THE MAIN APPLICATION SECURITY TECHNOLOGIES TO ADOPT IN 2021

WhiteSource

Application Layer Continues to Be Most Attacked

2020 was a difficult year. COVID-19 and the resulting lockdowns and quarantines sent tens of millions of global workers home. As a result of this dramatic increase in remote work, the number of ransomware, phishing attacks, and accidental breaches by employees working at home rose sharply. Despite the increases in these exploits, however, the application layer continues to be the most attacked.

Why is the application layer the most vulnerable? Applications constitute the largest attack surface of all the layers in the enterprise stack. In addition, they are the hardest to defend because they are the most accessible and most exposed to the outside world. Finally, the amount of data that passes through the application layer makes it highly attractive to malicious actors.

If you look at the application security market, you can see its growth reflects the heightened importance of application security. According to Forrester, the global application security market was valued at approximately USD 4.5 billion in 2020, but is expected to grow to more than USD 7.1 billion by 2023.

One outcome of the 2020 global pandemic is that organizations have been forced to reassess their security strategies and infrastructure. In today's incredibly diverse software ecosystem, the challenge becomes protecting all your endpoints when they can be anywhere, including on devices you don't control. Think anything in the cloud. As digital transformation accelerates, organizations are moving more and more systems to the cloud, which, again, requires a reimagining of the security paradigm.

Traditional approaches to security are being challenged, but you don't need to be left exposed. More than ever, security tools are focusing on both guiding development so developers create more secure code and giving advice on how to remediate vulnerabilities. In essence, what we're seeing is security shifting both left and right so that the entire development process is fully merged with security.

In This White Paper

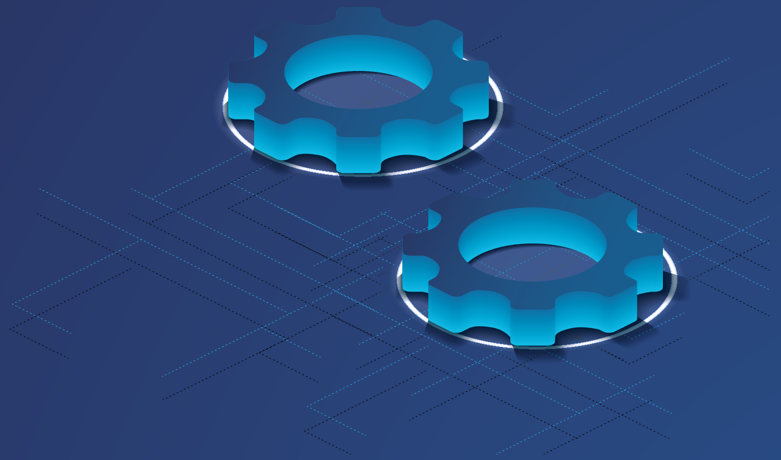
This white paper presents three trending application security technologies that are important to implement in the next year to keep organizations' application security posture up to date and resistant to modern threats. It also highlights best practices when implementing each technology so organizations can plan their application security strategy.

