OX DOC

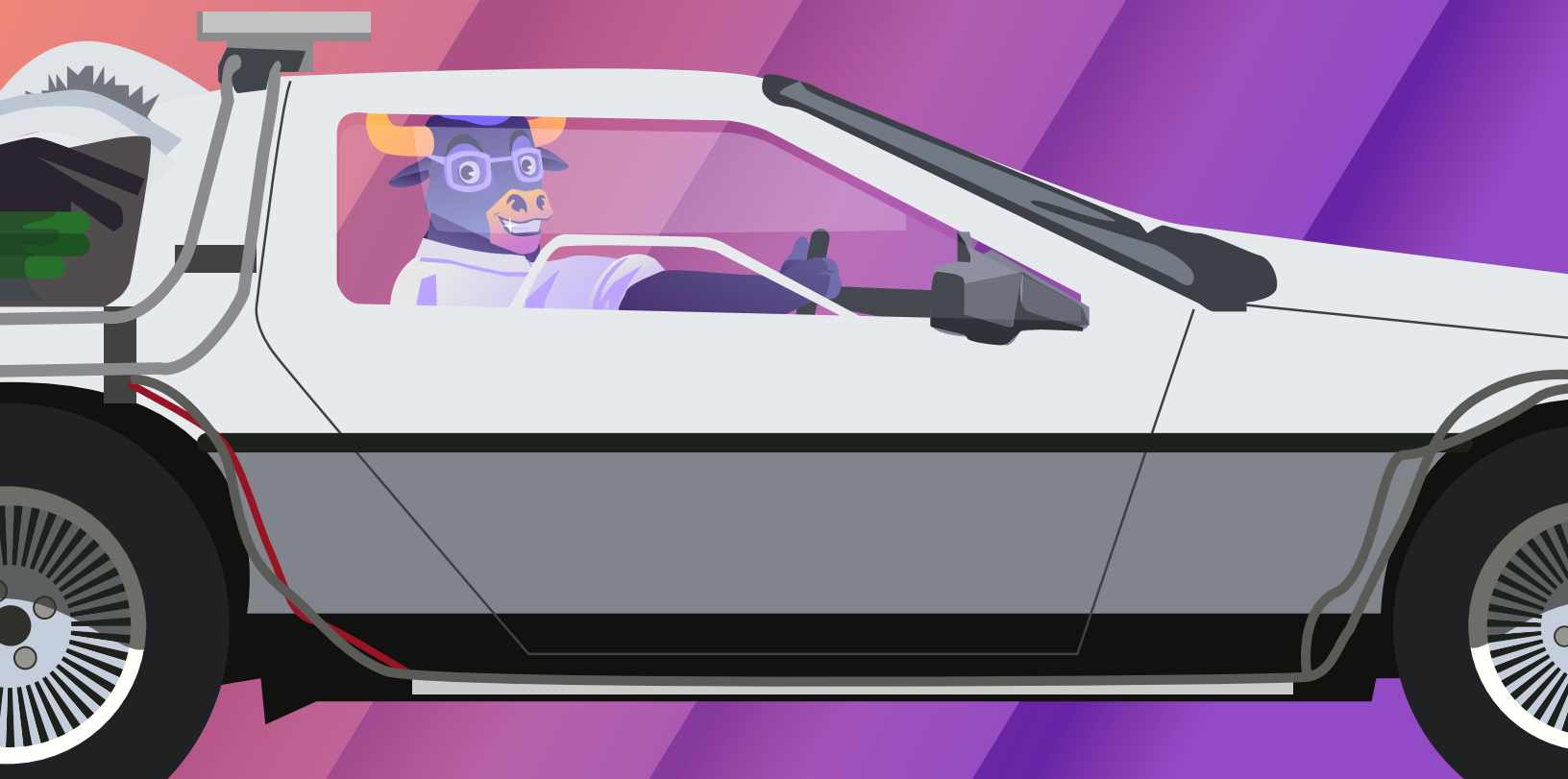
ox security

BACK TO THE FUTURE:

What AppSec can learn from 30 years of cybersecurity

Fragmented security solutions. Security gaps. Alert fatigue. Scalability issues... AppSec 2024, meet cybersecurity 2010.

And learn.



History doesn't repeat itself, but it often rhymes

Just over 20 years ago, Watts Humphrey declared that every business was a software business.

