Publication date:
17 Dec 2020
Author:
Rik Turner
Principal Analyst

Omdia Market Radar for Next-Generation Application Security: Runtime

Brought to you by Informa Tech

Summary

Catalyst

Web application security has multiple dimensions, both in the development pipeline and at runtime. In recent years Omdia has perceived a trend for providers of one of these dimensions to add several others, sometimes through internal development but frequently via acquisition. Some even cross the pipeline/runtime divide and offer security across both environments.

While all this activity allows vendors to boast "one-stop shop" status, however, there is a deeper significance here, namely the ability to amass a body of threat intelligence to inform multiple strands of an application security offering. And as ever more private applications (i.e., ones that face not the general public but rather an organization's employees and business partners) are relocated to the cloud and are accessed over the internet instead of a private WAN link, these next-generation application security (NGAS) portfolios gain even greater importance.

Omdia view

Defending web applications has never been more important. Not only have ever more commercial interactions been driven online by the COVID-19 pandemic, but enterprises are increasingly adopting cloud computing, whether it be the software-as-a-service (SaaS) variant that serves all sizes of companies or the infrastructure- or platform-as-a-service (IaaS and PaaS) flavors that larger enterprises embrace as they gain more confidence in the cloud.

All the applications sitting in these cloud environments are, of course, accessed remotely using either private WAN links or, increasingly, the public internet. As the web option for access grows, access becomes more convenient, but the apps themselves become more exposed to cyberattacks.

It is no coincidence that the range of application security technologies has expanded in recent years as attackers have grown more sophisticated in their modus operandi. Nor is it any surprise that some of the larger vendors have been acquiring specialist startups to broaden their portfolios and, ultimately, to offer a one-stop-shop approach to application security.

That said, there must be more to the appeal of such offerings than the sort of attraction a supermarket has in comparison with a specialist store: these vendors are also accumulating threat intelligence across all their different application security services, pooling it at the backend, and in the case of the most advanced providers, aggregating it with third-party threat intelligence and running analytics on it to detect emerging threats, adopting defensive strategies across their portfolio to protect their customers' apps better and faster.

Key messages

- Distributed-denial-of-service (DDoS) mitigation and web application firewall (WAF) are the cornerstones of NGAS.
- Bot management and API security are now just as important.
- Runtime application self-protection (RASP) has not fulfilled its early promise.

Recommendations

Recommendations for enterprises

Weigh the advantages of a broad service portfolio versus a specialist

If, like many enterprises, you are making an increasing proportion of your application infrastructure, including your "private" apps, available over the public internet and are considering how best to protect it, there are broadly speaking two routes open to you. You can either invest in multiple platforms/services from specialists in, say, bot management or API security, deploying them alongside one another for a comprehensive application security infrastructure, or you can opt to take all such services from a single provider.

Both approaches have their pros and cons. Clearly, getting all your application security requirements from a single vendor is easier from a procurement perspective, and it gives you "one throat to choke" if there is a failure in your defenses. However, the specialist brings a laser focus to a particular problem area and can, therefore, be expected to react quickly to an emerging new type of threat within it, whereas with the best will in the world, a more generalist vendor with a broader portfolio has to juggle more priorities in product development and so may take longer to respond. On the other hand, a company offering multiple application security services will have a more comprehensive view of the overall threat landscape and should be expected to bring that breadth of knowledge to each of its services.

Ask your NGAS provider about keeping up with emerging threat vectors

Clearly, the threat landscape is continually evolving, and security vendors large and small scramble to stay abreast of the latest developments. While the smaller, more focused specialists tend to do this via in-house development of new capabilities, larger, more diversified vendors often resort to acquisitions to bring in new functionality quickly and in an "oven-ready" manner (e.g., Akamai bought ChameleonX to have an offering in web supply chain compromise, or WSCC). If you see new types of threats against your apps emerging and are taking a slew of NGAS services from a single provider, quiz it on how it intends to respond. If it is too slow, seek out a specialist, deploy its technology, then encourage the larger player to buy it.