Top Three Application Security Technologies to Adopt in 2021

Container Security

Software Composition Analysis

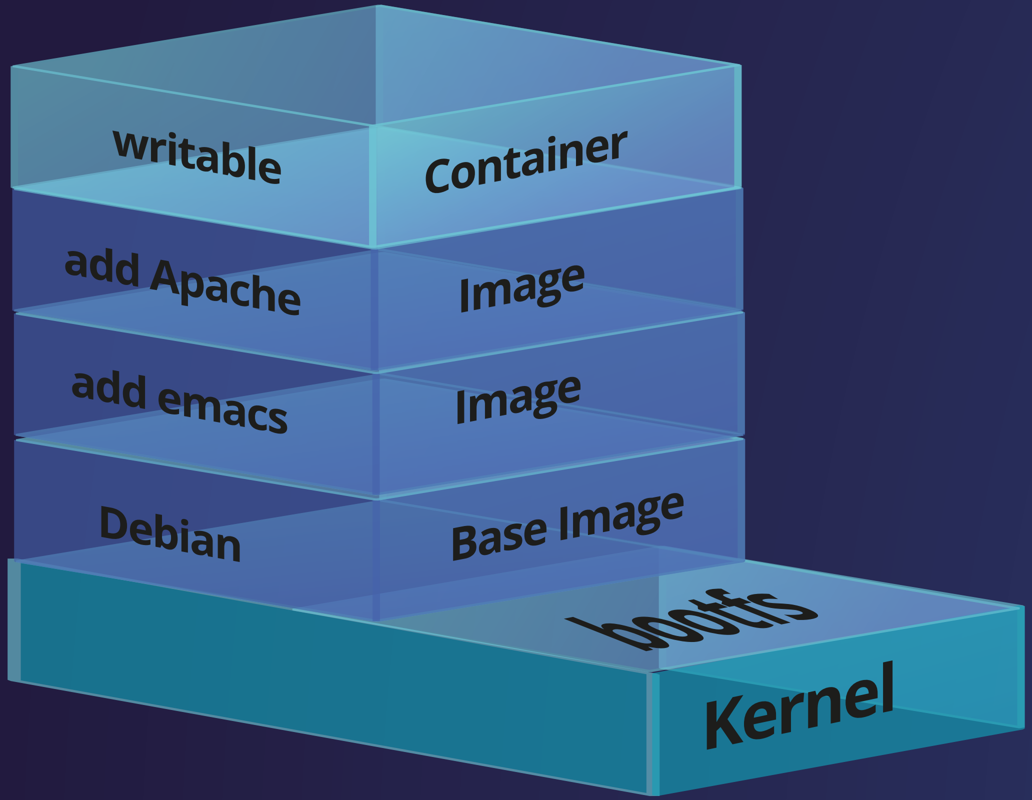
API Security



These three technologies were chosen because, although they represent maturing security market segments, they are not yet fully mature. Organizations would be wise in investing in container, open source, and API security because these technologies represent popular attack surfaces. In addition, all three technologies benefit from both shifting security left – placing more responsibility in developers' hands – and shifting security right – securing the application throughout the software development life cycle (SDLC) including in production environments.

Container Security

Container technology has been widely embraced because it makes building and deploying cloud native applications simpler than ever. Containers offer a logical packaging mechanism in which applications are decoupled from the operating system on which they run, which allows containers to be deployed easily and consistently. Containers are lightweight, fast, and can run virtually anywhere. The rise of containers has created a new application development methodology, one that enables microservices architectures and continuous development and delivery.



Why You Need Container Security

Despite the promise of less overhead in terms of system resources and greater portability and efficiency, containers present unique security challenges. Microservices increase data traffic as well as network and access control complexity. Monitoring containers, particularly during runtime, can be extremely difficult. A single compromised container can all too easily lead to other containers being compromised. Containers are built from a base image, and knowing whether to trust the source of a public image can be difficult.

Further complicating security, containers are built in layers and therefore need multi-level security. Not only does the application need to be thoroughly tested before the base image is built, but you need to secure container deployment environments and infrastructure, including hosts, runtimes, registries, and orchestration platforms.

Finally, integrating containers with enterprise security tools that meet or exceed existing organizational security policies is not always straightforward.

Container Security Best Practices

In a container environment, defects in the application image are perhaps the biggest security risk, though orchestration, data storage, and monitoring also present security complexities. In this section, we focus more on the container itself and less on its environment, though both are important to secure.

1. Reduce Your Attack Surface by Keeping Containers Lightweight

Containers were designed to be both lightweight and short-lived to run more efficiently. When vulnerabilities are found, container images are usually replaced, not patched or updated, which makes them inherently more secure. Many developers, however, don't use them this way. Too many developers add files to containers instead of updating the base image.

Once a vulnerability is identified in an image, it is best to remediate the vulnerability, then deploy new, clean containers. Always remember to minimize the number of files stored in a container and replace the containers you use frequently. Using a minimal base image and reducing the number of container components helps you shrink attack surfaces and thus prevent security breaches.

2. Use Images Only from Trusted Sources

Containers are built from images that are stored in one or more repositories, which could belong to a public registry such as DockerHub, Azure Container Registry, and Amazon Elastic Container Registry, or to a private registry like Docker Trusted Registry. Simply because a container image is publicly available does not mean that it is secure.