Not everyone agreed.

Today, sports shoe manufacturers, automakers and even barbecue brands are building, developing and shipping software at a pace that, a decade ago, would have made engineers' eyes bleed.

It's the kind of transformation that makes life difficult for AppSec teams: What do you do when the entire approach to software development has changed so radically? When developers must now be security-minded? Where release cycles have gone from a year to a day — and the output is going to the cloud? In a world where 91% of organizations experienced **at least one** software supply chain security incident in 2023, how do you maintain visibility and identify critical flaws and weaknesses in code when there's so much going on?

Fifty-four percent of software engineering leaders are now directly responsible for ensuring the security of applications. Like their IT security colleagues, they're finding that traditional security approaches can't keep pace with new realities. Balancing agile software development with proactive security — and extending that approach to cloud-native application architectures that already reduce the effectiveness of existing controls — has shifted toward a new playbook that includes automation, integration, risk management and new frameworks.

It's time for a new AppSec playbook, but where do you start?

You could start by asking your friends in cybersecurity. Because they've been here before, and the parallels are remarkable. In this eBook, we take a look at some of the approaches and frameworks that grew out of times of technology disruption and transformation for cybersecurity teams. We'll explore how they evolved from a reactive approach to a more proactive, flexible and intelligent process — and how that resonates with AppSec today. **So let's start at the beginning…**

How we got here...

Hello computer

The late '80s and early '90s saw a dramatic shift in network security. We moved from a world of centralized, perimeter-bound networks built by specialized companies for specialized companies into one where everyone had a computer and access to connected systems

Four new realities emerged:

1. **Networking became distributed**
2. **Many technology users weren't experts**
3. **Network assets became increasingly interconnected**
4. **Security tools and approaches for these new environments were inadequate**

The transformation was rapid: viruses and malware morphed from hypothetical and CompSci lab projects into the mainstream. Network attacks became an actual thing. Situations that were barely relevant before had evolved into massive challenges. And in response to this transformation, a completely new segment of IT security tools emerged.

Sound familiar?

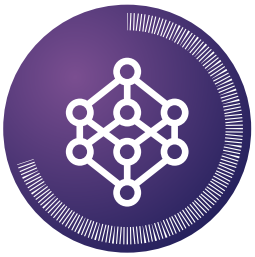
Needle meets haystack

The reality that first-gen tools came with limitations and surfaced new problems will sound familiar to AppSec practitioners. Back then, security tools like AV, firewalls and network protection couldn't handle this new wave of threats effectively. Signature-based and resource-intensive, they produced too many false positives, and (possibly most importantly) could only detect known attack types. As soon as the attack changed, even a little, detection failed.

False positives were a significant issue; anything falling within the scope of the signature was flagged, and inaccuracy bred inaccuracy, generating yet more false positives. While all of this was going on, the software, tools and threats themselves were evolving and changing. So a second generation of tools was invented...

By now, security teams were dealing with more technology, most of it working in silos, with little integration, normalization or correlation. Next-gen tools added more data —within each silo (sounds familiar?). Practitioners needed to move beyond signatures to behavioral-based approaches that would allow them to respond in a more intelligent, flexible way.

The big shift has happened: Instead of trying to address each individual risk after the fact, we see a switch in emphasis to prevention and integration of security earlier in the development cycle.



FAST FORWARD TO APPSEC 2024

Gartner predicts that 70% of platform teams will integrate application security tools as part of internal developer platforms to scale DevSecOps by 2026 — up from 20% in 2023.

Like IT Security in the past, AppSec is responding to the challenges of complexity by trying to get in front of risks before they become an issue.

10,000 into one must go

For security teams, the second generation of tools improved detection rates, but a new challenge emerged: managing everything. If you have ten deployed tools (silos), each detecting 10,000 threats…You now have 100,000 problems to manage. Context, accuracy and an aggregated data set is the next logical step. There was a huge need to consolidate the output of ten tools into a single console, without losing sight of everything…

Enter the security operations center (SOC) and next-gen tooling, along with the cybersecurity kill chain and MITRE frameworks —revolutions in their own right, driven by the same need for contextualization, prioritization and defense strategies that eliminate gaps, reduce noisy output and maintain workflows.

Haven't we been here before?

Success breeds... confusion?