Recommendations for vendors

Invest in customer education initiatives

Application security is a complex market segment that is still in rapid evolution as threat actors develop new and innovative ways to overwhelm, tamper with, or hoodwink an application. There is a good chance that even customers that are taking a range of NGAS services from you will not necessarily be aware of new and emerging threats and thus will not know what else they need in order to defend themselves. Vendors in this market need to host webinars, write blog posts, and offer regular news on these topics (e.g., 42Crunch owns and produces the website apisecurity.io, which has an option to sign up to a weekly newsletter). Such initiatives serve as valuable information sources for customers and the market in general while also burnishing your image as a thought leader.

DDoS mitigation and WAF are the cornerstones of NGAS

Life online drove the need for WAFs and DDoS mitigation

The first two staple technologies developed to protect web applications were WAFs and DDoS mitigation platforms. WAFs emerged in the late 1990s as e-commerce was taking off as a means of doing business and was attracting the attention of threat actors. Web server attacks became increasingly common as cybercriminals engaged in exploits such as SQL injection and cross-site scripting, and WAFs emerged as a specific form of application firewall devoted to stopping them.

Denial-of-service (DoS) attacks, and their more evolved descendant, DDoS attacks, similarly date from the last decade of the 20th century, with the very first attack thought to have taken place in 1996. Again, the incentive for mounting them grew as e-commerce became a vital component of companies' go-to-market approach, and e-government evolved as a delivery mechanism for public services.

Their objective was simple: to overwhelm a victim's internet-facing infrastructure with a flood of traffic, such as requests, whose sheer volume makes it impossible for the recipient systems to address. Think of a small rural post office suddenly faced with the incoming mail for an entire metropolis. DDoS attacks developed shortly thereafter, launching the attack traffic from multiple machines instead of a single server.

It was quickly recognized that this type of attack warranted a dedicated response in the form of mitigation software, instantiated in on-premises hardware appliances located in a "demilitarized zone" (DMZ). The leading vendors in that era were Arbor Networks, Radware, and F5.

WAF and DDoS mitigation move into the cloud

For different reasons, over the last decade both WAF and DDoS mitigation technology moved increasingly into the cloud to be delivered as services. In WAF's case, it was because of the sheer complexity and labor-intensive nature of managing a device, keeping up with evolution in the threat landscape, and adjusting the ruleset in the WAF to address those changes. As the rate at which threats evolved increased, it became virtually unworkable for most enterprises to operate their own WAF, so a managed service, delivered by a company with a team dedicated to keeping up with the changing landscape and updating the rules accordingly, became the logical option.

As for DDoS, it was the sheer volume of the attacks that drove the development of mitigation services delivered from the cloud. As narrowband connectivity gave way to broadband, and broadband to fiber, the bandwidth available for launching attacks mushroomed. At the same time, attackers grew more adept at marshaling and controlling armies of compromised servers (the so-called bot networks) from which to launch huge attacks. The era of volumetric DDoS attacks had arrived, and the easiest way to address the massive flood of bad traffic they unleashed at their targets was to soak it up on a network with vast bandwidth, that is, a service provider network instead of a corporate WAN.

State-of-the-art DDoS protection services nowadays come in different flavors:

- Always on, whereby a customer's traffic is permanently routed through the service provider's network, and protection from DDoS attacks is thereby continuous and instantaneous
- On demand, a hybrid approach in which local mitigation technology on the customer's premises handles all the smaller attacks such as the "low and slow" ones that do not rely on volume but escalates the response action to a cloud service, to which it then routes all traffic, as soon as it detects that a volumetric attack is underway. The challenge here is the length of the cutover between on-premises and cloud-based mitigation.

While the latter approach inherently sounds inferior on account of the time lapse involved in the switch from one mode to the other, it also presents a number of advantages in terms not only of cost but also of efficiency, in that addressing all types of DDoS attack with an always-on service will, for the low-bandwidth attacks, be like trying to swat a fly with a blunderbuss, when a simple flyswatter would have been the requisite tool. Consequently, most DDoS service providers offer permutations of both flavors of service.

