Hype Cycle for Application Security, 2023

Published 24 July 2023 - ID G00791986 - 110 min read

By Analyst(s): Dionisio Zumerle

Security and risk management leaders must familiarize themselves with innovations that can modernize their application security programs. By selectively employing application security innovations, they can increase the effectiveness of their layered defense, and mitigate quickly emerging threats.

Analysis

What You Need to Know

Application security is experiencing a period of intense transformation and growth. [1, 2] Organizations are producing their own applications and are using faster and more agile methodologies to accomplish this. In doing so, they rely on developers to secure code, rather than security practitioners. At the same time, new application architectures, such as cloud-native architectures, are challenging the established approaches to addressing enterprise security. All this is rendering existing security controls less effective. As if these challenges were not enough, we are experiencing an increase in application- and workload-level real world attacks. [3]

Luckily, there are several innovations that will be maturing over the next five years that will reshape the way enterprises conduct application security programs. Organizations are experimenting with artificial intelligence, automation of application security workflows and offerings that bring together infrastructural and application security controls.

The Hype Cycle

A driving force for application security, DevSecOps, is now a mature, widely adopted discipline. However, there are still challenges for organizations, and innovations are emerging to address them. For example, application security posture management (ASPM) allows organizations to implement their DevSecOps policies. A much newer innovation, policy as code (PaC) allows security teams to demand or enforce security policies and auditing controls. While it is not a new practice, this year we recognized secure code training as a stand-alone discipline.

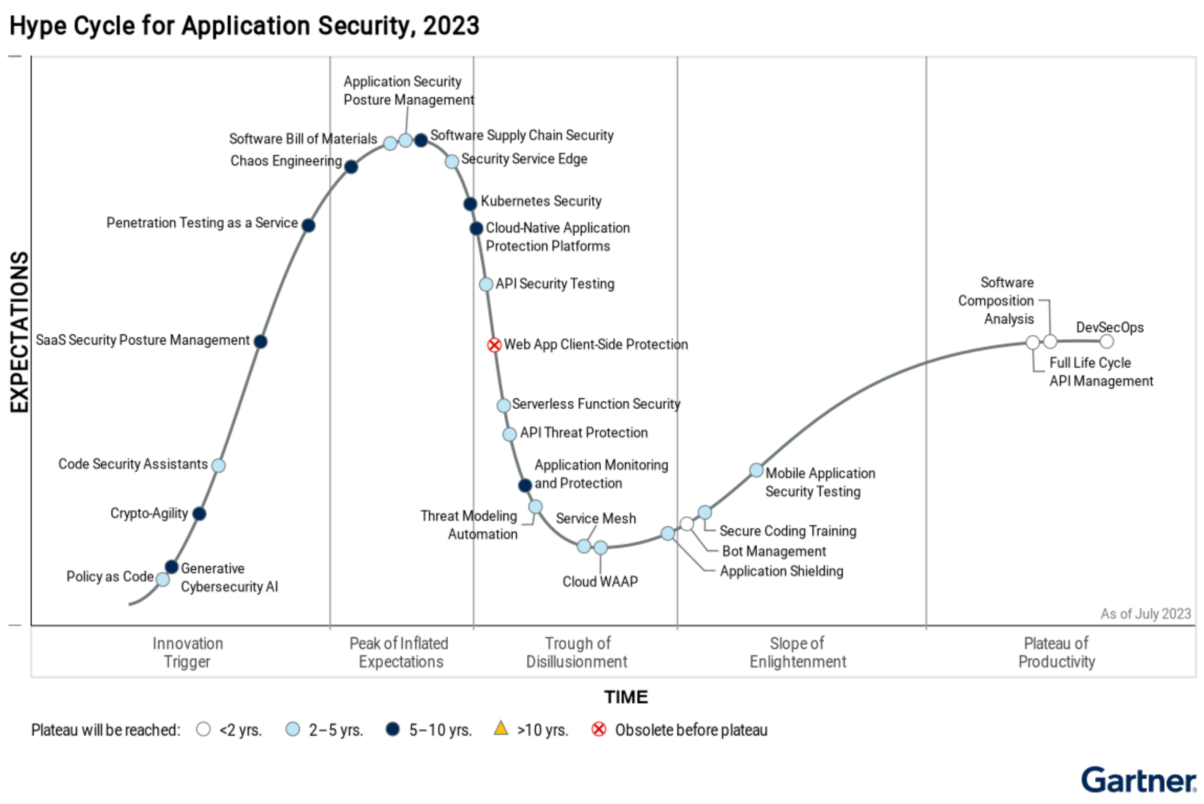
Innovation in the application security domain is not limited to solving processual issues. As organizations migrate their applications to cloud, they need tools that can span from development to runtime, and cover both application and infrastructural aspects. Cloud-native application protection platform (CNAPP) tools exemplify this, providing container and infrastructure as code scanning, cloud security posture management (CSPM), runtime workload protection and configuration scanning in a tightly integrated tool. API threat protection emphasizes a particular architectural aspect of modern applications — APIs — that needs increased focus.

Cloud-native applications rely on distributed software development models, and third-party software components. This combination facilitates software supply chain attacks, which are rapidly increasing. [4] Software supply chain security (SSCS) is emerging as a discipline that merges various concepts, from securing development environments, to software composition analysis (SCA) and software bills of materials (SBOMs).

Generative AI is facilitating application security attacks at the hands of bad actors. However, it is also helping organizations improve application security. Secure code assistants and generative cybersecurity AI are two nascent innovations that help organizations automate tasks such as application vulnerability identification and remediation.

Application security is not just about homegrown applications. Securing third-party and SaaS applications is a growing problem. To address it, SaaS security posture management (SSPM) provides security posture and identity controls.

Figure 1: Hype Cycle for Application Security, 2023



The Priority Matrix

There are seven transformational innovations in this year's Hype Cycle.

Three innovations have received a lot of hype. Organizations are starting to experience their benefits, while also trying to adapt them to their realities:

- Security service edge (SSE) secures user access to the web, cloud services and private applications. It provides adaptive access control, data security and visibility.
- ASPM tools manage application risk through collection, analysis and prioritization of security issues to enable the enforcement of security policies and facilitate the remediation of security issues.
- SSCS is the set of processes and tools used to curate, create and consume software in ways that mitigate attacks against software and prevent its use as an attack vector.

Two innovations have just emerged and are rapidly evolving:

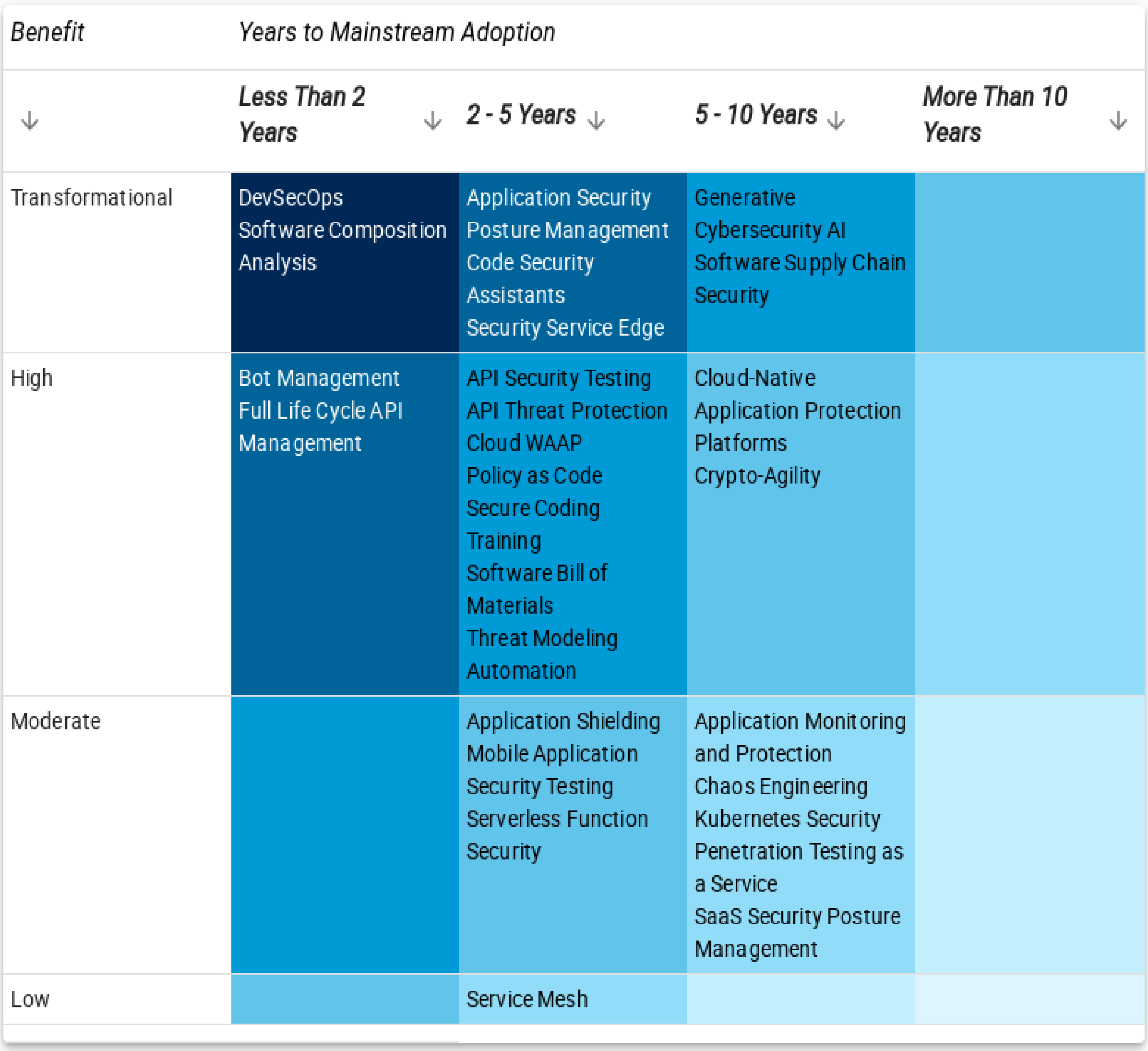
- Code security assistants use artificial intelligence to help developers identify and remediate security vulnerabilities in code.
- Generative cybersecurity AI facilitates and expedites various cybersecurity tasks including reporting and code remediation.

Two innovations have reached maturity:

- DevSecOps is the discipline that enables security teams to keep pace with development and operations teams in modern application development.
- SCA, which identifies vulnerabilities in open-source packages and other artifacts, is a fundamental element of SSCS and an essential element of application security testing.

Table 1: Priority Matrix for Application Security, 2023

(Enlarged table in Appendix)



Benefit	*Years to Mainstream Adoption*			
↓	*Less Than 2 Years* ↓	*2 - 5 Years* ↓	*5 - 10 Years* ↓	*More Than 10 Years* ↓
Transformational	DevSecOps Software Composition Analysis	Application Security Posture Management Code Security Assistants Security Service Edge	Generative Cybersecurity AI Software Supply Chain Security	
High	Bot Management Full Life Cycle API Management	API Security Testing API Threat Protection Cloud WAAP Policy as Code Secure Coding Training Software Bill of Materials Threat Modeling Automation	Cloud-Native Application Protection Platforms Crypto-Agility	
Moderate		Application Shielding Mobile Application Security Testing Serverless Function Security	Application Monitoring and Protection Chaos Engineering Kubernetes Security Penetration Testing as a Service SaaS Security Posture Management	
Low		Service Mesh		

Source: Gartner (July 2023)

Off the Hype Cycle

A number of innovations are off the Hype Cycle this year. Some of them, such as web application firewall appliance, have reached full maturity. Others, such as enterprise app stores, DevOps test data management, externalized authorization management (EAM), privacy by design and digital product analytics are still maturing and are present in other Hype Cycles. However, with the revised scope of the Hype Cycle for Application Security and the additional innovations introduced this year, they have been removed from this Hype Cycle.

Other changes include:

- Application security requirements and threat management has been renamed to threat modeling automation.
- Application Security Orchestration and Correlation has been replaced by application security posture management.
- Securing developer environments was incorporated into software supply chain security.
- The crowdsourced software security testing platforms (CCSTPs) innovation has been replaced by penetration testing as a service (PTaaS).

On the Rise

Generative Cybersecurity AI

Analysis By: Jeremy D'Hoinne, Avivah Litan, Mark Horvath, Wilco van Ginkel

Benefit Rating: Transformational

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

Generative cybersecurity AI technologies generate new derived versions of security-related and other relevant content, strategies, designs and methods by learning from large repositories of original source data. Generative cybersecurity AI can be delivered as a public or privately hosted cloud service or embedded with security management interfaces. It can also integrate with software agents to take action.

Why This Is Important

Enterprises witness many applications leveraging foundation models that can read multimodal objects (such as sensory data and images), following the first applications based on large language models (LLMs).

Cybersecurity technology providers can exploit generative cybersecurity AI to improve existing workflows, be a proxy of existing analytics, and generate security configuration or realistic attack data. Soon, applications will include autonomous agents, which can work using high-level guidance without a need for frequent prompting.

Business Impact

Existing vendors and new startups will add generative cybersecurity AI, expanding or replacing features. They will start implementing it with resource-intensive tasks, such as incident response, exposure or risk management, or code analysis.

Organizations will benefit from generative cybersecurity AI as it can improve efficiency and shorten response times to cybersecurity risks and threats. The pace of adoption will vary across industries and geographies due to security and privacy concerns.

Drivers

- ChatGPT is one of the most hyped and fastest-adopted AI technologies ever. It relies on generative AI foundation models, which are largely trained on massive internet datasets.
- Security operations center (SOC) teams cannot keep up with the deluge of security alerts they must constantly review, and are missing key threat indicators in the data.
- Risk analysts need to speed up risk assessments, and be more agile and adaptable through increased automation and prepopulation of risk data in context.
- Organizations continue to experience skill shortages and look for opportunities to automate resource-intensive cybersecurity tasks. Use cases for the application of generative AI include: synthesizing and analyzing threat intelligence; generating remediation suggestions for application security, cloud misconfigurations and configuration changes to adjust to threats; generating scripts and codes generation; implementing secure code agents; identifying and graphing key security events in logging systems; conducting risk and compliance identification and analysis; automating the first steps in incident response; tuning of security configuration adjustment; creating general best practice guidance.
- Generative cybersecurity AI augments existing continuous threat exposure management (CTEM) programs by better aggregating, analyzing and prioritizing inputs. It also generates realistic scenarios for validation.
- Generative AI offerings include the ability to fine-tune models, develop applications using prompt engineering and integrate with prepackaged tools and plugins through APIs. These possibilities open up a path for providers to add generative cybersecurity AI.
- Microsoft has already demonstrated a preview version of its security co-pilot feature, which is expected to drive competitors to embed similar approaches.
- Security program performance solutions and activities can solve their increasing demand for business alignment. Further, they can perform scenario planning for budget (re)allocation, and efficiency and effectiveness indicators and corrections.

Obstacles

- The cybersecurity industry is already plagued with false positives. Early examples of “hallucinations” and inaccurate responses will cause organizations to be cautious about adoption or limit the scope of their usage.

- Best practices and tooling to implement responsible AI, privacy, trust, security and safety for generative AI applications do not fully exist yet.
- Privacy and intellectual property concerns could prevent sharing and usage of business- and threat-related data, reducing the accuracy of generative cybersecurity AI outputs.
- As generative AI is still emerging, establishing the trust required for its wider adoption will take time. This is especially true for the skill augmentation use cases, as you would need the skills you are supposed to augment, in order to ensure the recommendations are good.
- Uncertainty on laws and regulations related to generative AI may slow down adoption in some industries, for example regulated industries in EU countries subject to GDPR compliance.

User Recommendations

- Pick initial use cases carefully. First implementations might have a higher error rate than more mature techniques already in place.
- Monitor the addition of generative AI features from your existing providers and beware of "generative AI washing." Don't pay a premium before obtaining measurable results.
- Choose fine-tuned models that align with the relevant security use case or fine-tune in-house models from base models offered by the providers.
- Refrain from sharing sensitive and confidential data with hosted models until verifiable privacy assurances are provided by the host.
- Apply AI security frameworks, such as AI TRiSM. Work with your legal team on data privacy and copyright issues.
- Implement a documented approval workflow for allowing new generative cybersecurity AI experiments.
- Make it mandatory from a policy standpoint that any content (that is, configuration or code) generated by an AI is fully documented, peer-reviewed by humans and tested before it is implemented. If not possible, consider any AI-generated content as "Draft Only" when used for critical use cases.