AppSec teams face an average of 118,000 vulnerability alerts across their software supply chain. If even 1% of those are being exploited in the wild, finding — and triaging — then in a sea of noise is difficult at best.

Throw in multiple tools — on average, security teams need to monitor 129 applications and manage 100,000+ alerts — and a development environment where third-party code makes up a significant portion of the codebase, and it starts to look like a tsunami.

Has our capacity to detect threats improved? Yes, it's been massive — but it's a double-edged sword: 85% of CISOs believe that vulnerability noise and alert fatigue are significant challenges to finding, responding to and remediating vulnerabilities. Again, the parallels with cybersecurity's past are notable:

- **Fragmented security solutions**
- **Alert fatigue**
- **Security gaps**
- **Manual security processes**
- **False positive rates**
- **Lack of real-time intelligence**

These are all issues for AppSec teams in 2024, but they were also massive problems for cybersecurity a decade ago. Back then, a second generation of tools had made security more proactive and flexible. Detection rates were through the roof. The problem was, there seemed to be a siloed tool for every scenario. Defenders were working across ten or more consoles, each detecting thousands of threats. The next challenge was how to combine the power of ten or more consoles into a single one — without losing sight of everything.

I got tools, they're multiplying

AppSec today is in a similar place. As we saw earlier, teams are monitoring over a hundred (sometimes static) Application Security Testing (AST) tools, including:

Software Composition Analysis (SCA): 40-80% of lines of code in new software projects comes from third parties, much of it from open source projects. SCA tools enhance AppSec by auditing third-party components and recommending patches, flagging out-of-compliance code and facilitating software fixes while developers are coding.

Dynamic Application Security Testing (DAST): Also known as "outside in", DAST takes an attacker's approach, simulating attacks on applications to expose vulnerabilities. It helps to find security gaps in code or identify unforeseen outcomes that could have a knock-on effect on security. DAST can be automated or performed manually.

Interactive Application Security Testing (IAST): Essentially a combination of SAST and DAST, IAST tests applications for vulnerability while they're in use. Sensor modules track application behavior while the tests are running, and will send alerts when/if a vulnerability is detected.

While these tools enable a more holistic approach that integrates security into DevOps processes, and help prevent siloes, they don't often "talk" to each other — literally the case with IAST, because it's programming-language dependent, meaning that some tools require a code change in order to work (or won't work with your stack at all).

Many current tools lack sufficient context to manage growing software supply chain risk, meaning that defenders are staring down the barrel of a coverage and visibility gap compounded by alert fatigue and inadequate remediation capacity. AppSec teams are also at risk of becoming bottlenecks in today's accelerated software development lifecycle (SDLC).

Again, these are waters that cybersecurity teams have already charted, so there is scope for AppSec to apply the wisdom...

Context is king

The revolution will not be siloed

For cybersecurity teams, the development of SIEM started the trend toward consolidation and orchestration. It brought the logs of multiple systems into one, giving defenders the context, accuracy and single management console they needed to analyze security data from multiple sources. SIEM also provided the data teams needed to identify threats, along with historical analysis and compliance reporting capabilities.

So far so good. Until, yet again, the data volumes increased, integration difficulties emerged and alert fatigue threatened to overwhelm security teams. Regular fine-tuning of logs for good hygiene, to ensure quality data and prevent "garbage in, garbage out"added a new administrative burden.