This should go without saying, but if you don't know the provenance of a container image, you shouldn't be using it. The container image might be poorly configured and inadvertently contain security vulnerabilities. Worse yet, the image may have been created by a malicious actor and intentionally contain malware that opens a backdoor into your sensitive data. In either scenario, you're exposed.

Luckily, there's an easy way to reduce the threat of attacks: Use container images from trusted sources only, and sign images from public registries. If you're building your own image, make sure you're scanning each layer for vulnerabilities across the container life cycle, using tools like software composition analysis for open source components, then store your image in a private registry.

3. Never Run as Root

Perhaps this is obvious, but it bears repeating: Always protect the host by properly configuring your container. Never run as root – or any other highly privileged account! If your container is compromised, the host will be exposed. Operate under the principle of least privileged user.

Software Composition Analysis (SCA)

Software composition analysis is the segment of the application security testing market that deals with the management of open source software components. SCA solutions provide a complete inventory of an organization's open source components, manage open source licenses, and identify and remediate open source security vulnerabilities. They also give organizations the power to set organization-wide policies to enforce the usage of open source components. SCA solutions often integrate with developer tools throughout the software development life cycle (SDLC), including IDEs, repositories, package managers, build tools, CI servers, and more.

Why You Need Software Composition Analysis

Software composition analysis solutions are becoming a baseline requirement for any organization concerned about application security. Just look at the math. The application layer continues to be the most attacked surface, and open source components comprise 60-80% of organizations' code base. This adds up to open source components representing a significant portion of an application's overall risk. If these components aren't being managed, then it follows that an organization's code base isn't being secured.

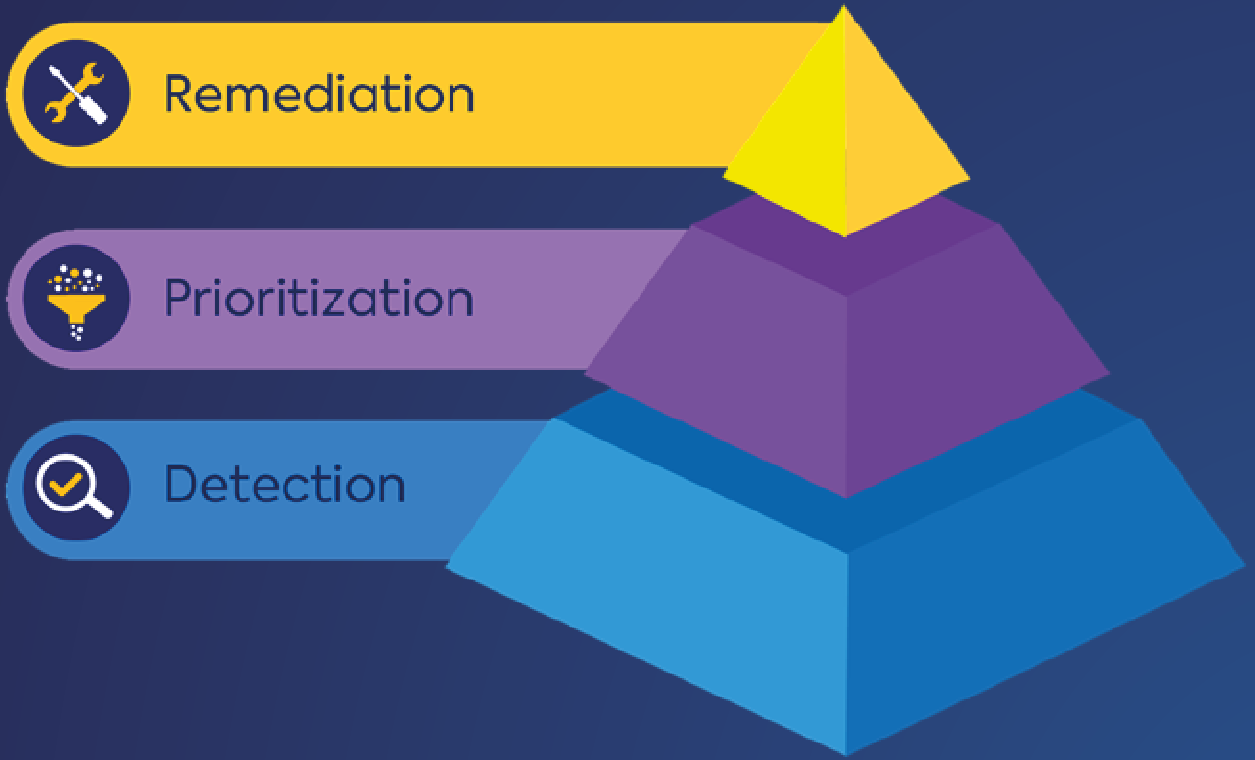
SCA solutions are more important than ever especially considering the increasing amount of government regulations like the European Union's General Data Protection Regulation (GDPR), the California Privacy Rights Act (CPRA), and HIPAA. These regulations tightly control consumer information and privacy, and organizations are assessed fines if consumer data is breached. By adopting secure coding best practices such as using an SCA solution that integrates with your IDE, repo, or CI server, you can prevent security vulnerabilities from entering your code, thus thwarting potential cyber attacks.