Gartner Recommended Reading

4 Ways Generative AI Will Impact CISOs and Their Teams

Innovation Insight for Generative AI

Market Guide for AI Trust, Risk and Security Management

Policy as Code

Analysis By: Paul Delory

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Emerging

Definition:

Policy as code (PaC) languages express governance and compliance rules as code, so they can be enforced programmatically by automation tools. PaC languages are often domain-specific and declarative. With PaC, policies are treated as software, making them subject to version control, code review and functional testing. The most mature PaC tools can render any business logic in code. You can use them today to enforce infrastructure compliance, authorization, Kubernetes admission control, and more.

Why This Is Important

In the most mature automation pipelines, infrastructure and operations (I&O) engineers mostly spend time on optimization, governance and compliance. They no longer build infrastructure; that work has been automated and turned over to others. Now, the I&O function builds the guardrails around the infrastructure services that their end users consume. I&O must align with security and compliance teams. PaC brings policy enforcement into their automation pipelines, while preserving a separation of duties that mirrors a typical IT org chart.

Business Impact

Policy as code improves:

- **Security, compliance and automation:** PaC combined with infrastructure automation implements policies automatically, with implicit compliance guarantees.

- **Alignment of security and operations teams:** PaC allows security and compliance teams to interface directly with automation pipelines to ensure conformance.
- **Visibility and auditability:** PaC provides both documentation of policies and evidence they are being enforced.
- **Time and effort spent:** PaC means less toil for operators.

Drivers

- **PaC tooling:** Several dedicated PaC tools are now on the market, many of them are open-source. The Open Policy Agent, a Cloud Native Computing Foundation project, has become the *de facto* standard for PaC. Indeed, even some other PaC tools now use Open Policy Agent policies alongside or instead of their own policy engines.
- **Increasing regulation:** New regulations such as GDPR have increased both the difficulty of compliance and the pressure on compliance teams. PaC allows compliance teams and auditors to document their policies in detail, and to verify that they are being enforced.
- **Security breaches:** Similarly, a spate of newsworthy security breaches at public companies — caused by infrastructure misconfigurations — has put every IT organization's security and compliance practices under increased scrutiny. No I&O team wants its security failures to be the reason for its company getting negative headlines.
- **Growth of DevOps and DevSecOps:** More and more companies are embracing DevOps and DevSecOps — which means more and more companies are encountering the hard governance problems of automation. Many teams that implement infrastructure as code quickly are finding that they need better policy enforcement, and PaC can help.
- **Cloud optimization and cost control:** Beside their benefits for security and compliance, PaC tools can also be used to enforce the build standards for infrastructure, including budgets. In the public cloud, where oversized or unnecessary infrastructure incurs direct out-of-pocket costs, programmatically enforced policies can help to control spending.

Obstacles

- **Scarcity of downloadable content:** PaC tools will not gain real traction until they have an extensive library of community-generated content. Ideally, users would simply download the policies they need from a free, public repository, rather than having to write their own policies. Over time, as the user base expands and commercial offerings see increased adoption, PaC tools will reach a critical mass of downloadable content that supports real-world use cases.

- **Skill set:** Many I&O professionals lack the skills to operate automation and PaC tools effectively. Gartner clients routinely report that their automation and policy management are hindered primarily by people, not tools. PaC will magnify these existing skills challenges.

- **Organizational inertia:** PaC promises improved collaboration between I&O and security or compliance teams. But in some organizations, this change would be unwanted. Internal resistance of this kind will slow the rate, scope and scale of PaC initiatives.

User Recommendations

- **Start small:** Choose a pilot use case where PaC is likely to provide real business benefits, expanding to others once PaC has proven its value.

- **Upskill staff:** PaC languages are not always intuitive. Technical staff will need practice to reach proficiency.

- **Prioritize existing templates:** Focus your PaC efforts on use cases that have ready-made implementation templates — ideally, publicly available downloadable content. For example, almost every PaC tool on the market has a canned implementation of the CIS benchmarks.

- **Break down team silos:** Use PaC to build a common workflow for automation and policy enforcement that spans I&O, security and compliance teams.

- **Integrate PaC into automation pipelines:** Use PaC to build guardrails for automation tools, so that they cannot take actions that are out of compliance.

- **Measure before and after:** Use observability tools and value stream mapping to define your starting state, then compare it to the end state. Collect real data to quantify the value of PaC.

Sample Vendors

HashiCorp; Palo Alto Networks; Progress; Pulumi; Styra

Gartner Recommended Reading

Using 'Policy as Code' to Secure Application Deployments and Enforce Compliance

How to Protect Your Clouds With CSPM, CWPP, CNAPP and CASB

Innovation Insight for Continuous Compliance Automation

Innovation Insight for Cloud-Native Application Protection Platforms

Magic Quadrant for DevOps Platforms

Crypto-Agility

Analysis By: Mark Horvath

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Crypto-agility is the capability to transparently swap out encryption algorithms and related artifacts in an application, and replace them with newer, different and, presumably, safer algorithms. Because quantum computing poses an increasing threat to existing cryptography, Gartner expects crypto-agility to be a significant differentiator for technology vendors.

Why This Is Important

As quantum computing matures, asymmetric encryption faces the threat of being damaged or broken during the next five to seven years. Algorithms threatened are hard-wired into many applications, and most organizations lack a clear idea of the cryptography they're using.

Standard network/PKI protocols (e.g., Diffie-Hellman key exchange and TLS) will need quantum-safe deployments.

The National Institute of Standards and Technology (NIST) is standardizing quantum-safe alternatives for widespread use.

Business Impact

- Applications/communications using encryption or PKI need to be inventoried. Data about algorithms, key sizes, expiration dates and uses must be gathered into a metadata database.
- Suitable replacements must be identified for applications and use cases.
- The new algorithms are not drop-in replacements for cryptography, so alternatives must be tested and replacement policies must be generated.
- Vendor cryptography products must be identified, and vendors will need to provide a replacement schedule.

Drivers

- The development of quantum computers capable of breaking cryptography through Shor's or Grover's algorithms is estimated to be five to seven years away. This is much shorter than the typical life span of sensitive information found in most organizations.

- Keybreaking with Shor's algorithm goes linearly with key size but is limited by the number of available qubits; once it's been achieved, larger keys sizes will fall quickly, leading to a short runway to implement changes.

- Most organizations that have inventoried their cryptographic metadata have found they already have considerable technical debt with respect to PKI (e.g., expired certs, deprecated algorithms, short keys). Cleaning that up will substantially lower the organization's risk profile.

- New algorithms have additional uses that can foster novel business cases (e.g., stateful signatures, homomorphic encryption).

- Government organizations are beginning to require a quantum-safe encryption strategy for vendors selling to them (e.g., U.S. NSM-10, EO-14028). This includes all products or code that is included in the final product, including open-source software (OSS) and other vendor's products. Like with a software bill of materials (SBOM), this significantly expands the scope of these orders to include many vendors not otherwise selling directly to the government.

- Cryptography in vendor products will need to be identified, and vendors will need to provide a schedule for potential replacement.

Obstacles

- The time scale for change (five to seven years) is significantly beyond the expected tenure of many senior executives, leading to a "someone else's problem" mentality, leaving some decision makers to prioritize shorter-term projects.
- Many organizations don't know how to inventory their cryptographic usage, leaving them to choose between a vendor to do the work, or spinning up an internal program.
- The NIST standardization of current, approved algorithms has not been completed, leaving clients with nonstandard OSS versions of the algorithms.
- Many organizations lack the expertise needed to lead a cryptographic project of this magnitude.
- There is still some market confusion about what "quantum-safe" means, sometimes leading organizations to evaluate relatively useless technologies that talk about "quantum" but don't solve the problem.

User Recommendations

- Start looking at your cryptographic inventory sooner, rather than later. This will help scope the project and identify key systems and data, allowing a controlled rollover to quantum-safe encryption.
- Experiment with the OSS version of the NIST standardization candidates to determine what effect this will have on your infrastructure (e.g., lattice encryption is generally slower, and ciphertext sizes are larger than existing algorithms).
- Ask vendors what their plans are for replacing algorithms, and when quantum-safe versions of their products will be available.
- Explore off-brand uses for some of the new algorithms (e.g., homomorphic encryption).

Sample Vendors

Cryptomathic; DigiCert; Entrust; IBM; IronCore Labs; ISARA; Sandbox AQ; Thales; Utimaco; Venafi

Gartner Recommended Reading

Preparing for the Quantum World With Crypto-Agility

Infographic: How Use Cases Are Developed and Executed on a Quantum Computer

Cool Vendors in Quantum Computing

Code Security Assistants

Analysis By: Mark Horvath

Benefit Rating: Transformational

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Code security assistants are technologies that help developers identify and remand security vulnerabilities in code. To do so, they use various forms of artificial intelligence such as generative AI, through autoremediation suggestions, direct code assistance or chatbots. Code security assistants are primarily delivered as a feature of existing application security testing products. They use cloud services to analyze code and make recommendations.

Why This Is Important

- Developers are generally not well-trained in identifying and resolving security issues in their code. Code assistants offer a way around this.
- When well-trained on secure coding data, large language models (LLM) such as ChatGPT are excellent tools for taking complex ideas and formatting them for maximum clarity.
- Developers tend to write better secure code when they can ask questions and get effective answers. While most organizations today tend to supply answers through developer coaches, LLMs offer a new, more scalable way.

Business Impact

Impacts include:

- Security flaws created at the source are expensive to remediate later. This impacts not only security, but performance, reliability, usability, and cost to mitigate.

- Unlike most tools today, AI code assistants may be in a unique position to optimize several of these variables at the same time.
- Code security assistants are a tool for closing one of the remaining gaps — just-in-time training and remediation — in shifting security left in the modern secure software development life cycle (SDLC).

Drivers

- Code assistants like GitHub Co-Pilot are growing rapidly as they become increasingly popular with developers. Adding improved guidance for security is a natural evolution, and should improve both code quality and security outcomes.
- Shift-left initiatives — that is, moving work from the runtime environment (the right) to the developer (left) — are often slow to be adopted or fail completely because the channels needed to supply developers with solid security guidance are limited and somewhat fragile. Initiatives often rely on a few experts to help guide the whole organization. Good code security assistants offer an opportunity to add experts at many layers of the organization in a way that is easily consumed by developers.

Obstacles

- Most organizations are worried about exfiltrating private data or intellectual property through AI assistants. There are ways to protect against this, but trust is fairly low.
- A recent Stanford university study found that developers who used security assistants ended up making more mistakes after adopting them than before.
- LLMs occasionally hallucinate, identifying problems that are not real or are wrong, or giving explanations that sound plausible but are unsecure or otherwise unwise to use. This can be hard to detect.
- Code ownership issues are a concern, especially if an AI assistant has contributed a significant amount of code. Given that the assistants are generally using code they were trained on, rather than creating original code, accidentally acquiring another organization's IP is a concern.
- Overcompensating for security while ignoring issues like performance is typically how these kinds of optimizers have failed in the past, and there is some concern that could happen with AI assistants.

User Recommendations

- Continue to use traditional application security testing (AST) tools (like SAST or SCA) to measure security and code quality issues over time when adding an AI assistant. While it's not uncommon to see an immediate increase in security metrics, be sure to watch the other indicators of code quality over time.
- Check with vendors about how they handle privacy and confidentiality issues with both your code and code used to train the model.
- Keep, but adjust existing methods of improving code security as assistants become a bigger part of the developer experience.
- Randomly sample the assistant's code suggestions to check for quality and security.
- Use AI code assistants that work with LLMs to offer developers customized explanations of common security flaws. LLMs have the ability to rephrase dry, technical security jargon in a way that is more meaningful to developers and will help them understand more about how to write secure code.

Sample Vendors

BurpGPT; BLACKBOX AI; GitHub; Orca AI; Ox Security; Semgrep; Tabnine

Gartner Recommended Reading

Emerging Tech: Generative AI Code Assistants Are Becoming Essential to Developer Experience

Innovation Insight for Generative AI

Invest Implications: Emerging Tech: Generative AI Code Assistants Are Becoming Essential to Developer Experience

Magic Quadrant for Application Security Testing

SaaS Security Posture Management

Analysis By: Charlie Winckless

Benefit Rating: Moderate

Market Penetration: 1% to 5% of target audience

Maturity: Adolescent

Definition:

SaaS security posture management (SSPM) continuously assesses the security risk and manages the security posture of SaaS applications. SSPM tools offer a range of capabilities, such as reporting on and suggesting improvements to native SaaS security settings, managing identity permissions, and identifying interconnected SaaS applications. Some tools also provide a comparison against industry frameworks, data visibility and democratized or fully automated remediation.

Why This Is Important

SaaS is the delivery model for many critical enterprise applications, leading to large amounts of sensitive information being stored outside the traditional controls of the corporate network. These applications and their interconnections (commonly to other SaaS applications) are increasingly complicated to configure and manage security, an issue compounded by the fact that everyone is different. SSPM platforms provide consistent and automated visibility into the security of SaaS applications

Business Impact

SaaS usage is both widespread and growing in Gartner clients. Security service edge (SSE) vendors provide protection of sensitive data and user access at the network layer, but are often blind to complex configuration errors and SaaS-to-SaaS communication. SSPM tools reduce exposure to common SaaS risks by continuously scanning for and reducing configuration mistakes, suspicious SaaS-to-SaaS communications, and overly scoped permissions.

Drivers

- As more valuable data is processed using SaaS, attackers will increasingly target these applications to breach sensitive data. The primary source of cloud breaches is a failure in configuration or theft of a token. It is extremely rare for it to be an underlying flaw in the platform.
- SaaS's rising popularity, complexity and interconnectedness have created blind spots (for example, an unprotected connected application might be able to access data with minimum restrictions) and control gaps (particularly regarding SaaS configuration) in ever-more critical applications. Traditional controls and even SSE controls cannot manage and mitigate these gaps effectively. In such cases, SSPM tools provide the necessary visibility and protection.
- Regulation is becoming increasingly strict, imposing large penalties for data breaches. Regulators are starting to show concern about SaaS applications such as Cybersecurity and Infrastructure Security Agency (CISA)'s reference architecture for SaaS, Secure Cloud Business Applications (SCuBA).
- SaaS is not simple as a service. Trying to control SaaS via manual processes does not scale. Increased automation of configuration validation and remediation is necessary for effective control.

Obstacles

- There is increased overlap and consolidation of SSPM features and functions within SSE platforms and few organizations wish to have another console to govern another set of clouds.
- Many enterprises lack focus on SaaS security and lack any role responsible for SaaS security and therefore a buyer for SSPM. Acquisition and configuration of SaaS often reside within the business.
- Most current SSPM tools lack discovery capabilities, so rely on alternative tools (such as SSEs or next-generation firewalls) to identify the SaaS applications in use across the enterprise, or assume they are integrated into an enterprise identity provider.
- SSPM tools rely heavily on robust APIs from the SaaS provider for visibility into configuration and identity permissions. While many SaaS apps provide APIs, little standardization exists and most smaller vendors have limited or no API-based visibility.

User Recommendations

- Evaluate the current SSPM-like capabilities of existing tools, particularly if you have an SSE or SMP. If they provide sufficient visibility and management of SaaS native controls, use them. Don't buy yet another product and invest tactically when you decide to purchase to address any identified shortcomings.
- Prefer SSPM tools with broad capabilities, not ones that focus just on interconnection, or just on configuration.
- Configure the SSPM tool to crawl through each new release of governed SaaS applications to discover new functions and potential attack surfaces, such as exposed APIs, to maintain full visibility and compliance.
- Pressure vendors in established cloud security and management markets to broaden their capabilities to offer SSPM capabilities, including automation for SaaS control with a specific focus on SSE and SMP providers.
- Establish shorter-term contracts if SSPM is needed to address a gap.

Sample Vendors

Adaptive Shield; AppOmni; Axonius; Microsoft; Netskope; Obsidian; Palo Alto Networks; Suridata.ai; Valence

Gartner Recommended Reading

Tool: Framework for Developing SaaS Security Policy

Magic Quadrant for Security Service Edge

Critical Capabilities for Security Service Edge

Market Guide for SaaS Management Platforms

Quick Answer: Cloud, Kubernetes, SaaS — What's the Best Security Posture Management for Your Cloud?

Penetration Testing as a Service

Analysis By: Mitchell Schneider, Jeremy D'Hoinne, Carlos De Sola Caraballo, William Dupre

Benefit Rating: Moderate

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Penetration testing as a service (PTaaS) provides technology-led, point-in-time and continuous application and infrastructure testing aligned with penetration testing (pentesting) standards, which have traditionally relied heavily on human pentesters using commercial/proprietary tools. The service is delivered via a SaaS platform, leveraging a hybrid approach of automation and human pentesters (crowdsourced or vendors' in-house team) to increase the efficiency and effectiveness of the results.