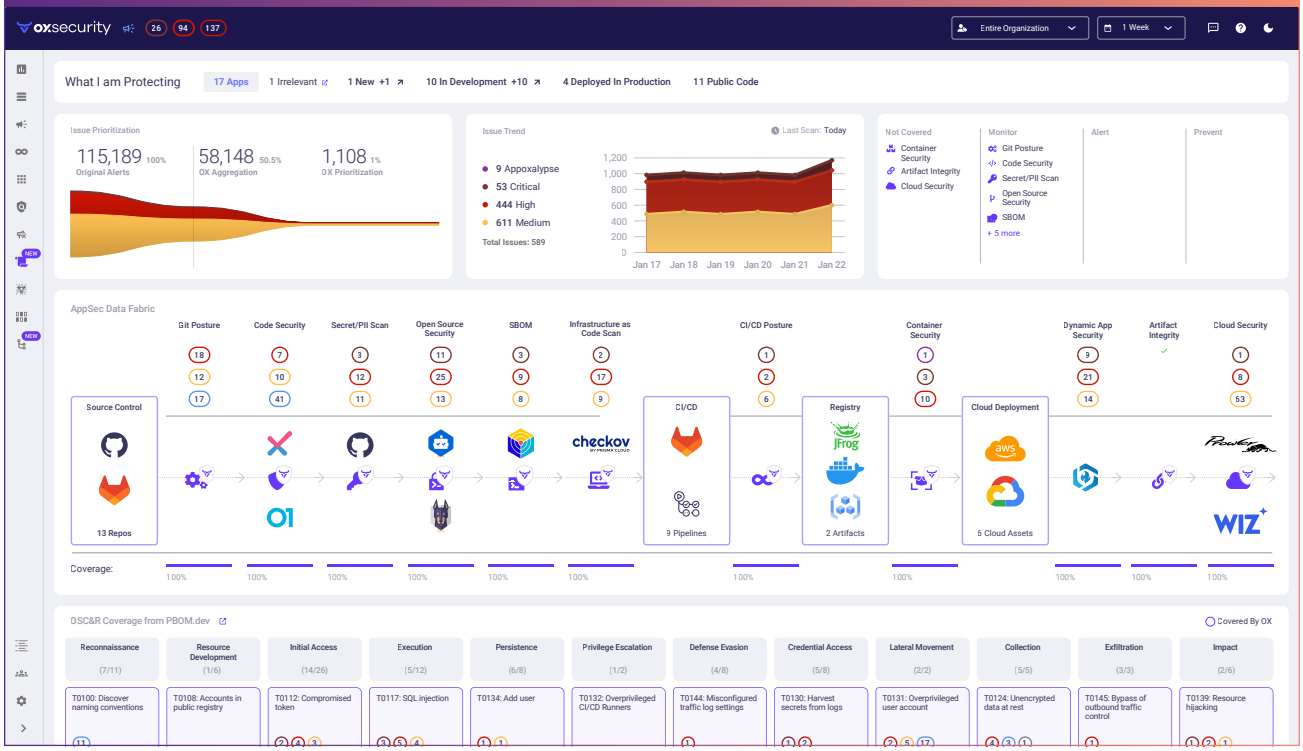
By 2020, Gartner was reporting that **57% of security leaders** weren't getting sufficient value from their SIEM investments. A more integrated, efficient approach to threat detection and response was needed and extended detection and response (XDR) emerged as both the answer and the next revolutionary step. Integrating data from multiple layers, from the endpoint to the network and everything in between, XDR brings advanced analytics and automated response to the table. No one is likely to complain about its claimed **50% reduction** in mean-time-to-detect compared to SIEM either.

ASPM: AppSec's SIEM moment

From an AppSec perspective, we're currently sitting somewhere in the SIEM-meets-XDR zone. A newer category — Application Security Posture Management (ASPM) — takes multiple siloes like static application security testing (SAST), software composition analysis (SCA), secrets detection, and infrastructure as code (IAC), and brings them into a single management plane. The next jump? Closer to XDR. ASPM provides the aggregation and correlation between products; additional data sources can bring deeper insights and context.

For AppSec defenders, ASPM is a new approach. It involves removing historical siloes between application and vulnerability scanning tools, providing more context and giving AppSec practitioners the ability to prioritize, fix and track issues throughout the SDLC. For example, AppSec teams can now not only identify software libraries used in their code, but also see code flaws, dependencies, and vulnerabilities. The ability to trace source code to all its sources — along with accompanying vulnerabilities — allows teams to move beyond simple identification and into holistic risk management. With ASPM, AppSec and developer teams can evolve from simply finding CVEs to understanding and flagging libraries that are badly maintained/ have poor hygiene/are out of date. This adds the contextual component missing from siloed, traditional AppSec and DevOps processes.

Turning fragmented AppSec siloes into unified action



The emergence of context is transformational. For instance, organizations can move from seeing an issue to seeing the specific container it was generated from, and then analyzing it to learn that the issue isn't within the container, so you can discount it. Or it is inside the container and that makes it very important. Or the issue can be mapped to another location where it can be fixed.

Once again, cybersecurity teams have a playbook that AppSec can adapt to meet its own unique needs…

Wouldn't it be great if we had a common language to describe all of this, so everyone can read from the same playbook?