SCA Security Best Practices

As the market matures, best-of-breed SCA solutions will focus more on the prevention and resolution of security vulnerabilities than simply detecting issues. In 2021, look for SCA solutions that are able to prioritize security vulnerabilities with a high degree of accuracy, incorporate auto remediation of vulnerabilities, and scale with your enterprise.

1. Prioritization

For years, managing open source risk focused primarily on the detection of vulnerabilities and license compliance issues. Due to the overwhelming number of alerts and the problem of alert fatigue, identifying problems is no longer enough. In the words of Gartner analyst Neil MacDonald, "Perfect security is impossible. Zero risk is impossible." The focus of application security, therefore, must shift toward curing the most significant defects first. This is done through the prioritization of vulnerabilities.



Organizations must adopt a mature SCA security model that includes prioritization on top of detection so developers and security professionals can focus first on the vulnerabilities that represent the greatest potential risk. By using a solution that automatically identifies these most significant security vulnerabilities, organizations are able to address their highest priorities first. Developers and security professionals don't waste their time and resources sifting through pages of alerts trying to determine what vulnerabilities are the most important, possibly leaving highly exploitable vulnerabilities running in a production system.

2. Auto Remediation

The auto remediation of vulnerabilities is the next logical step after the prioritization of vulnerabilities. It goes beyond just showing developers where the vulnerability is located to actually suggesting a fix and providing data on how likely the fix will impact a build. Automated remediation workflows can be initiated based on security vulnerability policies triggered by vulnerability detection, vulnerability severity, CVSS score, or when a new version is released. One of the most reliable risk mitigation strategies is to keep your open source components continuously patched to avoid being exposed to known vulnerabilities. A mature SCA solution with auto remediation helps you achieve this.