Why This Is Important

Pentesting is foundational in a security program and mandated by various compliance standards (e.g., PCI). PTaaS delivers a platform that enables faster scheduling and execution of pentests, and real-time communications with testers and visibility of test results. It provides API access to enable integration with existing DevOps and ticketing solutions for workflow automation. It also provides the ability to document and track pentesting results to demonstrate progress over time to leadership/auditors.

Business Impact

PTaaS complements vulnerability scanning and application security testing, and provides cost-optimization and quality improvement of pentesting output and validation of vulnerability status. PTaaS enables organizations to elevate their security posture through continual assessment. It integrates validation earlier in the software development life cycle compared with traditional pentesting phases by giving access to real-time findings delivered through the platform, therefore enabling faster reduction of exposure.

Drivers

- Organizations are turning to PTaaS to deal with the increase of attack surfaces due to accelerating use of public cloud and expansion of public-facing digital assets. PTaaS allows developers to talk to and receive guidance from pentesters instead of arguing with scanners, such as dynamic application security testing/static application security testing (DAST/SAST) scanners.

- Organizations with limited in-house security expertise must meet their compliance and risk management objectives, in addition to improving their security posture, and therefore look to pentesting services to meet these initiatives.

- In order to meet fast production deadlines, security-aware organizations must integrate a more agile way of conducting pentesting into their continuous integration/continuous delivery (CI/CD) pipelines for their DevSecOps practices.

- Gartner clients have expressed an appetite to test on a more frequent basis; however, manual pentesting is cost-prohibitive in modern infrastructure (e.g., IaaS, SaaS and third-party subscriptions).

Obstacles

- Selecting a suitable PTaaS vendor in the market will be difficult, as their capabilities vary. Vendors use one or a combination of automation and human testers, which are in-house or community-led — typically vetted freelancers — to perform penetration testing for the client organization.

- Most of the PTaaS vendors in the market focus on the internet-facing digital assets, like web and mobile applications, which may only partially fulfill client requirements.

- There is confusion between PTaaS and bug bounty programs, as many bug bounty vendors also now offer PTaaS.

- Heavily regulated industries may still be required to contract a third party to perform a traditional, consulting type of pentest due to compliance requirements; therefore, PTaaS may not be an acceptable alternative.

User Recommendations

- Determine which option/mix of penetration testing programs is best for your organization: compliance-driven service engagement; PTaaS; in-house red team leveraging an automated pentesting tool; or bug bounty.
- Identify and evaluate the pentesting scope and requirements that PTaaS vendors will be able to fulfill before engaging with vendors. PTaaS is well-aligned to both application testing and external infrastructure testing. Not all of the vendors will be able to replace internal infrastructure pentests, wireless, social engineering and physical assessments.
- Favor hybrid scanning models that combine human analysis and automation to increase both effectiveness and efficiency.
- Select a PTaaS vendor that aligns with relevant compliance requirements, and not just focused on internet-facing infrastructure and applications.
- Seek PTaaS vendors that provide customized and tailored guidance throughout the life cycle of their service to alleviate the security skills gap.

Sample Vendors

Bishop Fox; BreachLock; Bugcrowd; Cobalt Labs; Evolve Security; HackerOne; NetSPI; OccamSec; Raxis; Synack

Gartner Recommended Reading

How to Select a Penetration Testing Provider

Understand the Types, Scope and Objectives of Penetration Testing

At the Peak

Chaos Engineering

Analysis By: Jim Scheibmeir, Hassan Ennaciri

Benefit Rating: Moderate

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

Chaos engineering is the use of experimental and potentially destructive failure testing or fault injection to uncover vulnerabilities and weaknesses within a distributed system. Chaos engineering tools provide the ability to systematically plan, document, execute and analyze an attack on components and whole systems throughout the life cycle of the system. This planning may include the injection of random timing or attack executions.

Why This Is Important

Many organizations rely on test plans that overemphasize functionality and underemphasize validating the system's reliability and resilience. The distribution and complexity of systems makes understanding them more difficult. Chaos engineering (CE) shifts the focus of testing a system from the "happy path" toward testing how it can degrade gracefully or continue to be useful and secure while under various levels of impact. Applying CE enables improvements to system knowledge and documentation.

Business Impact

CE is aimed to minimize time to recovery and the change failure rate, while maximizing uptime and availability. Addressing these elements helps improve customer experience, satisfaction, retention and acquisition. Gartner inquiries regarding CE increased by over 11% between 2021 and 2022.

Drivers

- Increased complexity of systems and increasing customer expectations are the two largest drivers of CE and the associated tools.

- As systems become more rich in features, they also become more complex in their composition and more critical to digital business success.

- Overall, CE helps organizations become more resilient across their processes, knowledge and technology.
- Teams often lack the confidence to handle failures and the psychological safety to take action to resolve incidents. CE can help build that confidence.

Obstacles

- Within many organizations, the predominant view of CE is that the practice is random, first implemented during production, and increases, rather than reducing, risk.
- Organizational culture and attitudes toward quality and testing can present barriers to the adoption of CE. When quality and testing are only viewed as overheads, there will be a focus on feature development over application reliability.
- It can be challenging just to secure the time and budget to invest in learning CE and associated technologies. Organizations must reach minimum levels of expertise so that value is returned.

User Recommendations

- Utilize a test-environment-first approach by practicing CE in preproduction environments.
- Incorporate CE into your system development, CI/CD or testing processes.
- Build out incident response protocols and procedures, as well as monitoring, alerting and observability capabilities, in tandem with the advancement of the CE practice.
- Utilize scenario-based tests — known as "game days" — to evaluate and learn about how individual IT systems would respond to certain types of outages or events.
- Investigate opportunities to use CE in production to facilitate learning and improvement at scale as the practice matures. However, Gartner believes that very few organizations purposely use CE in their production environments.
- Formalize the practice by adopting a platform or tool to track the activities and create metrics to build feedback for continuous improvements.

Sample Vendors

Amazon Web Services; ChaosIQ; Gremlin; Harness; Microsoft; Steadybit; Verica

Gartner Recommended Reading

Quick Answer: What Metrics Should We Use to Assess and Improve Software Quality?

Predicts 2023: Observing and Optimizing the Adaptive Organization

Top Strategic Technology Trends for 2023: Digital Immune System

Predicts 2023: How Innovation Will Transform the Software Engineering Life Cycle

Application Security Posture Management

Analysis By: Dale Gardner

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Application security posture management (ASPM) tools continuously manage application risk through collection, analysis and prioritization of security issues from across the software life cycle. They ingest data from multiple sources, correlate and analyze findings for easier interpretation, triage and remediation. They enable the enforcement of security policies and facilitate the remediation of security issues while offering a comprehensive view of risk across an application.

Why This Is Important

Modern applications are complex, and siloed data sources make visibility and control exceptionally difficult. ASPM supports integration and interoperability between application security and the DevOps environment, enabling organizations to implement DevSecOps policies and processes in their software development life cycle (SDLC). Orchestration enables specific test regimens and release controls. Prioritization and triage supports the ability to focus on the most critical issues while assessing risk in terms meaningful to all stakeholders.

Business Impact

ASPM aids security and software engineering teams by integrating and orchestrating application security tools and controls, improving visibility and control, and enabling the measurement and management of risk. Triage of application security data (including test findings and monitoring) brings increased productivity by prioritizing resources to focus on the most critical issues. ASPM delivers clarity and improved insights — both from an operational and risk-oriented view — into application security status.

Drivers

- ASPM evolved from and replaces application security orchestration and correlation (ASOC), adding breadth of coverage and additional features. Because of this, and the state of the market, its position on the Hype Cycle has been reset.
- AppSec and software delivery teams struggle with prioritizing remediation and mitigation efforts throughout the SDLC, given increased application complexity and the rapidly growing volume of data provided by application security tools. ASPM solves this challenge by ingesting information from multiple sources, correlating results and automating triage.
- Prioritization capabilities aid in gaining acceptance and support for security efforts among engineering teams. Too often, these teams are inundated with data from multiple tools. Consolidating information, evaluating its validity, and prioritizing remediation is a cumbersome process that doesn't scale to address the amount of data and the speed of engineering processes. Ensuring security concerns are addressed becomes more time-consuming and error prone, which fuels the perception that security is a barrier, not a benefit. ASPM can help validate warnings, ensuring teams focus on tasks offering the greatest risk reduction.
- Security and engineering teams have difficulty in reporting the business and other risk postures of applications, absent meaningful business metrics and threat intelligence. ASPM products can assist in translating raw vulnerability data into a form more relevant to executives and application owners.
- In organizations with diverse development and deployment processes, establishing automated controls for policy enforcement is complex, leading to a tendency to establish and enforce "one size fits all" approaches to policy definition. ASPM's integration and intermediation capabilities between application development and deployment processes and security controls offer the ability to centralize management of those controls and allow for more granular approaches to enforcement.

Obstacles

- Effective automation of security testing presumes an organization understands the overall risk posture of an application, the types of testing needed and how best to respond to findings. If the underlying activities needed to develop this understanding have not been undertaken, efforts to articulate policies on which prioritizations and triage efforts will rely are complicated, potentially leading to reduced benefits from ASPM tools.
- Vendors tend to emphasize integration with either development or operations security tools. This presents a barrier to delivering a “full stack” view of an application’s security risks, spanning both developed code and infrastructure components. Progress to more broadly integrated products is evident.
- ASPM tools work by aggregating and analyzing data, and some level of abstraction is inherent. That poses a risk that critical information may be inadvertently overlooked in processing, yielding the potential for false positives, or a false sense of security.

User Recommendations

- Prioritize ASPM in organizations with diverse development teams and a wide assortment of security tooling.
- Identify key stakeholders who will use an ASPM solution. Because underlying processes and outputs will affect many parts of the organization, stakeholder support is crucial to success in implementation and use.
- In particular, ensure the integration of ASPM with the development pipeline will provide a good developer experience, which will be a key to success.
- Evaluate the ability of tools to scale to process the amount of data generated across the application life cycle.
- Evaluate support for legacy applications, since many offerings focus on cloud native applications.
- Examine native ASPM capabilities (for example, as a feature of application security testing or other security tools) as an alternative to an investment in a dedicated solution. Such offerings may be effective in more homogeneous environments.

Sample Vendors

Apiiro; Bionic; Dazz; Enso Security; Kondukto; Legit Security; Rezilion; Synopsys; Tromzo; Wabbi

Gartner Recommended Reading

Innovation Insight for Application Security Posture Management

Guide to Application Security Concepts

How to Select DevSecOps Tools for Secure Software Delivery

Market Guide for Continuous Compliance Automation Tools in DevOps

Adapting Cybersecurity to the Agile Enterprise

Software Bill of Materials

Analysis By: Mark Driver

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

A software bill of materials (SBOM) is structured, machine-readable metadata that uniquely identifies a software package and the open source or proprietary components used to build it. SBOMs are designed to track and share the details of software components and their supply chain relationships across organizations. This enables greater transparency, auditability and traceability throughout the software supply chain, expediting resolution of security and compliance issues.

Why This Is Important

Gartner estimates that 40% to 80% of the lines of code in new software projects come from third parties (for example, runtime, libraries, components and software development kits [SDKs]). Most of this external code comes from myriad open-source projects; the remaining proprietary code comes from suppliers that provide little or no transparency to its status or condition. The quality and security of these external software assets (while leveraged across thousands of users) vary widely from one to another.

Business Impact

SBOMs aim to solve a fundamental problem with sharing open-source and third-party software. While organizations can use the same components, it’s inefficient to duplicate efforts around tracking vulnerabilities and analyzing compliance violations. SBOM standards such as Software Package Data Exchange (SPDX) and CycloneDX establish a common infrastructure and a data exchange format to share details about software packages. This standardization reduces time to remediate issues through better collaboration between organizations.

Drivers

- **Software License Identification and Risk Assessment —** Like commercial software, open-source software (OSS) is almost always distributed according to the terms of a license agreement articulating the ways the software may be used. Some are considered permissive, others restrictive. Restrictive licenses could pose substantial legal risk and the loss of intellectual property. SBOMs can indicate the license used to distribute a particular software package in order to support the assessment of legal risks.

- **Need for Improved Security and Compliance —** Regulatory authorities in the U.S. (like the Food and Drug Administration [FDA] and Federal Energy Regulatory Commission [FERC]) and in Europe (the European Union Agency for Cybersecurity [ENISA]) are mandating SBOMs as a prerequisite deliverable for all organizations transacting with government agencies and regulated organizations. The goal of these regulations is to provide greater visibility into software components and dependencies to accurately determine an organization's exposure to security risks and vulnerabilities.

- **Need for Improved Visibility and Traceability of Software —** SBOMs can provide software engineering and vendor risk management teams with increased transparency into how software gets built, which components make up that software, and how quickly security vulnerabilities can be identified and remediated. When teams discover flaws or vulnerabilities in a component, they can use SBOMs to quickly identify all software that is affected by the vulnerable component. SBOMs also provide them with information to assess the potential impact and risks introduced by the vulnerable component.

Obstacles

- SBOMs are not a panacea. They're only as useful as the processes and tools implemented to process, analyze and leverage the information they contain. The impediments to adopting SBOMs include sharing SBOM data due to nonstandard data formats; inability to automate software delivery workflows based on the contents in the SBOM; and the ability to generate, consume and securely distribute them at speed and scale.
- The seamless integration of SBOMs into software development, packaging and release activities will be critical for their widespread adoption. The challenges in scaling the use of SBOMs include policy automation based on the information in the SBOM, secure distribution across organizational boundaries and managing their life cycle. The attestation of SBOMs for provenance is a prerequisite to treating them as a trusted source of dependency metadata. The dynamic nature of dependencies and their versions requires regenerating and updating SBOMs every time the artifacts are updated.

User Recommendations

- Enhance visibility into the complete software supply chain by tracking dependencies between open-source components in the software development life cycle (SDLC) using SBOMs.
- Discover affected components and share vulnerability data by using SBOM standards to exchange information about components and their relationships.
- Ensure SBOMs are rebuilt along with software artifacts by generating SBOMs during the software build process, rather than relying on pregenerated SBOM data. Use software composition analysis tools to generate and verify SBOMs for third-party, open-source components.
- Mitigate software supply chain security risks by requesting commercial software providers to provide standards-based SBOMs (for example, software package data exchange [SPDX], software identification [SWID], CycloneDX) during the procurement process.

Sample Vendors

Apiiro; Codenotary; Cybeats Technologies; Eracent; Finite State; Invicti Security (formerly NetSparker); OX Security; Revenera (Flexera); Rezilion

Gartner Recommended Reading

Emerging Tech: A Software Bill of Materials Is Critical to Software Supply Chain Management

Innovation Insight for SBOMs

Market Guide for Software Composition Analysis

How Software Engineering Leaders Can Mitigate Software Supply Chain Security Risks

Software Supply Chain Security

Analysis By: Dale Gardner

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Software supply chain security (SSCS) is the set of processes and tools used to curate, create and consume software in ways that mitigate attacks against software or its use as an attack vector. Curation focuses on assessing risks of third-party software and assessing its acceptability. Creation focuses on secure development and the protection of software artifacts and the development pipeline. Consumption validates integrity of software through verification, provenance and traceability.

Why This Is Important

Given triple-digit increases in software supply chain attacks, global regulatory mandates, the widespread use of open-source tools and exposures due to remote ways of working, securing the overall software supply chain is paramount for all parts of an organization. This is particularly critical given growing digitized processes and services, making software the central location where most intellectual property now originates and resides.

Business Impact

- SSCS enables management and mitigation of a variety of risks, such as compliance failures and a host of security incidents.
- SSCS is necessary to satisfy both regulatory requirements and buyer demands for increased transparency around a vendor's application security measures.
- SSCS can help avoid increased friction and lost productivity from ad hoc efforts to secure the development environment and application artifacts.

Drivers

- Software supply chain attacks have become increasingly sophisticated, with malicious actors exploiting weaknesses at every stage in the software procurement, development and delivery life cycle. The number of attacks has increased dramatically, with the Sonatype report citing an average 742% year-over-year growth rate between 2019 and 2022. These attacks pose risks threatening the efficiency and productivity of business and organizations, and the basic ability of those organizations to operate. They may cause data breaches and other security incidents, and loss of intellectual property, and even pose threats to human safety.
- In response, governments and regulatory agencies throughout the world are turning their attention to ways in which organizations can be encouraged — or forced — to improve software supply chain security efforts. In the United States, a substantial focus is on the "power of the purse," with multiple initiatives across all sectors of the federal government focused on preventing vendors with poor security practices from reaching lucrative markets. These activities are not limited to the United States. Governments throughout North America, the European Union and the United Kingdom, and through Japan and Southeast Asia have taken actions ranging from the passage of legislation to efforts to improve awareness and understanding of the issues.
- Organizations have only just begun to tackle the issues associated with software supply chain risks, placing many in a "catch-up" position where action must be taken. They have been slow to take steps to secure their software supply chains. According to Sonatype's 8th Annual State of the Software Supply Chain Report, only about 7% of respondents have taken broad action. This is consistent with inquiry trends noted at Gartner. While there have been significant increases in inquiry around SSCS, many questions remain focused on specific subsets of the broader security challenge.