Bot and API security are now just as important

Bots fall into three categories

While WAFs and DDoS mitigation were moving into the cloud, another trend over the last decade has been the increased number and variety of bots as the technology underpinning them has grown more sophisticated and the companies using them have gained greater expertise. There are now, broadly speaking, three classes of bot:

- Good bots come to a website for valid reasons and should be treated accordingly. A classic example is Googlebot, the web crawler technology Google uses to collect information to build a searchable index for the Google Search engine. Since all websites want to improve their positioning in any Google searches, it behooves them to expedite Googlebot's experience on the site well.
- "Gray" bots are not malicious and can actually be useful but will need to be deprioritized at moments of high traffic volumes. For instance, if a site resells hotel rooms or airline tickets as an additional route to market for hoteliers or airlines, it will need to visit the providers' websites via a bot to check availability and block-book at certain times of the day. However, when those websites are themselves receiving visits from humans who are potential customers, their traffic should receive priority over the reseller's bot, whose traffic should be throttled back accordingly.
- Bad or malicious bots are designed to carry out everything from credit or gift card fraud, through credential stuffing to account takeover and inventory hoarding. These bots obviously must be blocked at all times.

This variation in the bot landscape led, in the 2010s, to the development of bot management or bot security platforms.

The API economy now also requires dedicated security

After bot management, the next requirement that emerged in application security was for technology to secure application programming interfaces (APIs). These are customer-facing customizable software interfaces that enable once-separate software components to communicate by overcoming the inherent incompatibilities created by differing software platforms.

APIs bring a new level of connectivity and data sharing to multiple applications, regardless of their platforms, data structures, and underlying technologies. So powerful has their attraction been that it is now common to hear talk of the "API economy," meaning, at the highest level, the exchange of value between consumers and providers through APIs. However, with the increased use of APIs for interconnectivity between applications comes an increase in an enterprise's attack surface, because the API itself becomes an attack vector, making API security a must-have.

Initially, it was thought that this could be provided by companies that developed API management software (i.e., the platforms that controlled the creation and publication of APIs), and indeed, several of the so-called API gateway vendors did boast of their security capabilities. However, it soon became clear that their security capabilities were limited to controlling which developers had access to the API in the development pipeline, what they could and could not do with that code, and when their access should be rescinded.

While this is a necessary feature for API gateways, it does not constitute a comprehensive API security platform. Technology is also required to guarantee that bogus calls cannot manipulate an API into providing access to sensitive data. Other forms of web security are not up to the task: for example, headless browser detection plays a large role in identifying hostile bots within website traffic, but for incoming API traffic, there is no browser environment to detect.

Thus API security has become a discipline in its own right. The capabilities of platforms in this segment include

- API schema ingestion, validation, and enforcement
- Reverse-engineering prevention
- Dynamic and adaptive traffic recognition to reshape the security posture in response to changes in traffic structure: as an API evolves, a platform needs to create new rulesets automatically, which administrators can then accept, reject, or modify as needed
- Encryption and continuous authentication of all communications between an API and an endpoint (typically, the URL of a server or service)

Runtime application self-protection has not fulfilled its early promise

RASP enjoyed considerable hype circa 2014

Another approach to runtime security that was much lauded through the first half of the 2010s was RASP.

This is an approach to application security that incorporates security into the running application by instrumenting the server on which it runs, adding client software to analyze both the app's behavior and the context of that behavior, effectively detecting and blocking attacks by taking advantage of information from inside the app itself.

RASP seeks to improve the security of software by monitoring its inputs and blocking those that could allow attacks while protecting the runtime environment from unwanted changes and tampering. Established RASP vendors include Imperva (which acquired Prevoty), which is featured in this report; Signal Sciences (now part of Fastly and also profiled in this report); Micro Focus; and Sqreen. Its advocates argue that RASP is uniquely well suited for catching and handling injection attacks such as SQL injection, LDAP injection, and command-line interface (CLI) injection, because it can understand the commands involved and distinguish ordinary, safe sequences from those that may contain additional, out-of-scope instructions or requests.

RASP can be implemented as a module to run alongside and in conjunction with a program's code, libraries, and system calls. It can also be combined with dynamic application security testing (DAST) technology to produce what is known as interactive application security testing (IAST). DAST is, of course, part of pipeline NGAS and, as such, will be covered in a subsequent report from Omdia, as will IAST.

The configuration information that RASP works on is independent of the applications it protects and can be continuously updated or modified to keep up with current threats and vulnerabilities. RASP technology does not need to understand or patch existing or suspected vulnerabilities in application code, nor does it need to locate and identify such things. Rather, it provides a virtual patch against vulnerability by blocking malicious or suspect inputs and preventing applications from producing unwanted or unsafe outputs or behaviors. RASP also handles all common application protocols, including Ajax, XML, HTTP, HTTPS, JSON, REST, and Simple Object Access Protocol.