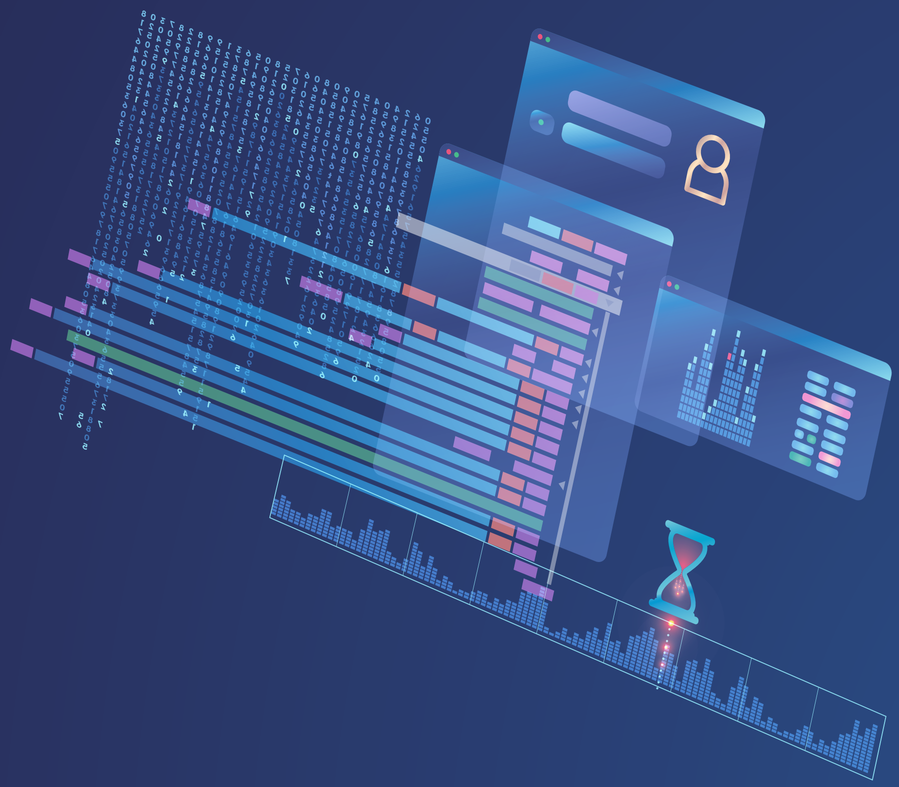
3. Delivery at Scale

The explosive growth of open source components in commercial software is due to the speed it brings to the development process. In the era of DevOps, speed – along with automation and integration – is king. Traditional application security has not been able to keep up the same pace as the current cycle of software development and delivery. So how do you deliver a secure product at scale?

For software composition analysis, this means selecting an enterprise-grade solution that targets every team in the security process. In addition to traditional security professionals, you also need to bring developers, DevOps, legal, compliance, management, and others into the fray. Security must be integrated throughout the entire SDLC and done so seamlessly through the tools these teams are already using. The only way to accelerate secure product at scale is by distributing security throughout the development process, in a sense, shifting security both left to development and right through delivery to cover the entire spectrum of your product's life cycle.

If everyone now owns security, it's vital that everyone is seeing the same snapshot of your current security posture. As security moves to the forefront, you need a way to measure the progress of your security posture against industry benchmarks. By default, an enterprise-grade solution must have robust, real-time reporting capabilities tailored to each individual team's specific needs.

You've made an investment in security. You want to know how that investment is performing and what you're getting out of it using measurable and actionable insights. Make sure you choose a solution that can deliver.



The Very Real Problem of Alert Fatigue

What is alert fatigue?

Alert fatigue is when an overwhelming number of alerts desensitizes those responsible for monitoring them. This desensitization leads to either missed or ignored alerts and results in a delayed response or no response at all.

How does alert fatigue apply to security?

In application security, alert fatigue refers to the overwhelming number of security alerts developers and security professionals receive on a daily basis. These alerts may be generated at any point in the SDLC from a scan of an application's codebase pre-build, during QA, or in production. High rates of false-positive alerts compound the problem.

Security alerts have more than doubled in the past five years, and prioritizing these high alert volumes is a significant challenge. It's no surprise that the vast majority of security teams identify alert fatigue as a serious problem.

What can you do to combat alert fatigue?

