

Security simply must be built into the applications themselves.

Next response: Secure software development

Building security into applications is not easy. By and large, enterprises adopt a few relatively consistent approaches:

- Make security a key part of the application development process itself, not an afterthought
- Check applications for vulnerabilities using Static (SAST), Dynamic (DAST), Interactive (IAST) application security testing tools and penetration tests before remediation
- Employ secure coding techniques and best practices whether fixing existing applications or releasing new applications
The above steps are absolutely appropriate and necessary, but many hurdles remain:

- A full Secure Software Development Life Cycle (SDLC) is expensive and time-consuming. Significant expertise is needed to ensure that processes are implemented correctly and consistently but today, less than 5% of security staff are application security specialists.¹

- Developers are prized resources, focused on releasing new applications and features to support the business. Security and legacy application remediation are rarely priorities – nor are they indicators used to measure job performance or effectiveness.

- Typically, application developers are not hired for their security expertise. Preventing sophisticated attacks requires a deep knowledge of the arcane world of security. With agile development, shorter release cycles and the liberal use of open source libraries, there are still vast opportunities to release vulnerable code.

- Application Security Testing (AST) tools are valuable, but have limitations in their ability to accurately diagnose vulnerabilities and can also suffer from too many false positives. AST and penetration testing also rely on known hacks, so there is still no protection against zero-day attacks.

- For organizations with large application portfolios, it is problematic to scale the testing and review process, and almost impossible to consider full-scale remediation.

Once again, in the words of Gartner analyst Joseph Feiman¹:

“Technologies and services that we use to test and diagnose our applications for security vulnerabilities fail to scale to test all applications and to test them with the necessary accuracy. There are too many apps, testing skills are scarce, and tools are too complex and inaccurate.”

¹ Source: Gartner - “Maverick Research, Stop Protecting Your Apps; It’s Time for Apps to Protect Themselves”, published September 25th, 2014
Introducing Prevoty Application Security

Prevoty has developed a new approach to application security that allows robust monitoring and protection to be quickly and easily built into applications without requiring security expertise from application developers.
Advanced Monitoring & Protection

All real-time analysis and prevention is performed by the Prevoty Security Engine. Our approach is not predicated on past definitions and signatures, regular expressions, taint analysis or learning. Instead, we are focused on understanding and preventing malicious intent.

The security engine understands syntax as well as context - e.g. cross-site scripting (XSS) executes in a browser, SQL injection happens in the database engine, etc. For effective security, a combination of language analysis and context is paramount.

By strictly enforcing content and language, applications are not subject to zero-day attacks in the same way that either a traditional regex/signature based or a learning-based behavioral analysis are vulnerable.

When something malicious is identified, the security engine alerts you via the Prevoty console and/or via a SIEM or logging tool. In protection mode, the Prevoty security engine always returns secured (cleaned, non-malicious) payloads to the application, so the application will always run successfully -- even when under attack.

Implementation

Self-protecting applications call the Prevoty Engine via the Prevoty API. This function is instrumented inside the application itself in one of two ways:

- **Prevoty Plug-ins**: Installed via a single command script, Prevoty Plug-ins provide a mechanism for existing applications to call the Prevoty Engine without requiring any changes to the applications themselves.

- **Prevoty SDKs**: Prevoty SDKs are available for all common languages, including Java, C#, PHP, Ruby, Python, and Node.js. The SDKs allow developers to quickly add Prevoty security to their applications, with a single API call made whenever analysis is required (e.g. GET and POST requests).
High Performance

The Prevoty Engine is developed with performance in mind. It is not a proxy; it does not intercept every request or every GET or POST request parameter – calls to the engine are only made when analysis is required.

Requests process at sub-millisecond speeds. The only performance impact is the cost of an API call whenever a payload is being processed. With Prevoty’s cloud-based service, round-trips typically take around 60ms. When deployed on-premise, Prevoty is positioned either at the app server tier or the web server tier – whichever makes sense for the enterprise’s topology. This configuration typically results in round-trips of only 2-3ms.