A trough of despondency followed initial enthusiasm

Despite all these benefits, however, it is fair to say that RASP has had its issues, not least of which is the performance hit that adding instrumentation to code at runtime inevitably entails. RASP vendors have certainly been keenly aware of this issue from the outset and have striven to improve the situation, to the point where many of them now claim only negligible impact on the app’s performance from their software. That said, RASP’s reputation with potential customers may have been irrevocably damaged by any early teething troubles.

Another issue with RASP is cost. Inevitably, it involves licensing costs for using third-party add-ins but also recurring costs to keep the RASP configuration up to date so as to deal with current threats and vulnerabilities.

RASP redux?

While RASP still has its proponents, including some of the companies in this report, it is notable that several of the companies that first emerged with the technology as their calling card have now moved on, mentioning it as one of the tools in their toolbox but by no means the only one, and certainly not leading with it in their marketing.

We have, nonetheless, included RASP as one of the weapons defenders can and should consider for inclusion in their arsenal. It is definitely a useful technology, even though its ability to contend for key capability status may still be questionable. Indeed, as this report was approaching publication, a new RASP player entered the fray in the form of Dynatrace. The application performance management and AIOps vendor created out of the Compuware business has made its first foray into the security market, adding a RASP add-on to its Software Intelligence Platform product called the Dynatrace Application Security Module.

In other words, RASP should not be written off as a clunky or failed technology experiment just yet, and Omdia plans a deeper dive into this market segment during 2021.

Further requirements continue to emerge

Web supply chain compromise protection is the latest

The four distinct app security capabilities listed above (i.e., WAF, DDoS mitigation, bot security, and API security) have all gravitated toward the cloud and gone from being licensed software to being delivered as services along the way. Beyond that, there has been considerable M&A activity in this area, enabling vendors to assemble comprehensive portfolios of runtime security for web applications that frequently nowadays boast all four of these services.

That said, the tech world and the threat landscape do not stand still: new attack vectors are continually emerging. The last couple of years have seen the growth in so-called script attacks on enterprises' web supply chains, a vector Omdia calls web supply chain compromise (WSCC).

Such attacks leverage the fact that, as the world of e-commerce has grown and gained in sophistication, it has become increasingly common for companies to invoke code from third and even fourth parties (the suppliers of services to their suppliers of services) on their webpages for purposes of payments, performance analytics, and supplementary information provision.

While this trend increases the functionality of a given website and can enhance customer experience, it also represents a new threat vector, because attackers can compromise this content, often by injecting malicious JavaScript, and use it to steal sensitive data such as credit card details.

WSCC attacks came into existence in 2014, but it was four years later in 2018 that they really hit the headlines. This was due to a couple of high-profile exploits against targets Ticketmaster and British Airways, with the latter being fined $240 million by the UK Information Commissioner's Office for its failure to protect the credit card details of half a million of its customers. The name behind a lot of these attacks is Magecart, which is, in fact, a consortium of hacker groups, six being particularly prominent. They gained their name from the fact that they target in particular the Magento online shopping cart system.

Akamai, Imperva, and PerimeterX are among the vendors profiled in this report that already have WSCC protection in their service portfolio.

The Open Web Application Security Project top 10 for web and API

As for the kinds of attacks against which runtime NGAS is deployed to defend applications, since 2003 the Open Web Application Security Project (OWASP), a nonprofit, collaborative online community, issues an annual top 10 list of web application security risks. Here is the most recent.

Top 10 web application security risks

- **Injection**. Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
- **Broken authentication**. Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
- **Sensitive data exposure**. Many web applications and APIs do not properly protect sensitive data such as financial and healthcare details and personally identifiable information (PII). Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.
- **XML external entities (XXE)**. Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and DoS attacks.
- **Broken access control**. Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data: they may access other users' accounts, view sensitive files, modify other users' data, change access rights, and so on.