Obstacles

- The majority of organizations lack a complete understanding of SSCS, and thus have yet to adopt a comprehensive view of the software supply chain and all its risks. Such a view would encompass security through the curation and selection of secure software (during development and procurement), during creation, and in consumption as organizations track the usage of code and evolving threats. For example, many organizations are focused on acquiring software bills of material (SBOMs), but have yet to establish how those artifacts will be evaluated, stored, and used.
- Efforts to articulate secure means of ensuring the integrity and provenance of software artifacts throughout the software creation, curation, and consumption processes are emerging, but uneven in scope, execution and adoption. For example, it's common for organizations to evaluate open-source software used in development for vulnerabilities, but most organizations are not proactive when identifying potential risks.

User Recommendations

- Adopt a comprehensive view of software supply chain security and do not fall into the trap of focusing solely on specific aspects of the problem.
- When evaluating third-party software for use, proactively assess it for security, legal and intellectual property risks.
- Evaluate and authorize the use of specific components, and establish a trusted repository of vetted code. Track the usage and deployment of third-party code to ease remediation efforts when vulnerabilities are discovered.
- Protect the entire development tool chain and associated artifacts from attack. Include the protection of code and other artifacts (e.g., IaC or configuration scripts) from unauthorized access or alteration, defending the delivery pipeline from attack, and hardening the operating environment.

Sample Vendors

Arnica; BluBracket; Chainguard; Cybeats; Cycode; GitGuardian; Legit Security; Ox Security; Palo Alto Networks (Cider Security)

Gartner Recommended Reading

Emerging Best Practices to Manage Digital Supply Chain Risks

How Software Engineering Leaders Can Mitigate Software Supply Chain Security Risks

Innovation Insight for SBOMs

Security Service Edge

Analysis By: Charlie Winckless, John Watts

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Security service edge (SSE) secures access to the web, cloud services and private applications. Capabilities include adaptive access control, data security, visibility and control. Further capabilities include an advanced threat defense and acceptable use control enforced by network-based and API-based integrations. SSE is primarily delivered as a cloud-based service and may include on-premises or agent-based components.

Why This Is Important

SSE improves organizational flexibility to secure the usage of web and cloud services, and remote work. SSE offerings are the convergence of security functions (at least, secure web gateways [SWGs], cloud access security brokers [CASBs] and zero trust network access [ZTNA]) to reduce complexity and improve user experience. They are delivered from the cloud. When organizations are pursuing a secure access service edge (SASE) architecture, SSE is paired with software-defined WAN (SD-WAN) to simplify networking and security operations.

Business Impact

Hybrid work is continuing to drive the adoption of public cloud services, especially of SaaS applications. Both hybrid work and the adoption of public cloud services remain business enablers for most Gartner clients. SSE allows the organization to support anytime-anywhere workers by using a cloud-centric approach to enforce a security policy when accessing the web, cloud services and private applications. Simultaneously, SSE reduces the administrative complexity of running multiple products.

Drivers

- Organizations need to secure user, application and enterprise data that is distributed, decentralized and requires secure remote access.
- For many enterprises, a significant amount of critical data is now hosted in SaaS. Therefore, there is a need to perform data loss prevention (DLP) on data that is located in, going to, and leaving these SaaS platforms.
- SSE enables flexible and primarily cloud-based security for users and devices without being tied to on-premises network infrastructure and connectivity. The same security outcome is delivered to users regardless of their location or connectivity.
- Administrators can have enhanced visibility on user traffic and a single configuration and monitoring location for this traffic.
- SSE allows organizations to implement a posture based on identity and context at the edge.
- By consolidating vendors, organizations reduce complexity, costs and the number of vendors used to enforce security policy. Using a single SSE platform rather than multiple point offerings, they can both reduce complexity and reduce gaps in security coverage.
- Sensitive data inspection and malware inspection can be done in parallel across all channels of access. SSE allows doing both inspections in parallel, leading to a better performance and more consistent configuration than doing them separately.
- An adaptive access can take into account more input signals and be more consistently enforced, regardless of the application location or type.
- Organizations look for deeper security capabilities when building a SASE architecture compared to vendors that may have a minimal set of security features as part of their SD-WAN offering.
- Tight integrations that exist between discrete SD-WAN and SSE vendors allow interoperability without requiring a single-vendor approach.

Obstacles

- As the market is being formed by the convergence of capabilities, vendors may be strong in certain capabilities and weak in others. Vendors may also lack overall tight integration between SSE capabilities or with SD-WAN vendors.
- Not all vendors provide sufficiently sensitive data identification and protection to manage business risks.
- Some vendors have focused less on SaaS security and integrations. However, businesses increasingly need this visibility and protection.
- Being cloud-centric, SSE typically doesn't address every need supported by on-premises controls such as internal firewalling.
- Organizations are concerned about uptime or availability of services that they depend on for their business. This is compounded by weak SLAs from some vendors.
- Not all vendors provide all features locally in all geographies, resulting in performance or availability issues.
- Switching costs from incumbent vendors or timing of contract expirations prohibit near-term consolidation.
- Migrating from a VPN will increase costs.

User Recommendations

- Exploit the converged market, consolidate vendors, and cut complexity and costs as contracts renew for SWGs, CASBs and VPNs by replacing them with a ZTNA approach.
- Approach SSE consolidation identifying which elements you may already have in place (for example, existing cloud-based CASB or SWG). Develop a shortlist of vendors based on your use cases regarding secure end-user requirements, the cloud services you use, and the data you need to protect.
- Inventory your equipment and contracts to implement a multiyear phaseout of on-premises perimeter and branch security hardware in favor of the cloud-based delivery of SSE.
- Global enterprises should validate that remote offices have acceptable performance and features with selected vendors. Vendor point of presence (POP) locations and service support are key.
- Actively engage with initiatives for branch office transformation, SD-WAN and Multiprotocol Label Switching (MPLS) offload to integrate cloud-based SSE into the scope of project planning.

Sample Vendors

Broadcom; Cisco; Cloudflare; Forcepoint; iboss; Lookout; Netskope; Palo Alto Networks; Skyhigh Security; Zscaler

Gartner Recommended Reading

2022 Strategic Roadmap for SASE Convergence

Magic Quadrant for Security Service Edge

Critical Capabilities for Security Service Edge

Adopt Security Service Edge (SSE) to Replace Stand-Alone SWG, CASB and ZTNA Products

Kubernetes Security

Analysis By: Charlie Winckless, Thomas Lintemuth

Benefit Rating: Moderate

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Kubernetes security refers to the implementation of security processes, testing and controls for the Kubernetes container orchestration platform. It operates in close conjunction with individual container security, but focuses on Kubernetes configuration and admission control.

Why This Is Important

Kubernetes expands the potential attack surface for an organization, as it provides the capabilities to deploy, scale, monitor and manage container infrastructures. This increase in attack surface and the broad adoption of Kubernetes by developers — and in managed cloud environments — means security teams need tools to provide visibility and control in these environments. Unsecure Kubernetes can expose containers and its ecosystem to various attacks.

Business Impact

Kubernetes is the most common container orchestration platform, and is in many ways a de facto standard. Kubernetes involves significant complexities in terms of access control and RBAC, container name spaces, Kubernetes networking, segmentation, admission control and other configuration, requiring automated tools that work closely with container security systems to protect the environment. Kubernetes security tools help address these potential exposures.

Drivers

- Developer adoption of containers and Kubernentes has been driven by their fit to DevOps style microservices. This adoption has expanded the attack surface, forcing security teams to use different tools and approaches to address orchestration exposures.
- Container as a service (CaaS) offerings built on Kubernetes, such as Amazon Elastic Kubernetes Service (EKS), Azure Kubernetes Service (AKS) and Google Kubernetes Engine (GKE), are on the rise. These require security integrations to provide coverage.
- Security and risk management leaders must address the security of their container orchestration — primarily Kubernetes — to mitigate misconfigurations and overly permissive access.
- Container environments spread the attack surface across many containers, each of which is a unique network endpoint. East-west traffic is vastly increased, and the value of segmentation is correspondingly higher. Kubernetes security solutions can address these segmentation requirements.
- Shift-left and general container security is bolstered by effective use of admission control to limit the deployment of malicious or vulnerable container images.
- Cloud-native application protection platforms are incorporating kubernetes security capabilities.

Obstacles

- Kubernetes security blurs the line between application security, infrastructure security and container security, creating overlap in vendors, offerings and responsibilities within the organization.
- Kubernetes security alone cannot address the overall security of a containerised application. Multiple other attack points exist.
- Kubernetes is often just one part of an architecture that also includes PaaS, and even serverless functions. Teams struggle to understand the whole system in these cases — not just the Kubernetes risks.
- While Kubernetes is widely adopted for container orchestration, other orchestration platforms do exist and may not be supported by the same products as Kubernetes.
- Unless the Kubernetes security solution is designed to provide minimal friction for developers, their adoption will be at best, resisted and, at worst, actively circumvented.

User Recommendations

- Use and enforce admission controls to prevent vulnerable or malicious containers from being deployed into your environment. Use these as a guard rail for your developers.
- Ensure that your tools support validating the Kubernetes orchestration environment for proper patch levels and correct and compliant configuration, both in development and in production.
- Ensure that the Kubernetes tools you adopt have the ability to do more than forward logs, and to support both native Kubernetes and Kubernetes-based CaaS options.
- Enforce least privilege and minimal access — avoid wildcard permissions and use name spaces to reduce permission access. Use RBAC for all users and service accounts.
- Use an industry standard baseline like the CIS benchmarks and automate scanning of your environment to ensure continuous compliance.
- Address Kubernetes networking risks by adopting tools that support network policy controls.
- Collaborate with engineering teams to implement security controls that require application context.

Sample Vendors

Aqua Security; ARMO; Lacework; Orca; Palo Alto Networks; Red Hat; Sysdig; Trend Micro; Wiz

Gartner Recommended Reading

Market Guide for Cloud Workload Protection Platforms

How to Make Cloud More Secure Than Your Own Data Center

Guidance Framework for Securing Kubernetes

Container Supply Chain: 10 Security Vulnerabilities and How to Address Them

Sliding into the Trough

Cloud-Native Application Protection Platforms

Analysis By: Neil MacDonald, Charlie Winckless

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Cloud-native application protection platforms (CNAPPs) are an integrated set of security and compliance capabilities designed to help secure and protect cloud-native applications across development and production. CNAPPs consolidate a large number of previously siloed capabilities, including container scanning, cloud security posture management, infrastructure as code scanning, cloud infrastructure entitlements management and runtime workload protection.

Why This Is Important

Comprehensively securing cloud-native applications requires the use of multiple tools from multiple vendors that are rarely well-integrated. This lack of integration and automation slows developers down and creates fragmented visibility of risk and friction. CNAPP offerings allow an organization to use a single integrated offering to protect the entire life cycle of a cloud-native application.

Business Impact

Cloud-native application protection platforms consolidate disparate fragmented security testing and protection tools that increase cost and complexity for IT. Using a CNAPP offering will improve developer and security professional efficacy. It will also reduce complexity and costs while maintaining development agility and improving the developer's experience.

Drivers

CNAPPs:

- Reduce the chance of misconfiguration, mistake or mismanagement as cloud-native applications are rapidly developed, released into production and iterated.

- Converge and reduce the number of tools and vendors involved in the continuous integration/continuous delivery (CI/CD) pipeline.
- Reduce the complexity and costs associated with creating secure and compliant cloud-native applications.
- Facilitate the reporting and auditing of cloud security posture/status.
- Improve developer acceptance with security-scanning capabilities that seamlessly integrate into their development pipelines and tooling.
- Place an emphasis on scanning proactively in development and rely less on runtime protection, which is well-suited for container as a service and serverless function environments.

Obstacles

- Cloud workload protection platform (CWPP) vendors that are good at runtime protection aren't necessarily good at integrating into development and vice versa.
- Cloud-native workloads in the form of containers and serverless functions don't require heavyweight runtime protection capabilities.
- There is no single CNAPP offering that does everything. Convergence of capabilities will occur, but will take place over several years.
- Organizations may have siloed purchases of application security testing tooling that is chosen by a different team that manages the runtime protection of workloads. Even at runtime, a separate team may be responsible for web application protection.
- Organizational immaturity in terms of cloud-native application development may inhibit adoption and fragment buying motions.
- Buying centers and influencers are shifting to newer roles such as DevOps architects and cloud security engineering, requiring information security teams to coordinate with these users.

User Recommendations

- Sign contracts of only one to two years because the market for CNAPP is changing rapidly.

- Solicit CWPP vendors to scan containers in development and add cloud security posture management (CSPM) capabilities, including infrastructure-as-code scanning.
- Select integrated offerings with flexible licensing models that allow you to only pay for the capabilities your organization is prepared to use.
- Evaluate the CSPM vendor's ability to add scan of Kubernetes security posture management (KSPM) as well as provide runtime Kubernetes protection capabilities.
- Consolidate open-source software (OSS) vulnerability scanning and software composition analysis through integrations or replacement within a CNAPP offering.
- Scan containers proactively in development for all types of vulnerabilities, not just vulnerable components, including hard-coded secrets, malware and Kubernetes misconfiguration.

Sample Vendors

Aqua Security; Cisco; Lacework; Microsoft; Orca Security; Palo Alto Networks; Rapid7; Sysdig; Trend Micro; Wiz

Gartner Recommended Reading

Market Guide for Cloud-Native Application Protection Platforms

How to Select DevSecOps Tools for Secure Software Delivery

How to Make Cloud More Secure Than Your Own Data Center

API Security Testing

Analysis By: Dale Gardner

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Adolescent

Definition:

API security testing is a specialized type of application security testing (AST) aimed at identifying vulnerabilities in APIs. Checks should include both traditional application vulnerabilities (such as injection attacks) and API-specific issues (such as broken-object-level authorization). Availability of an API specification (such as OpenAPI) is sometimes a prerequisite for effective testing. Discovery technologies help ensure that unknown APIs are identified and tested for vulnerabilities.

Why This Is Important

APIs represent a major attack surface for web-enabled applications. Attacks on and abuse of APIs result in serious adverse consequences, including data breaches and other security incidents. DevSecOps teams focus on the need for API security testing in development to prevent this, but APIs pose unique risks. Many organizations rely on traditional AST tools extended to support these requirements or speciality tools designed for APIs. Some vulnerabilities may require manual testing to detect.

Business Impact

APIs are a foundational element of many organizations' digital transformation efforts. Hence, securing APIs from attack and misuse is a continuing concern for many security and risk management (SRM) professionals. API-specific testing — pre- and postdeployment — builds a solid foundation for an overall API security strategy. Automated API discovery is needed because many organizations struggle to maintain an inventory of APIs and need help locating them so they are tested and managed.

Drivers

- In support of digital transformation efforts, APIs have become common in application architectures to enable information flow and support transactions between processes, applications and systems. However, that growth has brought increased attention from attackers, and APIs have become the primary attack surface for many systems.

- API attacks have resulted in a stream of data breaches and other security incidents, yielding significant damage to organizations and individuals. As a consequence, DevSecOps teams — along with the business leaders whose applications are supported by APIs — are increasingly interested in API testing and security.

- Because the creation, development and deployment of APIs may be loosely managed, security teams often have to contend with undocumented or shadow APIs, which exist outside normal processes and controls. Such APIs may be especially deficient in secure design and testing, posing an even greater risk. Hence, the discovery of APIs deployed to production is another important need.

- API security testing helps avert the tangible and intangible costs associated with breaches and other types of security incidents.

- Traditional AST tools — SAST, DAST and interactive AST (IAST) — were not originally designed to test for some of the unique types of vulnerabilities associated with typical attacks against APIs, or for newer types of APIs (such as GraphQL). API-specific vulnerabilities and modern API formats prompt security and development teams to implement specialized API security tools that are focused on testing, discovery of shadow APIs and protection from threats (or some combination of the three).

Obstacles

- Traditional tools offer inconsistent support for detecting API-specific vulnerabilities. APIs are susceptible to most traditional application attacks, but other attacks are common. For example, improper authorization checks around object identifiers have been cited in a number of breaches. Specialized tools or penetration testing may be needed for reliable detection of these flaws.
- Testing tools may require an API specification to perform effective testing.
- Testing tools may not support all API protocols. SOAP remains in widespread use, although it is being supplanted by REST APIs. GraphQL-based APIs are increasingly common, so additional support is required from tool vendors for effective testing.
- Some confusion remains in the marketplace, with various types of companies offering products (including traditional AST vendors and API testing and protection vendors), complicating evaluation and selection efforts.