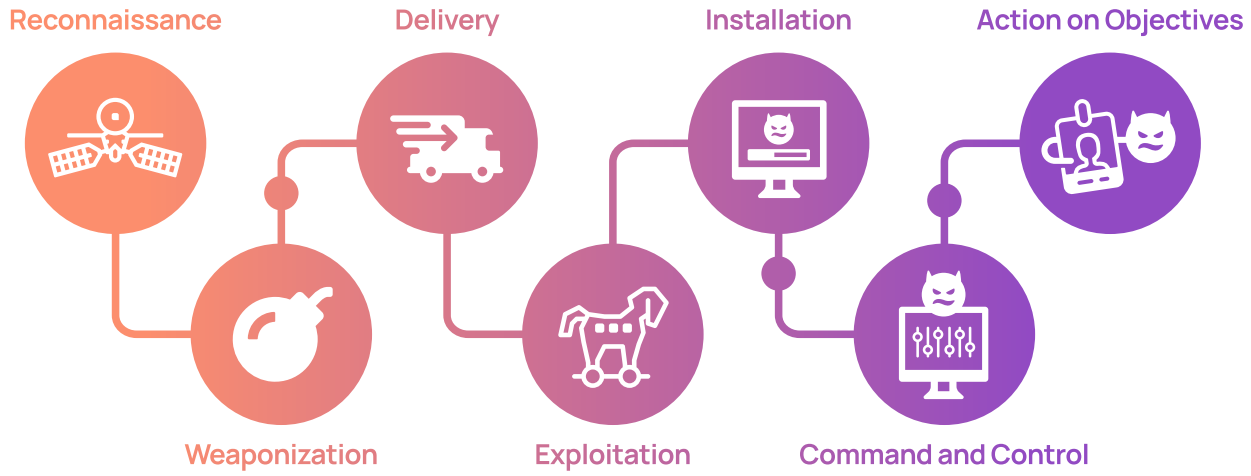
From kill chain to deep dive

Thinking like an attacker

Whether you're looking at SIEM logs or pulling insights from XDR or ASPM, the value of seeing vulnerabilities from the point of view of the attacker becomes crucial. And for that to really work, everyone — security, AppSec and development teams — needs to be on the same page, using the same frameworks and language.

Lockheed Martin's Cyber Kill Chain provided an early —and influential — framework for breaking down and understanding the stages of an attack. Adopting the perspective of the attacker, it broke cyber attacks into seven stages, each illustrating how an attacker moves through a network in search of vulnerabilities to exploit.

The Cyber Kill Chain



The thinking was that by understanding each stage of an attack, defenders could develop methods to prevent both the attack and any further progress up the kill chain. The challenge with this approach was its limited scope, lack of detail, and a linear model that didn't capture the reality of multi-faceted attacks. In addition, the process for describing an attack was complicated, a weighting system had to be incorporated, and it had vague descriptions; there wasn't a common vocabulary everyone could work from.

The MITRE ATT&CK framework evolved out of the need to address these shortcomings…

High and MITRE

The MITRE ATT&CK framework began life in 2013 as a wiki derived from the results of a Red Team (attacker) — Blue Team (defender) experiment in which researchers sought to better understand cyber threats.

"ATT&CK" stands for "Adversarial Tactics, Techniques and Common Knowledge" – MITRE essentially built a catalog of every type of attack, applied unified naming and vocabulary, and put a structure around it that enabled defenders to describe attacks in a way that security practitioners understood. This same language enabled comparisons and a unified way to talk about coverage and tools. Security professionals finally knew where they stood — and could explain it, simply, to executive teams and boards of directors. It was transformational.

Should we have a caption?

Tactic

A Tactic represents the goal that the attacker is trying to achieve

Technique

Techniques represent the different ways that cyber attackers can achieve the goals and objectives of the tactic

Procedure

Procedures represent the steps that an attacker takes to execute a technique or sequence of techniques

Over the years, the ATT&CK framework has evolved and expanded into a community-driven, global catalog of adversary tactics and new attacker techniques, from pre-attack and mobile threats to cloud security.

Maybe that kind of thinking could translate in the AppSec world. Wouldn't it be good to have a solid, unified framework for describing and understanding attacks on the software supply chain?

That was **then**, this is now

You are the weakest link. Hello.