- **Security misconfiguration**. Security misconfiguration is the most commonly seen issue. It is often a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, or verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.
- **Cross-site scripting (XSS)**. XSS flaws occur whenever an application includes untrusted data in a new webpage without proper validation or escaping, or it updates an existing webpage with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser that can hijack user sessions, deface websites, or redirect the user to malicious sites.
- **Insecure deserialization**. Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks including replay attacks, injection attacks, and privilege escalation attacks.
- **Using components with known vulnerabilities**. Components such as libraries, frameworks, and other software modules run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
- **Insufficient logging and monitoring**. Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allow attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, and breaches are typically detected by external parties rather than by internal processes or monitoring.

Furthermore, in 2019, as attacks against APIs started to proliferate, OWASP began to compile a separate list for APIs. It issued its first list that year.

OWASP API security top 10 vulnerabilities 2019

- **API1:2019 — broken object-level authorization.** Attackers substitute the ID of their own resource in the API call with an ID of a resource belonging to another user. The lack of proper authorization checks allows attackers to access the specified resource. This attack is also known as IDOR (insecure direct object reference).
- **API2:2019 — broken authentication.** Poorly implemented API authentication allows attackers to assume other users' identities.
- **API3:2019 — excessive data exposure.** An API exposes more data than the client legitimately needs, relying on the client to do the filtering. If attackers go directly to the API, they have it all.
- **API4:2019 — lack of resources and rate limiting.** An API is not protected against an excessive amount of calls or payload sizes. Attackers can use this for DoS and authentication flaws such as brute-force attacks.
- **API5:2019 — broken function level authorization.** An API relies on the client to use user-level or admin-level APIs as appropriate. Attackers figure out the "hidden" admin API methods and invoke them directly.

- **API6:2019 — mass assignment.** An API takes data that a client provides and stores it without proper filtering for white-listed properties. Attackers can try to guess object properties or provide additional object properties in their requests, read the documentation, or check out API endpoints for clues to where to find the openings to modify properties they are not supposed to on the data objects stored in the backend.
- **API7:2019 — security misconfiguration.** Poor configuration of the API servers allows attackers to exploit them.
- **API8:2019 — injection.** Attackers construct API calls that include SQL, NoSQL, LDAP, OS, or other commands that the API or the backend behind it blindly execute.
- **API9:2019 — improper assets management.** Attackers find nonproduction versions of the API (for example, staging, testing, beta, or earlier versions) that are not as well protected as the production API and use those to launch their attacks.
- **API10:2019 — insufficient logging and monitoring.** A lack of proper logging, monitoring, and alerting allows attacks and attackers to go unnoticed.

Key capabilities

Clearly, one way of looking at the vendors featured in this report is to consider the breadth of their offering, that is, whether they include DDoS mitigation, WAF, bot management, API security, RASP, and WSCC. However, that breadth alone should not be the criterion for where each vendor ranks in this emerging market segment. For this report, therefore, we have selected a series of other criteria against which to assess the vendors, covering areas such as their technology's usability and the individual vendor's ability to deliver on its potential.

Criteria

The criteria Omdia has used in scoring the various vendors of runtime NGAS services are as follows:

- **Configuration and customization.** Here we consider how easy it is for the customer to configure the service to meet its requirement and thus also how flexible it is.
- **Discovery and visibility.** This is where we gauge the technology's ability to identify all the assets it needs to inspect, which is a key consideration when it comes to shadow APIs, for instance. We also score the services on the degree to which this discovery process can be automated.
- **Attack detection.** This criterion considers whether the service uses rules, heuristics, machine learning (ML), behavioral analysis, or a combination thereof, and how comprehensive we feel it is as a result.
- **Scope of protection.** This looks at what percentage of the top 10 API security vulnerabilities and risks, as published annually by OWASP, are covered, as well as whether the vendor's portfolio spans the entire set of runtime requirements and even whether it extends beyond, into pipeline security (which some do).

- **Capacity to execute.** Here we look at a provider's scale and capabilities, not only in terms of how many points of presence (PoPs) they have on their global network from which to deliver services but also at the size and scope of their partner ecosystem, the geographies covered, and so on.
- **Visualization, analysis, and reporting.** We consider not only the ease of use for the security professionals tasked with protecting the applications but also the service's ability to produce reports that can be understood by nontechnical people higher up in an organization, such as at the board level.
- **Development lifecycle fit.** Although this report focuses on the runtime aspects of NGAS, there is increasingly a "shift left" in the approach to embedding application security earlier in the development process. This is especially true in containerization and in DevOps and Agile development and deployment models. So this category looks at integration with development tools and methods and at the capabilities to embed runtime application security policies during the development process.