User Recommendations

- Begin evaluation and selection efforts by focusing on the criticality of APIs; their security and business risks; and the technical requirements they pose.
- Examine the testing and discovery capabilities provided by existing tools in your application security portfolio. Many have added support for APIs, although actual tests may still focus primarily on traditional application vulnerabilities and lack support for API-specific vulnerabilities.
- Evaluate newer alternatives where gaps are found. They may also offer additional options, including audit of design-stage API specifications, discovery, vulnerability scans and threat protection.
- Complement automated testing tools with penetration testing services for comprehensive coverage as traditional AST is often unable to detect some common API-specific vulnerabilities.
- Evaluate the ability of a given tool to address multiple requirements. API security tool capabilities — including discovery, testing and threat protection — often overlap.

Sample Vendors

42Crunch; APIsec; Contrast Security; HCLSoftware; Noname Security; OpenText; PortSwigger; StackHawk; Synopsys; Veracode

Gartner Recommended Reading

API Security: What You Need to Do to Protect Your APIs

API Security Maturity Model

How to Deploy and Perform Application Security Testing

Research Index: Everything You Should Do to Address API Security

Critical Capabilities for Application Security Testing

Web App Client-Side Protection

Analysis By: Dionisio Zumerle

Benefit Rating: Low

Market Penetration: 5% to 20% of target audience

Maturity: Emerging

Definition:

Web app client-side protection is a security innovation that defends against application-level attacks that initiate on the client side rather than on the server side. A typical example of an attack would be a malicious script injection against client-side JavaScript, such as Magecart. Client-side security innovations monitor the activity of users and applications, and detect malicious actions and components.

Why This Is Important

Client-side attacks experienced in the wild have proven to be particularly impactful as they exploit the increasingly decentralized design of modern applications. In particular, single-page applications migrate the control and software logic on the client side, where it is exposed to attacks. For example, by injecting malicious scripts into JavaScript applications, attackers have lured thousands of visitors to banking and online commerce websites to hand over their credit card information.

Business Impact

Companies in industries such as financial services, ticketing and online retail have experienced a proliferation of fraudulent script injections into their web applications. These attacks steal customer payment information and expose organizations to significant fines. Organizations have turned to client-side security recently to protect their applications from these types of attacks.

Drivers

- In the past few years, there has been a rise in client-side attacks, such as those carried out by the Magecart groups. Online retail, airline and ticketing websites are the main targets for client-side attacks, where attackers inject malicious scripts that lure website visitors to hand over their credit card details.

- The effects of the global pandemic have prompted many retailers to speed up their digital transformation, shifting more business to online web applications, thus making them more profitable targets for client-side attacks.

- Recent regulatory guidance in the payments industry refers to the need for client-side protection capabilities.

- Despite lacking an established best practice to provide client-side protection, most approaches in this space focus on web JavaScript monitoring. After observing the web application, the client-side protection can start identifying abnormal or unsanctioned behavior and, depending on the settings, report on or block it.

- Content Security Policy (CSP) and Subresource Integrity (SRI) are standardized mechanisms used to increase security within browsers. They allow organizations to define lists of allowed webpage components and perform integrity checks on the client side. These security controls have been considered alternative techniques to JavaScript monitoring. They are emerging as add-ons for client-side protection tools.

- Cloud web application and API protection (WAAP) providers have been introducing client-side protection capabilities as part of their offerings.

Obstacles

- Client-side security is technologically complex. Many organizations will need time to reach the required maturity and understanding to conclude they must protect their client-side applications. At the same time, while client-side attacks remain relevant, there have been fewer client-side attacks recently after a period of intense activity.
- There is little data about the efficacy of client-side protection and possible business-disruption effects. Additionally, many script injections continue to target servers rather than clients, making client-side protection a partial solution to script injection threats.
- A variety of approaches have merits, but, at the moment, none provide an established standard that enterprises can follow.
- Three different entities in enterprises — development, security and line of business — are involved in the buying process. The lack of clear ownership of this technology slows down adoption.
- Concerns about performance impacts, possible customer privacy issues and false positives that block business traffic often steer enterprises away from client-side protection products.

User Recommendations

- Identify applications that might benefit from client-side security, such as public-facing JavaScript web applications that contain software logic on the client side. More broadly, single-page web applications and mobile web apps will also be good candidates to receive this sort of protection.
- Assess your incumbent application security vendors — for example, WAAP — and the self-defending capabilities they currently offer or plan to introduce in their roadmaps before looking at stand-alone products.
- Implement client-side security protection for critical web applications used to carry out bookings or transactions. Do so favoring approaches that monitor JavaScript and identify malicious, unsanctioned or abnormal behavior.

Sample Vendors

Akamai; Cloudflare; Ensighten; F5; Feroot Security; Imperva; Jscrambler; PerimeterX; Reflectiz; Source Defense

Gartner Recommended Reading

Critical Capabilities for Cloud Web Application and API Protection

Serverless Function Security

Analysis By: Charlie Winckless

Benefit Rating: Moderate

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Serverless function (also referred to as "function PaaS") security technologies are designed to address the unique security and compliance requirements of serverless function protection. Comprehensive solutions start with proactive vulnerability and configuration scanning in development, entitlement and access checking — typically combined with lightweight runtime protection and behavioral analysis.

Why This Is Important

Serverless functions are available in all hyperscale platforms, and may be included in cloud-native applications as a simple and scalable way to enable an event-driven microservices architecture. These services appeal to developers, but present risks from both vulnerable code and environment misconfiguration while being hard to protect with traditional controls due to their serverless nature.

Business Impact

By ensuring the security and compliance of the serverless functions they create, information security organizations can securely enable the developer-driven adoption of these technologies without slowing down the same developers. In the longer term, serverless functions have the potential to improve overall enterprise security profiles by migrating responsibility for significant elements of the attack surface to the hyperscaler, rather than the enterprise.

Drivers

- Driven by developers, adoption of serverless functions is increasing across hyperscalers.
- Serverless functions have a differentiated attack surface driving the need for security capabilities, such as software composition analysis, vulnerability scanning, API security, correct and compliant serverless PaaS configuration.
- When new types of attacks emerge against function PaaS, serverless function security is uniquely positioned to help organizations detect and respond to those attacks, as it provides visibility and security controls into the PaaS environment.
- Cloud permissions are extremely complex, and serverless function security allows for automatic detection and remediation of overly permissioned functions that increase risk.

Obstacles

- In most cases, information security is blind to the use of serverless functions and unaware of the risks they pose. Additionally, few attacks have been clearly documented on serverless code, meaning that any perceived risk is low for the effort and money invested.
- Serverless function security must have minimal friction for developers to avoid its adoption being disrupted by the developer community.
- At runtime, since serverless functions live for a matter of seconds or minutes, the need for additional runtime security other than monitoring is minimal. Very few options are available, short of injecting or wrapping serverless functions with runtime protection code.
- Serverless security tools are still maturing, and standards for secure deployment across multiple platforms are yet to be defined.

User Recommendations

- Engage with cloud-native development teams on the scope of serverless function usage and planned usages. Run a discovery project to see if serverless code is in use that you aren't aware of.
- Scan for vulnerabilities, vulnerable open-source code and misconfiguration automatically during development, as you would for any static application code.
- Require your cloud security posture management (CSPM) tool to provide risk visibility and configuration/permissions management of the entire IaaS configuration, including serverless.
- Adopt a least-privilege security posture, including serverless function permissions and network connectivity. Automate discovery of over-privileged use of serverless functions and reduce this to the least possible.
- Require an API gateway or event broker for invocation, providing a visibility and control point.
- Require your cloud workload protection platform (CWPP) or CSPM vendor to offer serverless function security and compliance capabilities — either now or as a roadmap item.

Sample Vendors

Amazon Web Services; Aqua Security; Check Point Software Technologies; Contrast Security; Palo Alto Networks; Rapid7; Snyk; Sysdig; Trend Micro

Gartner Recommended Reading

Market Guide for Cloud-Native Application Protection Platforms

5 Things You Must Absolutely Get Right for Secure IaaS and PaaS

How to Make Cloud More Secure Than Your Own Data Center

API Threat Protection

Analysis By: Mark O'Neill, Jeremy D'Hoinne

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

API threat protection is the defense of web APIs from exploits, abuse, access violations and denial of service (DoS) attacks. API gateways, web application and API protection (WAAP), and specialist API security tools provide API threat protection through a combination of content inspection of API parameters and payloads, traffic management, and traffic analysis for anomaly detection.

Why This Is Important

Applications use APIs to access data and to call application functionality. APIs are easy to expose, but difficult to defend. This creates a large and growing attack surface, leading to a growing number of publicized API attacks and breaches. Successful attacks on APIs result in data breaches, leading to loss of sensitive data as well as reputational harm.

Business Impact

Because APIs are typically used for access to data or application functionality, often linked to systems of record, the impact of an API breach can be substantial. Privacy regulations typically require reporting if private data is breached through an unsecure API. APIs are easily and intentionally programmable, so a vulnerability can leak large volumes of data. The challenge of distinguishing malicious access from valid access further complicates the task of securing APIs.

Drivers

- Publicized API breaches are driving awareness of the need for API threat protection.
- Financial services APIs are high-value by nature, and vulnerable to abuse and fraud.
- APIs are widely used by large language models (LLMs) for data access, leading to an increased need to protect that access to data.
- Some established vendors have begun to use machine learning to detect potentially harmful API usage patterns and thus protect APIs from threats. These include WAAP vendors.

Obstacles

- Many organizations lack visibility of their APIs, as many are used as part of web or mobile applications and not published directly. This means that a key requirement of API threat protection is API discovery. The main obstacle to API discovery is that the APIs used by an organization are typically dispersed across multiple platforms, including cloud services.
- Organizational ownership of API threat protection is a challenge. Whereas the security team, under a CISO, typically manages a WAF, API gateways are managed by API teams. This can lead to API threat protection being neglected due to a lack of expertise.
- Many API security issues are related to business logic. Protection against business logic threats is difficult to automate completely because the protection tool needs to understand the logic and identify unusual usage.
- Behavioral anomaly detection is, by nature, a generator of false positives. Despite the growing use of generative AI, some API threat detection tools currently require human intervention to process false positives.

User Recommendations

- Discover and categorize your APIs before implementing threat protection in runtime.
- Implement a layered API security model. At the outer (typically cloud-based) layer, use volumetric distributed denial of service (DDoS) protection and bot detection. Behind this layer, apply API-specific authentication, authorization and traffic management.
- Assess the API protection provided by your current WAF or API gateway. If it provides immature or insufficient API protection, then investigate API threat protection specialist vendors.
- Ensure that the API protection rules in your chosen API threat protection solution are adaptable, based on the nature of the API itself. Static rate limits or IP allow/block lists are rarely useful in production environments or at scale.

Sample Vendors

42Crunch; Akamai (Neosec); Cequence Security; FireTail; Imperva; Noname Security; Salt; Threat Hunter (China); ThreatX; Traceable

Gartner Recommended Reading

Research Index: Everything You Should Do to Address API Security

API Security Maturity Model

Innovation Insight for API Protection

Application Monitoring and Protection

Analysis By: Neil MacDonald

Benefit Rating: Moderate

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Application monitoring and protection is the convergence of security and operational application monitoring to consolidate approaches and simplify instrumentation. When combined, it can alert application owners to anomalous application behaviors indicative of a potential service failure caused by hardware or software issues or malicious actors. It could also take protective actions such as moving the workload, spinning up a new image, throttling a request and blocking a potential attack.

Why This Is Important

Over the next decade, operations monitoring, security monitoring and protection use cases will converge for cloud-native applications, driven by DevSecOps-style collaboration and integration. In most mature development organizations, the DevOps product teams or site reliability engineering teams will be responsible for service continuity regardless of whether the issue is a software bug, hardware failure or an attacker. Converged security and application performance monitoring will support this.

Business Impact

From the business unit and end-user perspectives, IT services delivery failures due to hardware failure or security attack are not separate problems. By combining performance and protection monitoring, IT organizations can improve application stability, performance and service quality. They should also avoid suboptimal results caused by a lack of coordination between potential performance and security processes, and reduce the total number of vendors and the licensing and support costs.

Drivers

- Increasingly, DevOps product owners are taking primary responsibility for their products' service levels, including availability and frontline security monitoring.
- Using one vendor for security monitoring and another vendor for operational monitoring duplicates efforts, increases complexity, and potentially inhibits the performance and stability of the application the products were intended to protect.
- Multiple monitoring vendors are pursuing this strategy of combining application performance and security monitoring capabilities for cloud-native applications and are bringing competitive offerings to the market.
- The adoption of managed container and serverless code services is increasing, and information security can't rely solely or even primarily on agent-based approaches for visibility.
- Container- and Kubernetes-based applications offer new approaches for instrumentation, including privileged containers, DaemonSets and sidecars.
- Initiatives such as OpenTelemetry, eBPF and WebAssembly in Envoy will help to consolidate monitoring approaches.

Obstacles

- The need for this type of monitoring is driven by two or three separate teams — security, application operations and infrastructure operations — which may have different vendor preferences and requirements as well as budgets.
- There is a reluctance to adopt any security monitoring by teams that are incorporating application performance monitoring (APM) solely for production visibility because of fears of performance and instability.
- Most operational monitoring vendors are immature in security capabilities, and most security monitoring vendors are immature in operational monitoring.
- Many organizations don't have the skills on their DevSecOps team to handle frontline security monitoring responsibility.
- The proliferation of languages and frameworks has made it difficult to pursue a single monitoring strategy.
- SaaS application monitoring requires new approaches. Visibility here requires more network performance and digital-experience-monitoring-type approaches.

User Recommendations

- Design a process for continuous IT services risk and operational monitoring by converging service security and operational monitoring responsibilities and giving this responsibility to the DevOps product teams.
- Establish a small working group comprising key IT operations and security personnel to initiate pilot projects for converged application monitoring and protection.
- Evaluate the pros and cons of alternative application security protection strategies — for example, by deploying application monitoring and protection in the network or in the cloud using APIs, or by instrumenting the application platform using agents or containers. Don't assume that application instrumentation is the best approach.
- Require application monitoring and protection vendors to provide attack detection insight (ideally, also providing attack blocking and prevention capabilities that are part of runtime application self-protection).

Sample Vendors

Cisco; Datadog; Dynatrace; Elastic; Fastly; GitLab; Kadiska; Splunk; Sumo Logic; Sysdig

Gartner Recommended Reading

How to Make Cloud More Secure Than Your Own Data Center

Critical Capabilities for Application Performance Monitoring and Observability

How to Select DevSecOps Tools for Secure Software Delivery

Application Performance Monitoring and Application Security Monitoring Are Converging

Threat Modeling Automation

Analysis By: Dale Gardner

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

Threat modeling automation tools automate the creation of security requirements and threat models. They can integrate with software development life cycle (SDLC) tools to manage requirements and perform validation. Threat modeling automation tools dynamically highlight potential security ramifications of functional requirements and recommend secure coding practices or architectural countermeasures.

Why This Is Important

Threat modeling and security requirements creation are key to efforts to create secure applications. Tools automate and facilitate these processes by shifting security further left, to the very start of the SDLC. Although they do not secure code, they make it easier to create secure code and application architectures. They also help ensure appropriate security requirements, in contrast to broad standards that inadequately secure high-risk apps while overburdening low-risk projects.

Business Impact

Threat modeling automation tools significantly decrease the effort required to create and maintain threat models, security requirements and risk assessments. This ensures security specifications that are specific to individual projects can be defined early, while costs and risks are low, rather than later. This offers significant benefits to multiple groups within an organization, including architects, developers, security teams and even business stakeholders.

Drivers

- Threat modeling is a best practice in application development. Threat modeling automation addresses a challenge that has constrained adoption, the overwhelmingly manual nature of the task. Automation allows threat modeling to proceed at the pace of development, allowing more organizations to adopt the practice.
- The ready availability and increasing sophistication of threat modeling automation tools enable individuals or small groups to carry out threat modeling and requirements generation in a fraction of the time and effort typically required. They can also make the task much more approachable for architectural and development staff who may lack training in application security concepts and requirements.
- Organizations of all kinds continue to struggle to create secure applications. Issues include the creation — and the detection and elimination — of inadvertent vulnerabilities, as well as essential design flaws, all of which leave applications vulnerable to attack. Threat modeling automation tools can help solve these problems. With relatively limited effort — especially compared with manual approaches — these tools can generate relevant security-related requirements aligned with the threat model and attack surface of applications. The data generated in the process can also guide triage and assessment of vulnerabilities discovered, based on their potential impact.
- Automation of threat modeling helps organizations incorporate the process into more rapid development styles, which would otherwise be extremely difficult.
- Threat modeling automation tools make it easier to incorporate specific requirements associated with compliance mandates, helping to ensure those requirements are addressed.