Ninety-one percent of organizations experienced **at least one** software supply chain security incident in 2023. Chances are the other 9% are riding their luck: The average organization has nine high, critical or apocalyptic risks within their supply chain.

At the heart of the problem: Companies that aren't rooted in software development are building, developing and shipping software, often with little concept of how to secure it. Bake in accelerated release cycles and an output that's going to the cloud and it's easy to understand why a key finding of this year's **Data Breach Investigation Report** was a 180% increase in the exploitation of vulnerabilities as the critical path action to initiate a breach.

The call is coming from inside the house.

Move fast, break things

AppSec teams are in the eye of a software-defined storm. Code as everything – infrastructure, compliance, security, AI – is the new normal. The lines between developer and security pro are blurring and converging. At the same time, software release cycles have accelerated past a point where traditional security tools and approaches can keep pace. Teams can release in ops and design review isn't working because some organizations are releasing multiple times a day.

As for pen testing…short release cycles and multiple, rapid iterations make it more likely that vulnerabilities will be introduced, but keeping track and keeping up are becoming a massive challenge. Add increased reliance on open-source code and cloud-native technologies and the risk surface expands even more.

Hello weakness, my old friend

Like their colleagues in cybersecurity before them, AppSec teams are finding that traditional approaches and tools can't keep pace with the new realities. Despite advances in tools and information, research by Ox security analysts into more than one hundred million supply chain security alerts from tens of thousands of repositories, applications and organizations found that all three of the most prevalent software supply chain vulnerabilities have been around for years:

15.4% OF APPLICATIONS	12.4% OF APPLICATIONS	11.4% OF APPLICATIONS
Command injection	Sensitive data in log files	Cross-site scripting

Despite widespread awareness of them, threats like XSS are being introduced during the development process all the time. This isn't due to malice or oversight, it's due to the fact that managing security in the accelerated development environment we've just described is tricky. Modern web applications are often complex, with many interconnected components and dependencies – the likelihood of vulnerabilities slipping through the cracks or being introduced through recycles or third party code is high. And if your AppSec team is wading through 100,000+ alerts, things get overwhelming pretty quickly.

Pump up the volume

The average team now monitors 129 applications and over 119,000 alerts. The sheer volume of alerts being generated, coupled with an ever-expanding catalog of vulnerabilities, is creating a level of security debt that is in danger of overwhelming AppSec teams. Meanwhile, the gap between vulnerability and exploitation continues to shrink, while time to remediate 50% of critical vulnerabilities once a patch becomes available is 55 days.

Without alignment between vulnerabilities found in the wild and the focus of AppSec teams, organizations will continue to struggle with supply chain vulnerability. Because accelerated SDLCs make timeframes so short, there is no effective way to do this manually. Automation goes a long way towards consolidation, deduplication and contextual analysis, but as vulnerabilities continue to be passed into live applications, prevention is at least as important as detection. It's time for AppSec teams to think like an attacker…