Future market development

Omdia expects further M&A activity in the NGAS sector as more vendors seek to broaden their portfolios and as new requirements continue to emerge. Like other areas of security, NGAS is characterized by the frequent emergence of new startups as the challenges of application security continue to evolve, and that trend is sure to continue, creating opportunities for larger players to expand their offerings through acquisitions.

Vendor landscape

Profiled players

The primary criterion for a vendor's inclusion in this report is that it should be a provider of all, or at least some, of the key aspects of runtime NGAS.

That is not to say that it cannot also offer some aspect of pipeline NGAS, as indeed a couple of vendors in this report do, but its offering should be primarily designed to defend applications when they are actually in production. In the same way, one or two of the vendors that will appear in the next report, that is, the one covering pipeline NGAS, also have runtime capabilities, but Omdia opted to feature them in the pipeline report because the bulk of their business is in that use case.

Some of the vendors in this report fit into the description of niche players, in that they came into existence specifically to address only one of the requirements of runtime NGAS and have usually not been around long enough to expand significantly beyond that. This is particularly the case in API security, since it is a more recent development than DDoS mitigation or WAFs.

Through no fault of their own, these vendors tend to appear as *specialists* in the report, for the simple reason that they lack the breadth of portfolio of either the *leaders* or the *challengers*. We have included them, however, on account of the compelling nature of the technology they do offer, in the expectation that they will either expand their offerings into other areas of runtime NGAS or will be acquired by larger players looking to bolster their own portfolios.

On the Radar

42Crunch (Omdia recommendation: Specialist)

Vendor 42Crunch provides API security, both at runtime and in the continuous integration / continuous delivery (CI/CD) pipeline. Runtime API security requires technology beyond an API gateway or a WAF, and 42Crunch delivers such a capability. Omdia expects to see it grow its share of this emerging market, particularly because it can also address the requirement for security in the development pipeline.

Why put 42Crunch on your radar?

The ability of 42Crunch to secure both the CI/CD pipeline and the runtime environment makes it a compelling candidate for any API security project. The development of the local agent for internal APIs, meanwhile, expands its dynamic scanning capabilities even further.

Highlights

API security has evolved as a capability in its own right rather than an adjunct to API management platforms or WAFs, which was how the tech industry initially sought to meet the requirement. It is significant, in this context, that 42Crunch has founders from both those sectors.

API management platforms (a.k.a. gateways) treated security as an access problem pure and simple, defining which developer could access which assets and monitoring to see when inappropriate access was taking place. WAFs, on the other hand, addressed the issue with an API firewall, ruled by a deny list of prohibited connections and patterns to block. However, neither approach addressed some of the most widely exploited API vulnerabilities, such as broken object-level authorization, whereby an attacker manipulates the identity of an object contained in a request, which can result in unauthorized access to sensitive data. There was clearly a need for specific technology for API security, which is why companies such as 42Crunch have come into existence in recent years.