Obstacles

- Capabilities vary. Free and open-source tools enable easy adoption, but fall short when modeling more complex systems. This prompts consideration of commercial tools, though limitations constrain suitability for the most advanced users.
- Most organizations still focus on application security testing in establishing an application security program. These tools are essential, but fail to identify design flaws.
- The ability to accurately represent a rapidly changing application remains a weak spot of threat modeling automation tools. A threat model is only as good as its ability to provide an accurate representation of a system. Most of today's tools require user intervention to update models as applications change, which leads to abandonment. This is improving as vendors begin to link systems directly to cloud platforms or infrastructure as code files, ensuring changes are reflected automatically in the model, which will then automatically produce updated guidance.

User Recommendations

- Treat threat modeling, security requirements generation and enforcement as best practices within a secure SDLC model. Threat modeling automation tools can be used to help automate these tasks, while incorporating content knowledge from the tool vendors on emerging threats and security requirements.
- Take a risk-based approach, which these tools help enable, to align the level of effort involved in threat modeling with the risk posed by an application.
- Use these tools to automate what are otherwise manual or overlooked efforts. This ensures threat modeling and security requirements generation activities are incorporated into the SDLC process and development workflow. Consider integration of test cases to confirm that requirements are met.
- Train development, engineering, operations and architectural staff in the use and value of these tools. Encourage their use early and continuously in the development process and after deployment (to validate application threat protection efforts).

Sample Vendors

IriusRisk; Microsoft; OWASP Foundation; Security Compass; ThreatModeler

Gartner Recommended Reading

5 Frequently Asked Questions About Threat Modeling

12 Things to Get Right for Successful DevSecOps

Use Threat Modeling to Teach Secure Design (ADP)

Cloud WAAP

Analysis By: Aaron McQuaid, Rajpreet Kaur, Dale Koeppen

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Mature mainstream

Definition:

Cloud web application and API protection (WAAP) is a technology that mitigates runtime attacks exemplified by the Open Web Application Security Project (OWASP) Top 10 web application security vulnerabilities. Capabilities include web application firewall (WAF), distributed denial of service (DDoS) protection, bot management and API security. Innovation is driven by the need to defend against an expanding attack surface caused by growth in the number of web applications and APIs.

Why This Is Important

Organizations with web applications and APIs hosted on-premises or as infrastructure as a service (IaaS) often buy cloud WAAP solutions from security vendors to shield applications and APIs. They can be more flexible to deliver and manage than traditional virtual appliances, because they're easily deployed. Combined with continuous integration/delivery (CI/CD) pipelines, the proliferation of microservices architectures is causing API use to grow rapidly, increasing the need for WAAP.

Business Impact

Public web applications and APIs are high-risk targets. Modern applications are web-based and built with microservices architectures. APIs enable microservices to communicate with each other and with public-facing services, so it is important to protect them. Cloud WAAP solutions are easier to consume than their appliance counterparts. Cloud WAAP is highly scalable; it provides centralized management, centralized visibility and stronger integration with hyperscalers.

Drivers

- There is a proliferation of web-based applications and APIs. Modern software development has moved from monolithic application design toward microservices architectures. A microservices-based design decomposes a single monolithic application into several loosely coupled, independent software instances (typically containerized). These instances communicate with each other via APIs and externally via APIs or web-based protocols.
- There is evidence of an increasing number of Layer 7 application layer denial of service (DoS) attacks that require robust application layer security.
- Adversary groups continue to use and improve automated credential stuffing attacks. Bots are increasingly being leveraged in these attacks.
- The adoption of cloud-native architectures is shifting buyers from traditional appliance models to cloud-delivered services. The burden of deploying and managing a discrete set of appliances at every physical location is mitigated by adopting a cloud-delivered service with a common cloud-based management interface.

Obstacles

- **Privacy-related:** Compliance requirements can be issues for organizations. Some don't trust cloud decryption to log application traffic and host-related secrets. This is likely to become less pronounced as cloud adoption continues to gain acceptance.
- **Complexity:** Expanding features sets for WAAP are leading many organizations to adopt managed security services, which may increase costs.
- **Cost:** WAAP offerings that are part of an existing ADC might appear less expensive for organizations that don't want to redesign their solutions.
- **Application fit:** Organizations with on-premises applications might not see the value in cloud WAAP deployments or favor a unified management approach, where they use hosted virtual appliances to keep the same centralized console for on-premises and cloud-hosted applications.
- **Geographic presence:** Insufficient, regional point of presence (POP) density could lead to adoption of virtual or on-premises appliances.

User Recommendations

- Align your WAAP strategy with your future application architecture by adopting a cloud-first policy via the "follow the app" principle, when deciding from among an on-premises WAF appliance, a cloud WAAP or a distributed WAAP.
- Carefully evaluate the pros and cons of cloud WAAP. This includes simplicity of consumption, data privacy, DDoS protection, bot mitigation and API security, as well as deployment challenges, such as certificate management for transport layer security (TLS) inspection.
- Continue to improve your stance against bots and automated attacks by measuring the efficacy of existing controls and adding new techniques when needed.
- Implement products with automated API discovery and anomaly detection. Many WAAP solutions do not yet offer best-of-breed API security capabilities; compare them with offerings from dedicated API security vendors.
- Evaluate integrations with API gateways that help with API management when looking for consolidation of API management and API security.

Sample Vendors

Akamai; Barracuda; Cloudflare; F5; Fastly; Fortinet; Imperva; Radware; ThreatX

Gartner Recommended Reading

Magic Quadrant for Web Application Firewalls

Critical Capabilities for Cloud Web Application and API Protection

Service Mesh

Analysis By: Anne Thomas

Benefit Rating: Low

Market Penetration: 1% to 5% of target audience

Maturity: Adolescent

Definition:

A service mesh is a distributed computing middleware that manages communications between application services — typically within managed container systems. It provides lightweight mediation for service-to-service communications and supports functions such as authentication, authorization, encryption, service discovery, request routing, load balancing, self-healing recovery and service instrumentation.

Why This Is Important

A service mesh is lightweight middleware for managing and monitoring service-to-service (east-west) communications — especially among microservices running in container management systems, such as Kubernetes. It provides visibility into service interactions, enabling proactive operations and faster diagnostics. It automates complex communication concerns, thereby improving developer productivity and ensuring that certain standards and policies are enforced consistently across applications.

Business Impact

Service mesh is one of many management technologies that provide software infrastructure for distributed applications. Service meshes are most often used with services deployed in container management systems, such as Kubernetes. This type of middleware, along with other management and security middleware, helps provide a stable environment that supports “Day 2” operations of containerized workloads. However, the technology is complex and often unnecessary for smaller deployments.

Drivers

- Microservices and containers: Service mesh adoption is closely aligned with microservices architectures and container management systems like Kubernetes. Service mesh supports useful functionality in ephemeral environments, such as dynamic service discovery and mutual Transport Layer Security (mTLS) between services.
- Observability: As microservice deployments scale and grow more complex, DevOps teams need better ways to track operations, anticipate problems and trace errors. Service mesh automatically instruments the services and feeds logs to visualization dashboards.
- Resilience: A service mesh implements the various communication stability patterns (including retries, circuit breakers and bulkheads) that enable applications to be more self-healing.
- Bundled feature: Many container management systems now include a service mesh, inspiring DevOps teams to use it. The hyperscale cloud vendors provide a service mesh that is also integrated with their other cloud-native services.
- Federation: Independent vendors such as Buoyant, greymatter.io, HashiCorp, Kong and Solo provide service meshes that support multiple environments.

Obstacles

- Not necessary: Service mesh technology can be useful when deploying microservices in Kubernetes, but it's never required.
- Complexity: It's complex to use and administer, and there are increasing discussions on why not to use a service mesh in technology discussion groups and social media.
- Redundant functionality: Users are confused by the overlap in functionality among service meshes, ingress controllers, API gateways and other API proxies. Management and interoperability among these technologies is still nascent within the vendor community.
- Overhead: Service mesh technology consumes resources and typically adds overhead to the interactions it manages. Some vendors now support alternate architectures, such as a shared-agent model to reduce overhead, but this solution reduces the observability benefits.
- Competition with "free": Independent service mesh solutions face challenges from the availability of platform-integrated service meshes from the major cloud and platform providers.

User Recommendations

- Determine whether the value you might get from a service mesh in terms of improved security or observability is worth the increase in complexity and administration of the service mesh. A service mesh becomes more valuable as the number of service-to-service (east-west) interactions increases.
- Favor the service meshes that come integrated with your container management system unless you have a requirement to support a federated model.
- Reduce cross-team friction by assigning service mesh ownership to a cross-functional platform engineering team that solicits input and collaborates with networking, security and development teams.
- Accelerate knowledge transfer and consistent application of security policies by collaborating with I&O and security teams that manage existing API gateways and application delivery controllers.

Sample Vendors

Amazon Web Services; Ambient Mesh; Buoyant; Google; HashiCorp; Istio; Kong; Microsoft; Solo.io

Gartner Recommended Reading

How a Service Mesh Fits Into Your API Mediation Strategy

Application Shielding

Analysis By: Dionisio Zumerle

Benefit Rating: Moderate

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Application shielding serves to prevent and detect attacks such as tampering and reverse engineering. Application shielding is an in-app protection technology, meaning its capabilities are implemented directly within the application, rather than in-line or on the hosting system. Application shielding can be used for any type of application, but there is currently a particular focus on mobile apps.

Why This Is Important

Modern application architectures such as mobile move software logic to the front end, and place sensitive data on the devices. These architectures expose functionality that, unless shielded, can lead to attacks such as data leakage on the app or its back end, and fraudulent use or compromise of the user's device application functionality.

When the applications convey or store sensitive data, or enable payments, application shielding represents an important security measure.

Business Impact

Application shielding protects enterprise software and applications from cloning, fraud, intellectual property (IP) theft and other forms of abuse. It can also be used to enhance the user experience in certain sectors, including financial services and online retail. By hardening the application, an online retailer can minimize the restrictions and additional forms of verification requests (like step-up authentication) made to its customers.

Drivers

- Two broad families of functionality are being established as the basis of application shielding: hardening and anti-tampering. Hardening hinders the attacker from stealing information (such as IP or credentials) or cloning the application, by making reverse-engineering harder. Hardening includes techniques such as code obfuscation and white-box cryptography. Anti-tampering performs reconnaissance of the environment the application runs in to identify potential risks. Anti-tampering includes techniques such as emulation and debugging detection.
- Application shielding is suitable for applications that run on untrusted environments. Adoption is growing for consumer-facing mobile applications in industry verticals such as financial services, online retail, healthcare, insurance and automotive.
- Buyers increasingly ask for complete products. In addition to hardening and anti-tampering, buyers increasingly look for anti-malware capabilities and functionality that goes beyond shielding, such as multifactor authentication, to minimize the number of security products integrated in their application.
- Adoption is growing as enterprises and independent software vendors (ISVs) are becoming more aware of the availability of these solutions.

Obstacles

- Hardening technologies are mature, deriving from the protection of set-top boxes and digital media. However, they must be adapted to newer devices and operating systems, such as mobile and IoT devices, which makes them less mature.
- Protection techniques must keep pace with new and advanced attacks and are therefore in a constant state of evolution themselves.
- Application shielding is based on complex and research-intensive technology that many seasoned security and development practitioners are not particularly familiar with. Even if the risks may be clear, the techniques and efficacy to mitigate these risks may not be immediately apparent to end users.
- Application shielding products are perceived as costly by end-user buyers.

User Recommendations

- Identify high-value apps that store or access sensitive information (payment data or IP) and run in untrusted environments (e.g., on customer devices). Start with mobile devices, but also consider IoT and web applications with software logic on the client side.
- Prioritize the adoption of application shielding if your organization is in the financial services, e-commerce, insurance or healthcare industry verticals.
- Opt for postcoding on the compiled binary when you do not have access to the source code or do not want to impact the development life cycle. Favor implementation during development when you require complex functionality such as white-box cryptography and you have access to the source code.

Sample Vendors

Appdome; Build38; Digital.ai; Guardsquare; OneSpan; preEmptive Technologies; Promon; Verimatrix; V-Key; Zimperium

Climbing the Slope

Bot Management

Analysis By: Akif Khan, Jeremy D'Hoinne

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

Bot management solutions can detect and respond to automated applications and bots that interact with websites, mobile apps and APIs. They can block unwanted automated activity while ensuring that human users and legitimate bots have access as intended by the business. Assessing the humanity of interactions is typically achieved through a layered approach of examining network signals, devices and session attributes, and potentially forcing the user to solve a CAPTCHA.

Why This Is Important

Bot traffic continues to rise along with increasing bot sophistication. Attackers use bots to automate their attacks on online assets, including scraping of competitive data, inventory grabbing and credential stuffing, and full or partial denial of service — whether intentional or not. The rise of “hu-bot,” a combination of specialized bots with human-operated, fraud-farm services, requires ever-improving detection and response solutions.

Business Impact

Although initially used for protecting large B2C web applications, bot management has become increasingly relevant to any B2C or B2B public-facing endpoint where malicious automated traffic can directly impact business outcomes. Malicious bots can negatively impact user experience (UX) by causing slower page load times, hoarding inventory and facilitating account takeover via credential stuffing or credential cracking.

Drivers

- Need to prevent large-scale, low-sophistication automated attacks from basic bots.
- Leaders’ increasing recognition that bot management crosses multiple use cases and business units makes these capabilities more sought-after.

- Legacy CAPTCHA solutions can be solved by bots or hu-bots that delegate CAPTCHA solving to human farms, especially to assess humanity against large numbers of user volume. Also, poorly conceived CAPTCHA solutions can degrade UX. This drives further interest in bot management solutions that have minimal impact on UX.
- Bot management solutions can identify malicious bots and preserve the UX of legitimate application users and authorized bots. Enterprise-approved bots can include search engine crawlers, automated testing and web application monitoring software, robotic process automation (RPA) or other machine-to-machine (M2M) communication. M2M communication includes bots that run in environments like Microsoft Teams, Slack, Stride and some bot marketplaces.
- Bot mitigation is being increasingly bundled into content delivery network (CDN) solutions, thus lowering the barriers for organizations to explore the benefits of bot management.

Obstacles

- The fear of blocking a single legitimate user is often higher than the perception of the damage being caused by malicious bots.
- Concerns persist that many bot management vendors use CAPTCHA too frequently, often to test for false positives. This is usually less than 1% of traffic, and CAPTCHA solutions are proprietary and relatively optimized. However, organizations still fear the friction inherent in CAPTCHA challenges.
- Bundled offerings from WAAP providers create difficulties when they justify bot management deployment only for some applications or to protect certain key application features, such as login pages.
- The growing perception among the most targeted B2C mega brands is that no single bot management solution can mitigate all bot attacks. Accordingly, some organizations are now relying on multiple vendors to mitigate the broadest spectrum of attacks. This results in skepticism about bot management efficacy and value.

User Recommendations

- Assess the threat level to business assets and the impact caused by bots, starting with the most business-critical and most exposed web applications and APIs.

- Evaluate the capabilities of bot management solutions that come with your WAAP or CDN platform. If these are sufficient, they represent a quick win in terms of implementation ease and vendor rationalization. If the chosen bot management solution doesn't meet your requirements, evaluate stand-alone solutions.
- Select solutions based on their ability to detect malicious bots via various techniques rather than relying primarily on reputation controls to detect "known-bad" sources, such as IP reputation or attack techniques — for example, signatures.
- Gauge how dependent a solution is on CAPTCHA challenges to assess false positives and what that CAPTCHA experience looks like. Ensure internal stakeholders focused on UX are comfortable with the approach. Reject any approach relying on mandatory CAPTCHA solving.

Sample Vendors

Akamai; Arkose Labs; Cloudflare; DataDome; F5; hCaptcha; HUMAN; Imperva; Netacea; Radware

Gartner Recommended Reading

Critical Capabilities for Cloud Web Application and API Protection

Market Guide for Online Fraud Detection

Secure Coding Training

Analysis By: Aaron Lord, Manjunath Bhat

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Adolescent

Definition:

Secure coding training raises awareness of the impact and prevention of vulnerabilities in source code. Developers are trained in secure coding practices for specific coding languages and frameworks using different methods like just-in-time (JIT) training, gamified lessons, videos, workshops and challenges.

Why This Is Important

Seventy-five percent of software engineering leaders surveyed in the 2022 Gartner Software Engineering Leaders Role Survey stated that application security skills are a pain point in their organization. Secure coding skills evolve over time and are lacking from general software engineering education. Ensuring that software engineers are up-to-date with the latest secure coding skills will raise awareness of the risk of introducing vulnerabilities.