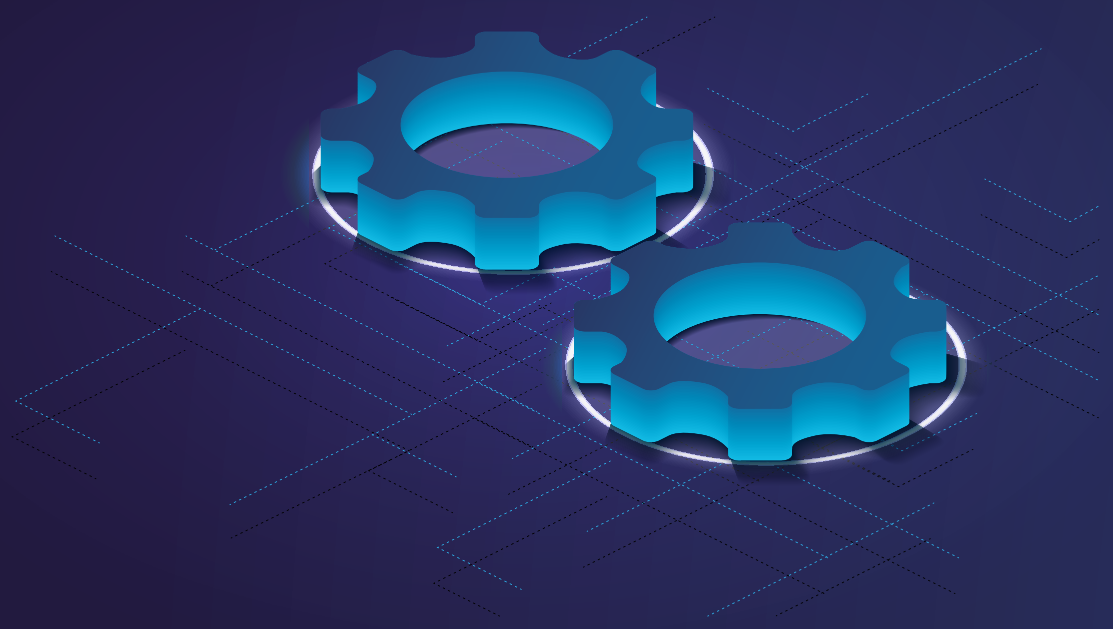
Take these first steps to prevent alert fatigue:

- Tune your system to reduce the amount of background noise. Look for security solutions that offer zero false positives and give you contextual data so that you understand the severity and relevance of alerts.
- Implement solutions that prioritize your security vulnerabilities, so you can concentrate first on the high-severity issues that would cause the greatest potential disruption.
- Automate as much of this process as possible so that your security professionals are free to remediate the most important threats.

Application Programming Interface (API) Security

By 2023, IDC predicts that digitally transformed enterprises will account for 52% of global GDP. Despite the 2020 global pandemic, investment in direct digital transformation is growing at a healthy compound annual growth rate (CAGR) of 15.5% from 2020 to 2023, reaching a market value of $6.8 trillion. Automation is powering this digital transformation, and APIs are key to driving automation and scaling productivity.

Think of APIs as the connective tissue linking ecosystems of technologies and organizations. APIs allow businesses to monetize data, form profitable partnerships, and open pathways of innovation. Using APIs, organizations are able to expand into new ecosystems. Unfortunately, many still struggle to secure APIs.



Why You Need API Security

With opportunity comes risk. Though APIs are key to digital transformation, they are simultaneously one of the biggest cybersecurity attack vectors and the most often overlooked. The average enterprise has approximately 1,000 applications, and this expansion of endpoints creates new attack surfaces. Robust API security is a necessity.

Unfortunately, API security differs from traditional application security, and it is hard. Because APIs move so much data, any broken, exposed, or hacked API can lead to a critical data breach. And API hacking does not require advanced technical capabilities. Even relatively inexperienced attackers can use basic tools to discover and exploit API traffic to perform credential stuffing attacks, exfiltrate databases, change account values, or conduct denial of service attacks on critical applications.

API Security Best Practices

API security is still relatively immature. The challenge of API security is that it needs to address client, backend, and network security risks, yet most of the attention is focused only on networks. When defining your API security strategy for 2021, don't treat API security as separate from your overall security plan. Adopt a holistic approach with a focus on authenticating identity and setting rate limits.

1. Adopt a Holistic Approach

API security shouldn't be its own separate security segment where you set token-based authentication once and then forget about it. It should be part of your overall security posture, actively managed and monitored. As a best practice, you should be building an ecosystem of security products and platforms that are integrated with each other and also provide insights across a wide range of platforms.

To achieve this, you need a comprehensive layered approach to protection across your entire ecosystem, including cloud, multi-cloud, and hybrid environments. Taking a holistic approach means looking at API security together with enterprise security and mobile security. You need an end-to-end approach because you will be attacked on all three of these fronts. Taking a wide view to securing APIs not only addresses potential vulnerability issues, but also offers protection for all of the infrastructure, networks, and data across your organization.