Application Security Monitoring

The Prevoty security engine alerts security teams in real-time. Prevoty comes with its own Console but can also deliver real-time updates to SIEMs and log management tools such as IBM QRadar, HP ArcSight and Splunk.

Additionally, Prevoty can integrate with syslog or other logging tools. The payloads contain all the relevant KPIs and analytics on a transactional level. In this way, Prevoty can also help companies that do root cause analysis (RCA).

Runtime Protection

Real-time protection capabilities include data exfiltration prevention, content sanitization, token & session ID management, and Man-in-the-Middle (MitM) protection.

Security policy management is centrally managed across all applications, regardless of application type.
Developer Security Toolkit

In addition to monitoring and protection functions, Prevoty SDKs include additional security services for developers, including:

- Standardized cryptography
- Typed input validation
- Data leakage detection
- Spam and profanity filters

Prevoy: Application security evolved

Using Prevoty’s technology solves issues related to WAFs and the limitations of secure software development alone:

Dynamic, distributed applications

The in-app, contextual approach means that no matter where the content, queries or users originate (the cloud, web services and API calls, RSS feeds, user generated content, etc.) Prevoty’s Engine will always be able to process the inputs and state changes, protecting even the most complex applications.

Application threat visibility

Prevoy’s threat intelligence allows security and application development teams to understand what is really going on with their application portfolio; in particular, allowing the identification of which applications are actually seeing attacks.

This allows the prioritization of valuable resources for remediation or an assessment that neutralizing vulnerabilities with Prevoty can enable the focus to be purely on releasing new, secure applications.
Active Prevention

The Prevoty security engine looks at content the same way any browser will render content, handles queries in the same way your database will execute queries, and manages tokens to reflect appropriate state changes in user sessions.

By determining actual behavior, the security engine understands whether or not something is malicious or forbidden. Since these processes occur a split second before actual execution in the application, malicious or forbidden intent is rejected or removed before rendering the application safe.

Secure SDLC

With centralized control of policy via the Prevoty Console and a single engine providing protection for all applications, using Prevoty makes the process of releasing secure code far less open to individual interpretations of security best practices by developers, increasing the efficacy of the SSDLC.

Including the Prevoty SDK in an application is simple. Calls to the Engine take literally one line of code wherever analysis is required in the business logic. Do this once and there will be no need to come back and remediate the application again.

Developers can focus on new application and feature development. Security teams can focus on the many other security challenges facing the enterprise.

Internal and external resource utilization

With Prevoty enabling more robust security to be built into applications across the enterprise with centralized policy management and analytics, less investment is required in expensive internal and external resources.
| **Existing backlogs of vulnerabilities** | Prevoty Plug-ins for Java and .NET applications can be applied instantly to deliver an immediate, dramatic reduction in vulnerabilities without the need to change or recompile any of the applications. For more granularity and for applications written in other languages, Prevoty SDKs allow existing applications to be quickly and easily instrumented. |
| **Vulnerabilities introduced in new application development** | By integrating Prevoty in the application development process, developers will automatically release highly secure code without introducing new vulnerabilities. |
| **Dealing with zero-day attacks** | Because the Engine uses contextual analysis -- not past definitions or learning -- the application protects itself against new forms of XSS, SQLi, and CSRF: it does not matter whether a threat matches a previously known attack. |
| **Application developers are not security experts** | For new application development, all a developer needs to know is how to call a REST API and the Prevoty security engine will automatically include the security required to protect the application. |
Summary

Application security has evolved. With Prevoty, it is now possible for applications to monitor and protect themselves. Prevoty’s comprehensive runtime application security solution helps enterprises trust that their applications in production will alert and self-protect against the top security threats. Additionally, newly released applications leveraging Prevoty SDK’s will not introduce new vulnerabilities and expose the organization to attack. This dual capability increases the overall effectiveness of an organization's security across the entire application portfolio:
Past: Instant remediation of existing application vulnerability backlogs
Present: Quicker time to market for applications without introducing new vulnerabilities
Future: Protection from zero-day attacks

To request a live demo, start a free trial, or simply get more information, please visit prevoty.com