Business Impact

Any organization that writes software needs to be concerned about security and reliability. Software engineering leaders and practitioners need to implement secure coding training with regular cadence. Secure coding training takes multiple approaches to learning, each with their own pros and cons. Secure coding training vendors are leaning heavily on the workshop and gamified approach, although that will be less effective for more established organizations with seasoned developers.

Drivers

The creation of secure software requires software engineers have the skills to apply secure coding practices:

- Application security is a growing concern for organizations that create software.
- Developers lacking security understanding produce unsecure applications by introducing vulnerable code, third-party components and infrastructure configurations.
- Staying up-to-date with secure coding practices will help reduce the risk of data loss and breaches.
- Organizations that implement security champions or coaches need to enable advanced learning for these individuals.

Obstacles

Enabling learning for software engineers, especially at scale, has a number of challenges:

- Just like any approach to teaching, types of secure coding training work better or worse for certain individuals (presentation, tests, workshops, etc.).
- Software engineer leaders are not investing in secure coding training and offering no incentives for its completion.

- When training is assigned annually, dropping knowledge retention between training sessions may lead to security mistakes.
- Offerings and pricing models for secure coding training can vary wildly, so it can be difficult to understand what is needed.
- Secure coding training has yet to be effective at a large scale across multiple regions.

User Recommendations

Take into account multiple factors for your engineering organization that apply to secure coding training needs to:

- Utilize a security champions program to push training and security awareness from within.
- Ensure that a secure coding training vendor offers training that matches the organization's technology stacks, languages and frameworks.
- It is preferable to use training that has a mix of approaches instead of just one (presentations, workshops, challenges, gamification, etc.).
- Ensure secure coding training offerings come in multiple languages that can support engineers across multiple regions.
- Integrate secure coding training into software development life cycle (SDLC) tooling for JIT training to maintain knowledge retention over time.

Sample Vendors

Avatao; Codebashing; Immersive Labs; Secure Code Warrior; SecureFlag; Security Compass; Security Journey; Snyk; Synopsys; Veracode

Gartner Recommended Reading

How to Select DevSecOps Tools for Secure Software Delivery

Infographic: Build These Essential Skills to Secure Modern Software Development

Develop Your Technical Skills Using Online Learning Platforms

Mobile Application Security Testing

Analysis By: Dionisio Zumerle

Benefit Rating: Moderate

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

Mobile application security testing (AST) identifies and helps remediate vulnerabilities within mobile apps for iOS and Android devices. Mobile AST analyzes source, byte or binary code and observes or attacks mobile apps to identify coding, design, packaging, deployment, and runtime conditions that introduce security vulnerabilities.

Why This Is Important

Mobile applications are central to a company's digital transformation. Ensuring these apps do not present vulnerabilities that can be exploited is essential to enable this transformative process. Mobile AST largely uses similar techniques to traditional AST adapted for the mobile device environment and the more agile development processes that come with it.

Business Impact

Depending on an organization's structure, mobile AST may be used by security and application development teams, and sometimes directly by the line-of-business departments. Even though every organization that delivers mobile applications should perform security testing, regulated and other high-security industries, such as financial services, healthcare and online retail, have a higher urgency to adopt mobile AST.

Drivers

- Organizations with mobile apps require AST that is adapted for mobile environments. Mobile AST tools must support mobile-specific programming languages and frameworks. They also need to identify, in addition to traditional application vulnerabilities, mobile-specific ones such as unsecured storage and hard-coded credentials.

- Often organizations that already employ AST for their web applications need a faster and possibly more cost-efficient alternative specifically for mobile AST.

- The OWASP Mobile Application Security Verification Standard (MASVS) has introduced a granular set of requirements for mobile application security testing, that is helping both mobile AST vendors and practitioners better frame mobile AST needs.

- Conducting mobile AST can contribute to streamlining the approval and publication in commercial app stores easier. For example, through the App Defense Alliance (ADA) Mobile Application Security Assessment (MASA), Google recognizes mobile apps that have undergone independent security testing against MASVS Level 1 requirements.

- Open-source software (OSS) components and software development kits (SDKs) are frequently used with mobile applications, and they are also frequently the cause of vulnerabilities for mobile apps. As software bill of materials (SBOMs) usage starts to emerge, the ability for mobile AST to assess third-party code becomes increasingly important.

- While mobile AST products are mainly used with homegrown apps, some enterprises are using them for application vetting. This allows organizations to identify leaky or malicious apps. This use case is also often addressed by mobile threat defense product offerings.

Obstacles

- The techniques used in mobile AST are the same mature static AST (SAST), dynamic AST (DAST), software composition analysis (SCA) and interactive AST (IAST) techniques that have been used for years to test web-based applications. However, when applied to mobile apps, testing must be adapted to identify client-side code vulnerabilities. Mobile platforms are still evolving, albeit more slowly than in the past; therefore, mobile AST has not yet reached full maturity.
- Many organizations have less-advanced application security programs and are not yet testing mobile app code.
- In many cases, the main source of risk is in the back end, making it less critical to include mobile app code in the application security program.

User Recommendations

- Perform mobile AST, whether your mobile apps are workforce- or consumer-facing, starting with the most critical ones — applications that run in untrusted environments, with software logic on the client side and applications that have transactional or IP value.
- Investigate whether the mobile AST capabilities it provides, directly or via a partnership, can suffice, if you already have an AST vendor. Typical AST selection considerations still apply, but for mobile specifically, the OWASP MASVS can be a practical reference guide to assess a solution's technical capabilities and coverage.
- Look into dedicated mobile AST vendor offerings if you do not have an incumbent AST solution or need a specialized or lightweight and speedier mobile AST product.

Sample Vendors

Appknox; Data Theorem; eShard; Guardsquare; HCLSoftware; ImmuniWeb; Lookout; NowSecure; Quokka; Zimperium

Gartner Recommended Reading

Critical Capabilities for Application Security Testing

Entering the Plateau

Full Life Cycle API Management

Analysis By: Shameen Pillai

Benefit Rating: High

Market Penetration: More than 50% of target audience

Maturity: Early mainstream

Definition:

Full life cycle API management involves the planning, design, implementation, testing, publication, operation, consumption, versioning and retirement of APIs. API management tools help creating API ecosystems, publishing and operating APIs, and collecting analytics for monitoring and business value reporting. These capabilities are typically packaged as a combination of a developer portal, API gateway, API design, development and testing tools as well as policy management and analytics.

Why This Is Important

APIs are widely used as the primary choice to connect systems, applications and devices; integrate business partners, and build modular and composable software architectures. The use of APIs as digital products — monetized directly or indirectly — is also on the rise. Digital transformation initiatives have accelerated the need for creation, management, operations and security of APIs. They have also made full life cycle API management an essential foundational capability for every organization.

Business Impact

Full life cycle API management provides the framework and tools necessary to manage and govern APIs that are foundational elements of multiexperience applications, composable architectures and key enablers of digital transformations. It enables the creation of API products, which may be directly or indirectly monetized, while its security features serve to protect organizations from the business risk related to API breaches.

Drivers

- Organizations are facing an explosion of APIs, stemming from the need to connect systems, applications, devices and other businesses. The use of APIs in internal, external, B2B, private and public sharing of data is driving up the need to manage and govern APIs using full life cycle API management.

- APIs that package data, services and insights are increasingly being treated as products that are monetized and enable platform business models. Full life cycle API management provides the tooling to treat APIs as products.
- Digital transformation drives an increased use of APIs, which in turn increases the demand for full life cycle API management.
- Organizations looking for growth acceleration and business resilience are adopting API-based architectures to eliminate or modernize their legacy and monolithic applications.
- Developer mind share for APIs is growing. Newer approaches to event-based APIs, design innovations and modeling approaches — such as GraphQL — are driving interest in full life cycle API management, leading to more experimentation and growth.
- Cloud adoption and cloud-native architectural approaches to computing (including serverless computing) are increasing the use of APIs in software engineering architectures, especially in the context of microservices, service mesh and serverless.
- Greater awareness of and increasing significance to API security are driving organizations to take charge of discovering and managing APIs. This often starts with operational management of existing APIs.
- Regulated, industry-specific initiatives in the financial, insurance and healthcare industries continue to provide a steady demand. However, use of APIs is spreading across most industries (especially retail, technology and telecom, manufacturing), increasing the demand.
- Commoditization and widespread availability of API gateways as part of cloud services, security solutions and other bundled software applications are increasing the need for distributed API management that involves multiple gateways, including those from multiple vendors.
- Rising influence of generative AI and large language models in software engineering is likely to increase and reshape the need for APIs.

Obstacles

- A lack of commitment to adequate organizational governance processes hinders the adoption of full life cycle API management. This can be due to a lack of skills or know-how, or due to putting too much focus on bureaucratic approaches rather than federated and automated governance approaches.

- A lack of strategic focus on business value (quantifiable business growth or operational efficiency) and too much focus on technical use cases can disengage business users and sponsors. This is particularly apparent in cases where API programs fail to deliver the promised return on investment.
- Traditional, single-gateway approaches to API management no longer fit well with modern, distributed API management approaches.
- A partial or full set of embedded API management capabilities provided by vendors in other markets — such as application development, integration platforms, security solutions and B2B offerings — can partially meet the use cases for API management and shrink the market opportunities.

User Recommendations

- Use full life cycle API management to power your API strategy and address both technical and business requirements for APIs. Select offerings that can address both well beyond the first year.
- Treat APIs as products managed by API product managers in a federated API platform team. Adopt business metrics for API products.
- Choose a functionally broad API management solution that supports modern API trends, including microservices, multigateway and multicloud architectures. Ensure that the chosen solution covers the entire API life cycle.
- Use full life cycle API management to enable governance of all APIs, including third-party (private or public) APIs that you consume.
- Ask full life cycle API management vendors about their support for automation, especially for API validation, testing and DevOps integration. Also ask about support for low-footprint and third-party API gateways.
- Full life cycle API management solutions are suitable for implementing a single-vendor strategy as opposed to a best-fit tooling approach for different API life stages. Ensure the choice aligns with your organization's goals.

Sample Vendors

Axway; Google; IBM; Kong; Microsoft; Salesforce (MuleSoft); Software AG

Gartner Recommended Reading

Magic Quadrant for Full Life Cycle API Management

The Evolving Role of the API Product Manager in Digital Product Management

How to Use KPIs to Measure the Business Value of APIs

API Security: What You Need to Do to Protect Your APIs

Reference Model for API Management Solutions

Software Composition Analysis

Analysis By: Dale Gardner

Benefit Rating: Transformational

Market Penetration: More than 50% of target audience

Maturity: Mature mainstream

Definition:

Software composition analysis (SCA) products are specialized application security tools that detect open-source software (OSS) and third-party components known to have security vulnerabilities, and identify potentially adverse licensing and supply chain risks. It is an essential element in strategies to ensure an organization's software supply chain includes secure and trusted components and, therefore, that the strategy aids in secure application development and assembly.

Why This Is Important

SCA promotes the use of OSS in application development by identifying known vulnerabilities, ensuring components are properly licensed, and advancing trust in the software supply chain. It is an essential activity, given the ubiquity of OSS in applications and the potential for significant risk.

Business Impact

- SCA must be considered a foundational element of application security testing to identify known vulnerabilities and potential supply chain risks in open-source packages and other artifacts.
- Its primary users are application development and security teams, tasked with ensuring the integrity of open-source components (and, less frequently, commercial software) in development.
- License assessments — once the primary use case for SCA — remain an important function for legal and sourcing groups.

Drivers

- SCA has advanced to mainstream usage due to increased usage of open source, and through its role in supporting software supply chain security. Supply chain risks — including the potential for malicious code, poorly supported packages, and other attacks — have emerged as a primary driver for SCA. Organizations are placing an increased focus on addressing these risks in response to high-profile incidents and expanding global regulatory and market mandates.
- An underlying driver for the growing importance of SCA is the ubiquitous use of OSS in application development. The prevalence of OSS, both as individual packages and within container images, has led to SCA becoming a key control for continuous integration/continuous delivery (CI/CD) pipelines and containerized environments.
- Repeated instances of high-impact vulnerabilities in OSS have become pervasive because of the underlying components' broad use across organizations. This, along with ever-present supply chain attacks, demonstrates the need to better understand the risks posed by OSS and commercial software packages.
- New requirements around the ability to address supply chain risks, and to enhance developer productivity when remediating issues, have prompted organizations to revisit their SCA tooling. More effective SCA tools now provide guidance on a preferred update version — balancing stability, remediation of flaws and potential adverse impacts on the functionality of existing code. In some cases, tools have begun to incorporate assessments of operational risk, in an effort to help prevent supply chain attacks. There are also varied levels of support for software bill of materials (SBOM) analysis and generation — another rapidly emerging requirement.
- Meeting compliance requirements and ethical and legal standards is a growing concern for organizations, and SCA technology can help ensure developers are meeting these requirements. Tools help reduce the likelihood of unwanted or unapproved code that creates risks to an organization's intellectual property.

Obstacles

- Tools can report a large number of issues, creating friction with developers. The ability to confirm usage of affected code or its exploitability has become a required feature, but is not universally available.
- Although SCA tools are often seen as a complete solution to the prevention of supply chain attacks, in practice, they offer only a partial defense. With few exceptions, they don't test code to identify new issues, nor do they automatically remediate issues. To establish a more complete solution, users will need to augment SCA itself, and consider other approaches to address tasks such as protecting the build process and artifacts.
- Commercial software libraries are often outside the scope of SCA, but pose many of the same risks, leaving a protection gap.
- Legal and procurement teams are traditional users of SCA tools, although products are increasingly optimized for use by development teams. Organizations must evaluate the ability of tools to support multiple use cases when evaluating solutions.

User Recommendations

- Implement SCA technologies as a key ingredient of every software security program. SCA tools can be acquired from open-source projects, as stand-alone solutions or as a component of a dedicated AST tool.
- Ensure legal experts analyze SCA license warnings to address the legal issues stemming from IP ownership or license obligations.
- Proactively use SCA tools on a regular basis to audit OSS repositories to ensure the software used by the enterprise meets organizational standards. Consider establishing a trusted repository containing approved code.
- Incorporate SCA tools within DevSecOps workflows, where scanning can be automated as part of the rapid development processes to identify issues and track usage of components.
- Use SCA tools in conjunction with a formal corporate IP strategy that has established clear responsibility across the company.
- Favor SCA tools with strong supply chain risk management support, including the ability to consume and generate SBOMs.

Sample Vendors

Checkmarx; GitHub; Mend.io; Phylum; Revenera; ReversingLabs; Snyk; Sonatype; Synopsys; Veracode

Gartner Recommended Reading

Market Guide for Software Composition Analysis

Critical Capabilities for Application Security Testing

Magic Quadrant for Application Security Testing

How to Manage Open-Source Software Risks Using Software Composition Analysis

DevSecOps

Analysis By: Neil MacDonald, Mark Horvath

Benefit Rating: Transformational

Market Penetration: More than 50% of target audience

Maturity: Mature mainstream

Definition:

DevSecOps is the integration and automation of security and compliance testing into agile IT and DevOps development pipelines, as seamlessly and transparently as possible, without reducing the agility or speed of developers or requiring them to leave their development toolchain. Ideally, offerings provide security visibility and protection at runtime as well.

Why This Is Important

DevSecOps offers a means of effectively integrating security into the development process, in a way that eliminates or reduces friction between security and development. The goal is to pragmatically achieve a secure, workable software development life cycle (SDLC) supporting rapid development. DevSecOps has become a mainstream development practice, although the specifics can vary between organizations based on their technology and the maturity of their development processes.

Business Impact

The goal of DevSecOps is to speed up development without compromising on security and compliance. Furthermore, the externalization of security policy enables business units and security organizations to define and prioritize policy guardrails and lets developers focus on application functionalities. Policy-driven automation of security infrastructure improves compliance, the quality of security enforcement and developer efficiency, as well as overall IT effectiveness.

Drivers

- Adoption of DevOps, and other rapid development practices, requires security and compliance testing that can keep up with the rapid pace of development.
- DevSecOps offerings are applied as early as possible in the development process, whereas traditional application security testing (AST) tools associated with older development models are applied late in the development cycle, frustrating developers and business stakeholders.
- Testing results need to be integrated into the development process in ways that complement developers' existing workflows and toolsets, and not require them to learn skills unrelated to their goals.
- The use of open source has greatly increased the risk of the inadvertent use of known vulnerable components and frameworks by developers.

Obstacles

- Incorrectly implemented, siloed and cumbersome security testing is the antithesis of DevOps. Due to this, developers believe security testing tools are slowing them down.
- Developers don't understand the vulnerabilities their coding introduces.
- Developers don't want to leave their development (continuous integration/continuous delivery [CI/CD]) pipeline to perform tests or to view the results of security and compliance testing tools.
- Historically, static application security testing (SAST) and dynamic application security testing (DAST) tools have been plagued with false positives or vague information, hence frustrating developers.
- The diversity of developer tools used in a modern CI/CD pipeline will complicate the seamless integration of DevSecOps offerings.