2. Focus on Identity

Hackers frequently impersonate users. As API security becomes more evolved, identity – representing people in encoded digital forms – becomes more important. On a basic level, identity involves confirming information about users, their rights, and their origin. Unfortunately, it's not uncommon for organizations to adopt HTTP Basic Authentication, API keys, or token-based authentication, yet ignore

The problem with this type of authentication is that it doesn't ensure that the person holding the key is supposed to have it, is authorized to use it, or even that the key is still valid. If you want to prevent vulnerabilities, you need a comprehensive plan that emphasizes identity.

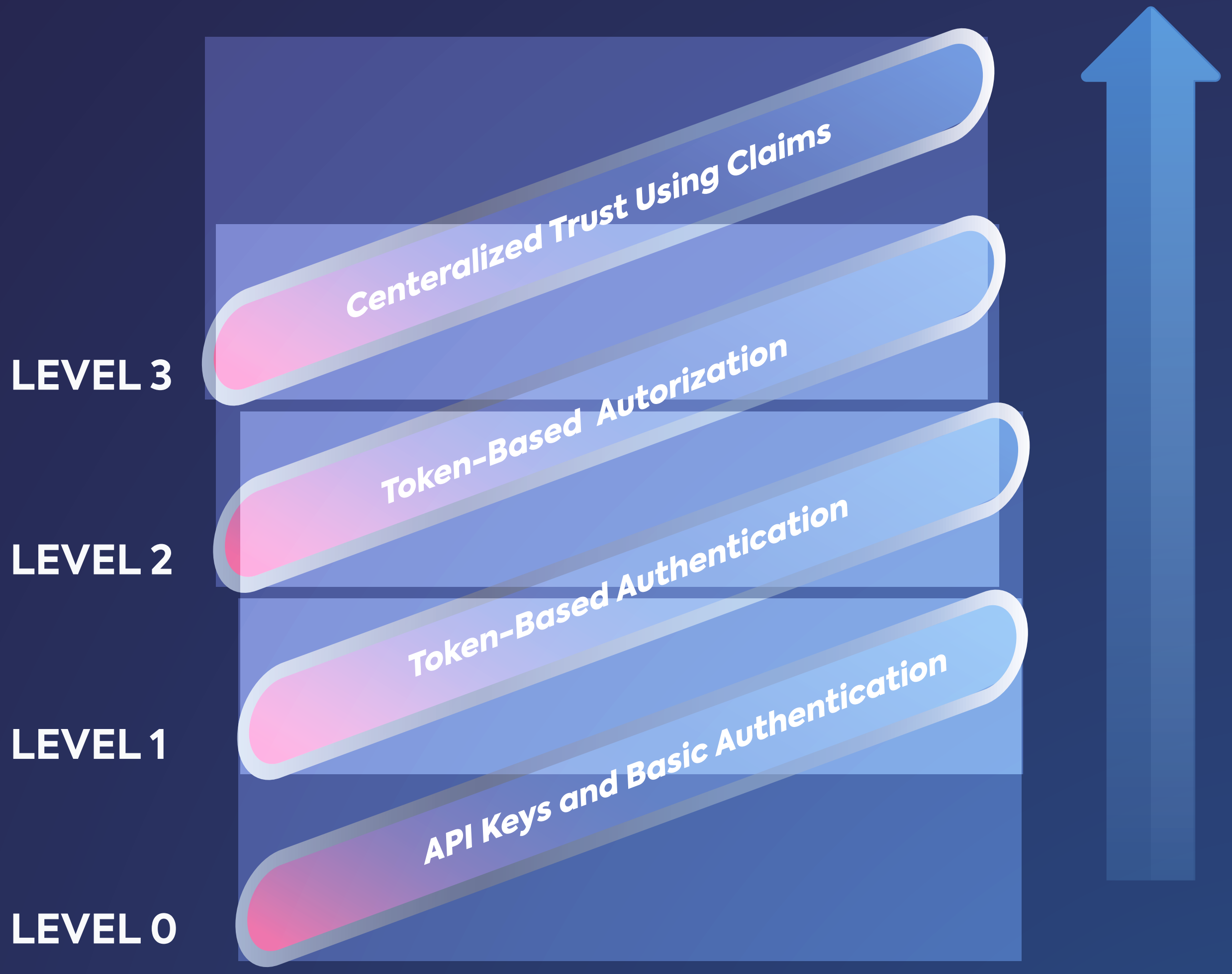
SolarWinds Data Breach, 2020

Widely accepted as the work of nation-state hacking group Cozy Bear, an arm of the Russian intelligence agency SVR, SolarWinds was the most headline-generating breach of 2020. Possibly the biggest breach of US networks in history, the attack impacted numerous U.S. government agencies, business customers, and consulting firms. Though this was a supply chain attack, SolarWinds Orion API also had an authentication bypass vulnerability that allowed attackers to execute API commands. The Orion API has highly privileged access to all platform components.

The API Security Model, based on the Richardson Maturity Model, describes API security in increasing levels of security, complexity, and efficiency. Level 0 is API keys and basic authentication. Level 1, which builds on Level 0, uses tokens to establish authentication. Level 2 is token-based authorization, which builds on previous levels but adds OAuth, an authorization standard that requires client requests be authorized by an OAuth server, to establish identity.

Level 3, the most mature API security level, is centralized trust using claims, usually through OAuth and OpenID Connect. To verify a claim, the requesting party calls the issuer, who returns data signed with a private key. The requesting party then sends this to a second party who replies and verifies the signature with a public key. Claims give you context and the ability to verify information. Essentially, if you trust the OAuth Server that issues keys, then you trust the claim being made.

The bottom line is you need to ensure that users are who they say they are and not malicious actors looking to penetrate your system. Focusing on identity and trust is the key to achieving mature API


LEVEL 3
Centeralized Trust Using Claims
LEVEL 2
Token-Based Autorization
LEVEL 1
Token-Based Authentication
LEVEL 0
API Keys and Basic Authentication

API SECURITY MATURITY MODEL

3. Monitor and Set Rate Limits