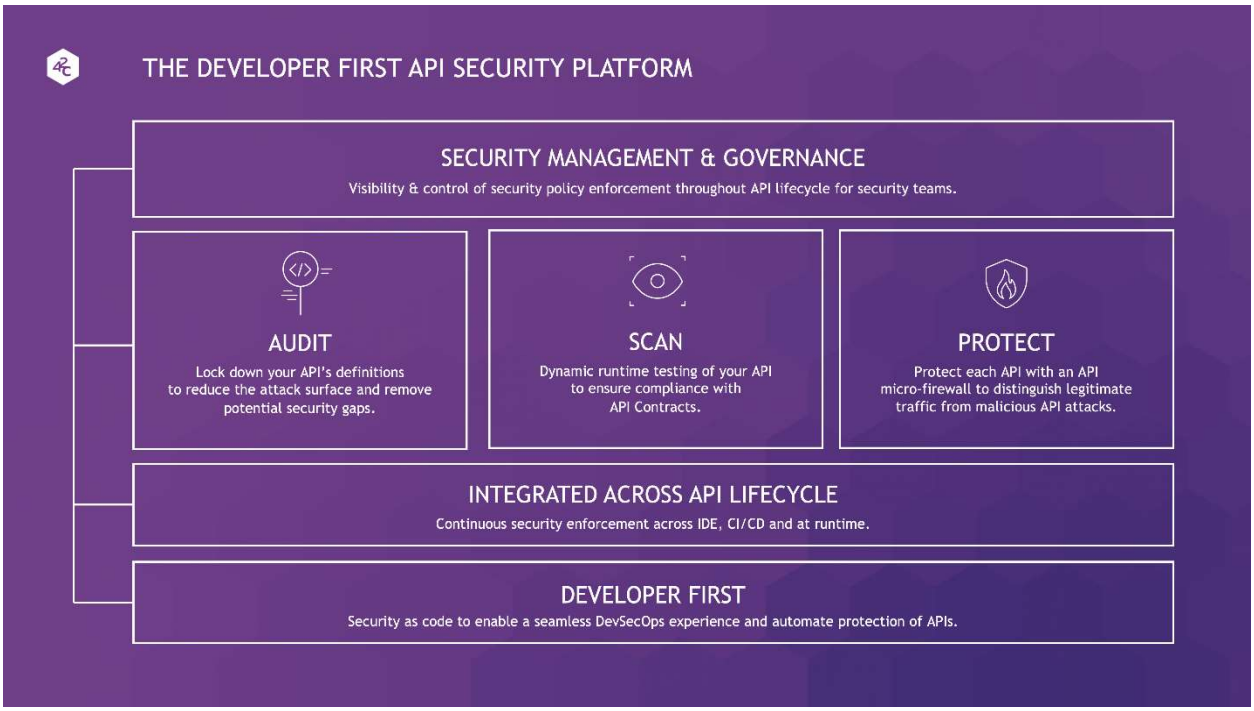
The vendor delivers three capabilities in its software, the first two in the development pipeline, the third when the API is in operation (i.e., in production):

- Static analysis, in which an OpenAPI (a.k.a. Swagger) file is inspected to detect any issues with the API definition itself
- Dynamic analysis, which finds vulnerabilities by performing attacks on the API rather than by inspecting the code
- Protection, which is 42Crunch's API firewall, designed to protect the API from attacks at runtime

All three features use the API "contract" (the definition of the API, written in accordance with the Linux Foundation's OpenAPI standard) making the technology deterministic in that something either complies with the contract and is thus allowed or does not comply and is blocked.

42Crunch's latest update to its platform includes enterprise single sign-on, command-line interface, an on-premises scan agent, and an extended range of security policies, including JSON Web Token (JWT), rate limiting and security headers.

Figure 1: 42Crunch API Security Platform



Source: 42Crunch

Background

CEO Jacques Declas, CTO Isabelle Mauny, and chief architect Philippe Leothaud founded 42Crunch in 2016.

Declas previously held senior VP roles at API security vendors Forum Systems and Vordel (now part of Axway) as well as at Intel (which bought API management firm Mashery in 2013 and sold it to Tibco in 2015). Mauny spent most of her career at IBM across a variety of technical roles before joining Vordel in 2009 and open source API management company WSO2 in 2012. Leothaud was previously CTO at BeeWare (now part of DenyAll) then principal architect at Vordel.

The vendor completed a private funding round for an undisclosed sum in February 2017 and is considering a further round later this year.

Current position

The vendor's technology is delivered in a mixture of SaaS and on-premises modes:

- The static component is suited to a cloud-based delivery model, since the API contract can be submitted to a cloud service for comparison with the industry security best practices for APIs. Some very risk-intolerant customers such as financial institutions shun a SaaS service where their API definitions would be in a shared environment, however, and for them 42Crunch can deploy the platform as a dedicated instance of its technology.
- The dynamic scanning part of the technology can also be a SaaS service, provided the API is readily accessible from outside the organization that owns it. Internal APIs, on the other hand, such as those found in staging environments or in the development lab, require the scanner itself to be on the customer's premises. However, 42Crunch says this is not a problem for customers using dedicated deployments in its network. For SaaS customers wishing to scan internal APIs, the vendor

currently offers a local agent, which has access to the contract, generates tests, and runs them locally, sending the result of the test to 42Crunch's cloud backend for analysis and incorporation into a report to the customer.

- Protection cannot be a cloud-based service on account of the additional latency that routing all API traffic through it would entail. Instead, the API firewall is deployed in the line of traffic, either as a Docker sidecar in microservices environments or as software added into an existing API gateway, if the customer has one.