User Recommendations

- "Shift left" and make security testing tools and processes available earlier in the development process.
- Prioritize the identification of open-source software (OSS) components and vulnerabilities in development (referred to as software composition analysis).
- Opt for automated tools with fast turnaround times, with a goal of reducing false positives and focusing developers on the highest-confidence and most-critical vulnerabilities first.
- Ask vendors to support out-of-the-box integration with common development tools and support full API enablement of their offerings for automation.
- Evaluate emerging cloud native application protection platform (CNAPP) offerings for technical control implementation.
- Require security controls to understand and apply security policies in container- and Kubernetes-based environments.
- Favor offerings that can link scanning in development to correct configuration, visibility and protection at runtime.

Sample Vendors

Apiiro; Aqua Security; Contrast Security; Dazz; Lacework; Palo Alto Networks; Qwiet AI; Snyk; Sonatype; Wiz

Gartner Recommended Reading

How to Select DevSecOps Tools for Secure Software Delivery

Market Guide for Cloud-Native Application Protection Platforms

Magic Quadrant for Application Security Testing

12 Things to Get Right for Successful DevSecOps

How to Manage Open-Source Software Risks Using Software Composition Analysis

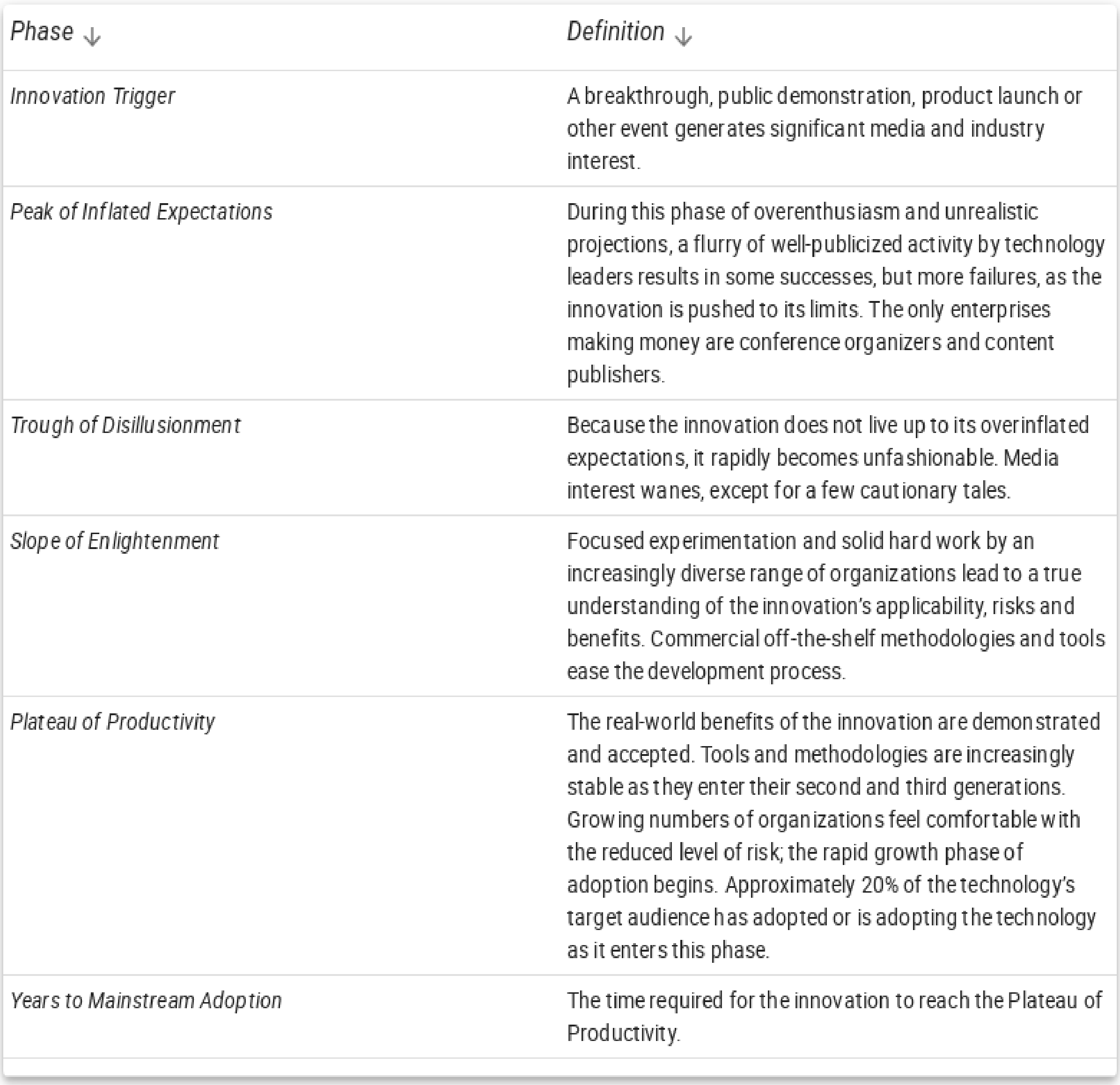
Appendixes

See the previous Hype Cycle: Hype Cycle for Application Security, 2022

Hype Cycle Phases, Benefit Ratings and Maturity Levels

Table 2: Hype Cycle Phases

(Enlarged table in Appendix)



Phase ↓	*Definition* ↓
Innovation Trigger	A breakthrough, public demonstration, product launch or other event generates significant media and industry interest.
Peak of Inflated Expectations	During this phase of overenthusiasm and unrealistic projections, a flurry of well-publicized activity by technology leaders results in some successes, but more failures, as the innovation is pushed to its limits. The only enterprises making money are conference organizers and content publishers.
Trough of Disillusionment	Because the innovation does not live up to its overinflated expectations, it rapidly becomes unfashionable. Media interest wanes, except for a few cautionary tales.
Slope of Enlightenment	Focused experimentation and solid hard work by an increasingly diverse range of organizations lead to a true understanding of the innovation's applicability, risks and benefits. Commercial off-the-shelf methodologies and tools ease the development process.
Plateau of Productivity	The real-world benefits of the innovation are demonstrated and accepted. Tools and methodologies are increasingly stable as they enter their second and third generations. Growing numbers of organizations feel comfortable with the reduced level of risk; the rapid growth phase of adoption begins. Approximately 20% of the technology's target audience has adopted or is adopting the technology as it enters this phase.
Years to Mainstream Adoption	The time required for the innovation to reach the Plateau of Productivity.

Source: Gartner (July 2023)

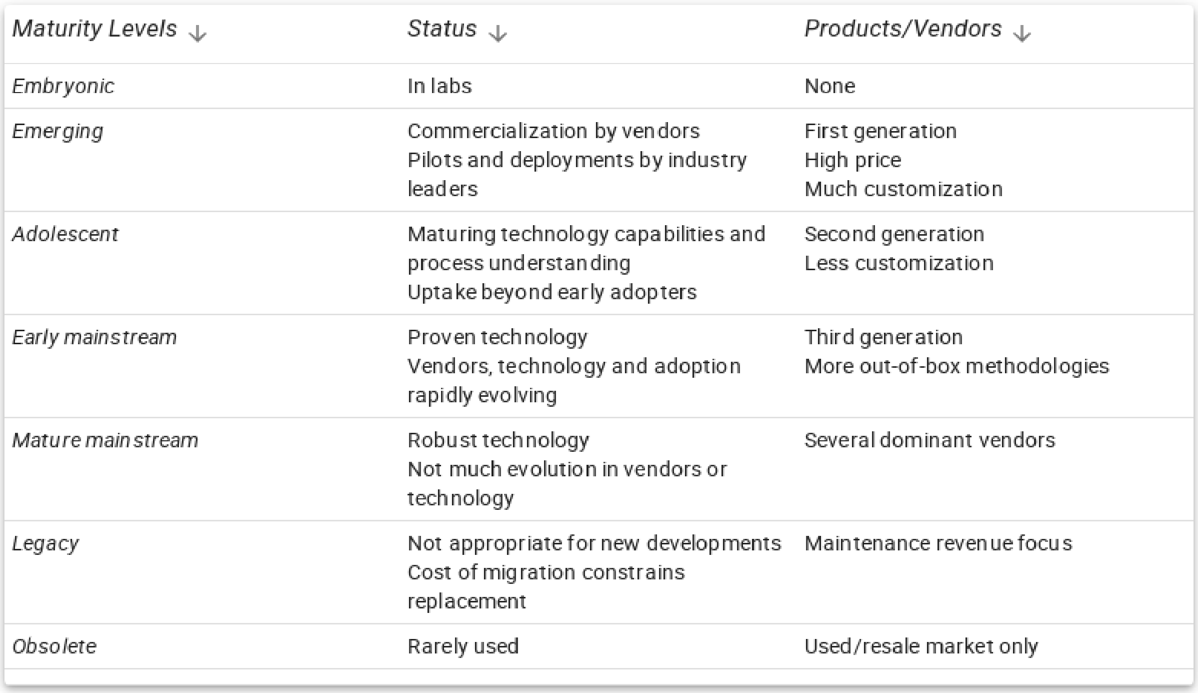
Table 3: Benefit Ratings

Benefit Rating ↓	*Definition* ↓
Transformational	Enables new ways of doing business across industries that will result in major shifts in industry dynamics
High	Enables new ways of performing horizontal or vertical processes that will result in significantly increased revenue or cost savings for an enterprise
Moderate	Provides incremental improvements to established processes that will result in increased revenue or cost savings for an enterprise
Low	Slightly improves processes (for example, improved user experience) that will be difficult to translate into increased revenue or cost savings

Source: Gartner (July 2023)

Table 4: Maturity Levels

(Enlarged table in Appendix)



Maturity Levels ↓	Status ↓	Products/Vendors ↓
Embryonic	In labs	None
Emerging	Commercialization by vendors Pilots and deployments by industry leaders	First generation High price Much customization
Adolescent	Maturing technology capabilities and process understanding Uptake beyond early adopters	Second generation Less customization
Early mainstream	Proven technology Vendors, technology and adoption rapidly evolving	Third generation More out-of-box methodologies
Mature mainstream	Robust technology Not much evolution in vendors or technology	Several dominant vendors
Legacy	Not appropriate for new developments Cost of migration constrains replacement	Maintenance revenue focus
Obsolete	Rarely used	Used/resale market only

Source: Gartner (July 2023)

Acronym Key and Glossary Terms

ASRTM	application security requirements and threat management
CASB	cloud access security broker
CI/CD	continuous integration/continuous delivery
EAM	externalized authorization management
SASE	secure access service edge
SAST	static application security testing
SRM	security and risk management
WAAP	web application and API protection
WAF	web application firewall
XDR	extended detection and response

Evidence

[1] Organizations that implemented changes to cybersecurity policies in the last 24 months evolved the role of the CISO from the cybersecurity controls owner to a facilitator and business value creator. This finding comes from the 2022 Gartner Shifting Cybersecurity Operating Model Survey.

The 2022 Gartner Shifting Cybersecurity Operating Model Survey was conducted to determine the impact of the changing technology governance environment on the security operating model at the macro level. The survey was conducted online from October through November 2022 among 462 respondents from North America (n = 148), Europe (n = 216), Latin America (n = 33) and Asia/Pacific (n = 61). Respondents were required to be cybersecurity or information security leaders.

Results of this survey do not represent global findings or the market as a whole, but reflect the sentiments of the respondents and companies surveyed.

[2] Application security is experiencing 24.7% growth in 2023. For more information, see Gartner Identifies Three Factors Influencing Growth in Security Spending.

[3] The 2023 Verizon Data Breach Investigations Report (form submission required) once again found basic web application attacks to be a major attack vector. Also, attacks using exploited vulnerabilities have doubled compared to the previous year.

[4] According to the Sonataype 8th Annual State of the Software Supply Chain Security report, supply chain attacks increased 742% on average per year from 2019 to 2022.

Document Revision History

Hype Cycle for Application Security, 2022 - 11 July 2022

Hype Cycle for Application Security, 2021 - 12 July 2021

Hype Cycle for Application Security, 2020 - 27 July 2020

Hype Cycle for Application Security, 2019 - 30 July 2019

Hype Cycle for Application Security, 2018 - 27 July 2018

Hype Cycle for Application Security, 2017 - 28 July 2017

Hype Cycle for Application Security, 2016 - 13 July 2016

Hype Cycle for Application Security, 2015 - 9 July 2015

Hype Cycle for Application Security, 2014 - 28 July 2014

Hype Cycle for Application Security, 2013 - 25 July 2013

Hype Cycle for Application Security, 2012 - 20 July 2012

Recommended by the Author

Some documents may not be available as part of your current Gartner subscription.

Understanding Gartner's Hype Cycles

Tool: Create Your Own Hype Cycle With Gartner's Hype Cycle Builder

Magic Quadrant for Application Security Testing

Quick Answer: How to Make Microsoft 365 Copilot Enterprise-Ready From a Security and Risk Perspective

Innovation Insight for Application Security Posture Management

Innovation Insight for API Protection

How to Select DevSecOps Tools for Secure Software Delivery

Gartner

Table 1: Priority Matrix for Application Security, 2023

Benefit	*Years to Mainstream Adoption*			
↓	***Less Than 2 Years*** ↓	***2 - 5 Years*** ↓	***5 - 10 Years*** ↓	***More Than 10 Years*** ↓
Transformational	DevSecOps Software Composition Analysis	Application Security Posture Management Code Security Assistants Security Service Edge	Generative Cybersecurity AI Software Supply Chain Security	
High	Bot Management Full Life Cycle API Management	API Security Testing API Threat Protection Cloud WAAP Policy as Code Secure Coding Training Software Bill of Materials Threat Modeling Automation	Cloud-Native Application Protection Platforms Crypto-Agility	
Moderate		Application Shielding Mobile Application Security Testing Serverless Function Security	Application Monitoring and Protection Chaos Engineering Kubernetes Security Penetration Testing as a Service SaaS Security Posture Management	

Benefit	*Years to Mainstream Adoption*			
↓	***Less Than 2 Years*** ↓	***2 - 5 Years*** ↓	***5 - 10 Years*** ↓	***More Than 10 Years*** ↓
Low		Service Mesh		

Source: Gartner (July 2023)

Table 2: Hype Cycle Phases

Phase ↓	*Definition* ↓
Innovation Trigger	A breakthrough, public demonstration, product launch or other event generates significant media and industry interest.
Peak of Inflated Expectations	During this phase of overenthusiasm and unrealistic projections, a flurry of well-publicized activity by technology leaders results in some successes, but more failures, as the innovation is pushed to its limits. The only enterprises making money are conference organizers and content publishers.
Trough of Disillusionment	Because the innovation does not live up to its overinflated expectations, it rapidly becomes unfashionable. Media interest wanes, except for a few cautionary tales.
Slope of Enlightenment	Focused experimentation and solid hard work by an increasingly diverse range of organizations lead to a true understanding of the innovation's applicability, risks and benefits. Commercial off-the-shelf methodologies and tools ease the development process.
Plateau of Productivity	The real-world benefits of the innovation are demonstrated and accepted. Tools and methodologies are increasingly stable as they enter their second and third generations. Growing numbers of organizations feel comfortable with the reduced level of risk; the rapid growth phase of adoption begins. Approximately 20% of the technology's target audience has adopted or is adopting the technology as it enters this phase.
Years to Mainstream Adoption	The time required for the innovation to reach the Plateau of Productivity.

Phase ↓	*Definition* ↓

Source: Gartner (July 2023)

Table 3: Benefit Ratings

Benefit Rating ↓	*Definition* ↓
Transformational	Enables new ways of doing business across industries that will result in major shifts in industry dynamics
High	Enables new ways of performing horizontal or vertical processes that will result in significantly increased revenue or cost savings for an enterprise
Moderate	Provides incremental improvements to established processes that will result in increased revenue or cost savings for an enterprise
Low	Slightly improves processes (for example, improved user experience) that will be difficult to translate into increased revenue or cost savings

Source: Gartner (July 2023)

Table 4: Maturity Levels

Maturity Levels ↓	*Status* ↓	*Products/Vendors* ↓
Embryonic	In labs	None
Emerging	Commercialization by vendors Pilots and deployments by industry leaders	First generation High price Much customization
Adolescent	Maturing technology capabilities and process understanding Uptake beyond early adopters	Second generation Less customization
Early mainstream	Proven technology Vendors, technology and adoption rapidly evolving	Third generation More out-of-box methodologies
Mature mainstream	Robust technology Not much evolution in vendors or technology	Several dominant vendors
Legacy	Not appropriate for new developments Cost of migration constrains replacement	Maintenance revenue focus
Obsolete	Rarely used	Used/resale market only

Source: Gartner (July 2023)