Ultimately, one of the best ways to secure your APIs is to watch what users are doing and limit their behaviors. You do this by monitoring APIs, which surprisingly, not enough organizations do. By monitoring APIs, usually in a production environment, you gain visibility into performance, availability, and functionality. You'll be able to identify unusual or suspicious behaviors more easily If you're looking for them. In addition to helping you identify malicious behaviors, API monitoring tools give you the added benefit of analyzing and optimizing API performance.

Rate limiting is both a critical part of API security and vital to scalability. If your API is receiving too many calls, it could be the sign of an attack. Malicious actors try to overwhelm APIs by flooding your system with requests in an attempt to bring it down. You want to make sure that your authorized users are able to access your API. Rate limits are an effective defense against DoS attacks and overloaded servers.

What the Future Holds: Key Takeaways for 2021

With environments like cloud native, serverless, containerized, microservices, IoT, CaaS, FaaS, and more, software development ecosystems are becoming increasingly more multilayered and diverse. As a result, the threat landscape is evolving. This, combined with a massive shift to remote work, is forcing organizations to rethink their security strategy and infrastructure.

We are staring down the precipice, and a new approach to security is necessary. The security requirements for cloud native environments are vastly different from an on-prem server. How do you protect all your endpoints when so many of them are out of your control? Long gone are the days when a firewall and antivirus software on your PC are enough.

Keeping diverse ecosystems and the lack of control in mind, several key trends to keep an eye on for 2021 include the following:

- The frenzy over all things cloud native will result in security gaps, challenges, and misconfigurations.
- Exposed APIs will be the next big attack vector for enterprises. You may see a large enterprise API breach make headlines.
- Software composition analysis solutions will offer better remediation advice with confidence scores based on crowd-sourced data.
- DevOps goals will be more aligned with strategic business outcomes, especially for risk management and security.
- Security as Code will become more common in protecting applications throughout the SDLC.

These days, the pace of development is so blisteringly fast that if the way security is delivered doesn’t change, we will all be exposed to malicious actors. It’s no longer sustainable to have security act as a discrete function at the end of development, nor can developers do it all on their own. Security must be a cross-functional team initiative that spans the entire product life cycle if we’re to keep the enterprise secure.