In addition, 42Crunch provides plug-ins to popular IDEs (such as Visual Studio Code, IntelliJ, and Eclipse) and CI/CD platforms (including Azure DevOps, Jenkins, BitBucket, Bamboo, and Azure Pipelines), allowing integration of its tests and protection updates directly into tooling already in use by its customers.

An important development for 42Crunch on the pipeline security side of things came in October this year, when the vendor announced that it was one of Microsoft's launch partners for the GitHub Code Scanning alerts facility. In other words, 42Crunch's REST API Static Security Testing product was made available as a GitHub Action, so developers can include REST API OpenAPI / Swagger definitions within static security tests, get their APIs analyzed for security flaws on each code change in the repository or code submission, and see all identified issues right in the GitHub security center. All issues come with the 42Crunch knowledge base articles. Customers can see where the flaw is in their code, read about the potential exploit scenario, and see 42Crunch's recommendations on remediation.

The 42Crunch technology enables developers to

- Discover REST APIs in their GitHub repositories
- Audit each API with 200-plus security checks from 42Crunch covering industry best practices across authentication, authorization, transport, and data validation
- Analyze the discovered vulnerabilities by looking into the details provided for each vulnerability within GitHub code-scanning alerts
- Fix the vulnerabilities by going through the prioritized alert list and fixing the issues with remediation suggestions provided for each alert
- Enforce security by setting criteria for their CI/CD workflows and automated pull request checks.

Future plans

Future plans include further expansion of testing coverage, more customization for customer-specific policies, and more testing and remediation functionality becoming available as plug-ins in other tools.

Table 1: Data sheet: 42Crunch

Product/service name	42Crunch API Security Platform	**Product classification**	API security
Version number	SaaS product	**Release date**	Latest update: October 2020
Industries covered	All: this is a horizontal technology platform	**Geographies covered**	Global, with most customers in the Americas and Europe, Middle East, and Africa (EMEA)

Relevant company sizes	Historically enterprise, now expanding into freemium and SMB	**Licensing options**	Mostly subscription, though enterprises can buy a perpetual license
URL	42crunch.com	**Routes to market**	Direct, resellers, and systems integrators (SIs); adding online marketplaces
Company headquarters	London, UK	**Number of employees**	40+

Source: Omdia

Appendix

Authors

Rik Turner, Principal Analyst, Cybersecurity

Rob Bamforth, Associate Analyst

askananalyst@omdia.com

Omdia Consulting

We hope that this analysis will help you make informed and imaginative business decisions. If you have further requirements, Omdia's consulting team may be able to help you. For more information about Omdia's consulting capabilities, please contact us directly at consulting@omdia.com.

Copyright notice and disclaimer

The Omdia research, data and information referenced herein (the "Omdia Materials") are the copyrighted property of Informa Tech and its subsidiaries or affiliates (together "Informa Tech") and represent data, research, opinions or viewpoints published by Informa Tech, and are not representations of fact.

The Omdia Materials reflect information and opinions from the original publication date and not from the date of this document. The information and opinions expressed in the Omdia Materials are subject to change without notice and Informa Tech does not have any duty or responsibility to update the Omdia Materials or this publication as a result.

Omdia Materials are delivered on an "as-is" and "as-available" basis. No representation or warranty, express or implied, is made as to the fairness, accuracy, completeness or correctness of the information, opinions and conclusions contained in Omdia Materials.

To the maximum extent permitted by law, Informa Tech and its affiliates, officers, directors, employees and agents, disclaim any liability (including, without limitation, any liability arising from fault or negligence) as to the accuracy or completeness or use of the Omdia Materials. Informa Tech will not, under any circumstance whatsoever, be liable for any trading, investment, commercial or other decisions based on or made in reliance of the Omdia Materials.

Citation policy

Request external citation and usage of Omdia research and data via citations@omdia.com.

Omdia consulting

We hope that this analysis will help you make informed and imaginative business decisions. If you have further requirements, Omdia's consulting team may be able to help you. For more information about Omdia's consulting capabilities, please contact us directly at consulting@omdia.com.

Copyright notice and disclaimer

CONTACT US

omdia.com

askananalyst@omdia.com