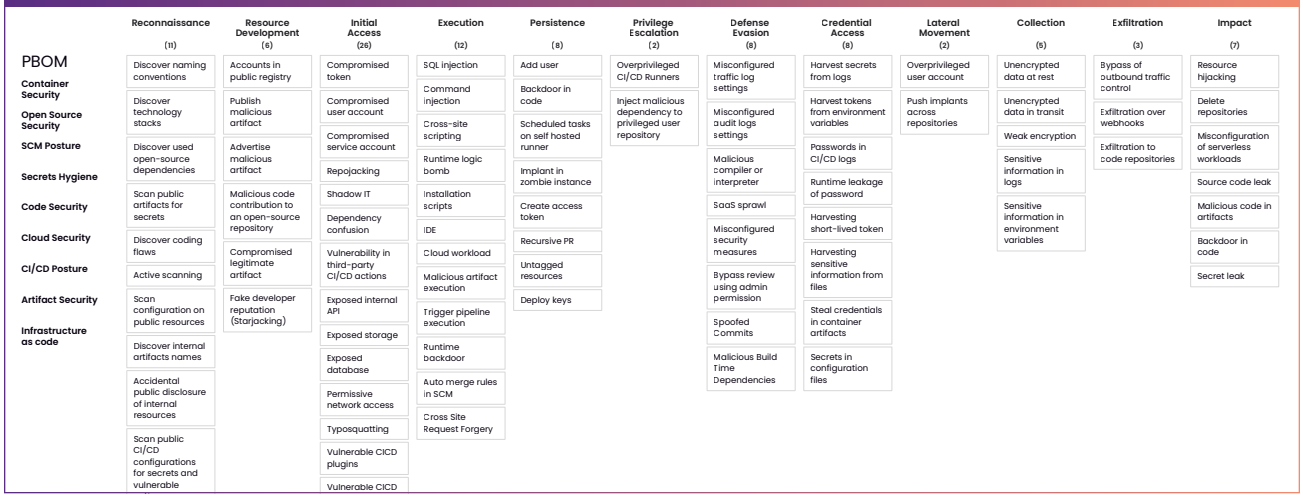
Something's gotta give

Understanding the nature of weakness and vulnerability is crucial for AppSec teams looking to develop a proactive security approach. Organizations that can think like an attacker and understand the root causes of vulnerabilities can minimize the risks and reduce the attack surface. Balancing agile software development with proactive security has shifted towards a playbook that includes automation, integration, risk management and new frameworks.

Earlier, we looked at how a new approach — Application Security Posture Management (ASPM) — is having a transformational impact, adding the contextual component that was missing from siloed, traditional AppSec and DevOps processes. The next step: a unified framework for describing and understanding attacks on the software supply chain.

Open Supply Chain Attack Reference (OSC&R)

A comprehensive, systematic and actionable way to understand attacker behaviors and techniques with respect to the software supply chain



An OSC&R-winning framework

Based on real-world, in-the-wild observations, the **MITRE att&ck framework** gave cybersecurity teams a common language and model for describing and understanding attacker tactics and techniques. Inspired by its success, OX collaborated with other experts from GitLab, Google and Microsoft to develop an ATT&CK-like open framework and model to understand the entire software supply chain. The result: **Open Software Supply Chain Attack Reference (OSC&R) framework**.

Like the MITRE approach, OSC&R creates a common language for discussing and analyzing the tactics, techniques and procedures malicious actors use to target the software supply chain. The framework takes tools to the next level, contextualizing risk and helping both AppSec and AppDev teams to keep up with the latest attack trends.

OSC&R takes an attacker-centric view, with phases and TTPs (tactics, techniques and procedures) specific to software supply chains, giving AppSec teams a new way of thinking about their environment. By understanding how attackers view and target the attack surface of the supply chain — and by using a common language to describe threats — AppSec, DevOps and security teams can align more effectively to mitigate risk at every stage of the SDLC, and avoid introducing it in the first place.

That was then, this is now

The new AppSec playbook

As we've seen over the course of this eBook, traditional approaches to AppSec no longer work. Software supply chains have become an ever-expanding attack surface. With the sheer volume of alerts and vulnerabilities, detection alone is not enough — it's time to address risk at every step of the Software Development Lifecycle (SDLC).

Like the MITRE framework before it, the OSC&R framework marks a significant transformation in how AppSec teams address these challenges. Coupled with Application Security Posture Management (ASPM), it brings a new approach, removing historical siloes and providing context to identify, prioritize, fix and track issues throughout the SDLC.

For anyone charged with understanding what the future of AppSec could look like, there's a lot to learn from our security past. The tools, frameworks and solutions that evolved to address changing cybersecurity needs provide a useful lens through which AppSec defenders can view the challenges they face today. Like our cybersecurity colleagues in the past, an ever-expanding volume of vulnerabilities and alerts has driven the evolution of new frameworks and approaches for insight and mitigation.

Start your Application Security journey with OX Security

The future of AppSec begins here

The evolution of cybersecurity has provided us with valuable lessons, but applying them to the unique challenges of modern application security requires the right tools and strategies. As we've explored throughout this eBook, the growing complexity of software supply chains and the increasing volume of vulnerabilities demand a proactive, integrated approach to security.

That's where OX Security comes in.

OX Security is at the forefront of transforming application security. By leveraging the OSC&R framework and our AppSec Data Fabric, the OX Active Application Security Posture Management (ASPM) platform helps you break down silos, prioritize risks and automate remediation throughout the Software Development Lifecycle (SDLC). Our solutions are designed to either work out of the box through our proprietary scanning and/or integrate seamlessly with your existing tools, providing a unified view that aligns your AppSec, Product Security and DevOps teams.

Why choose OX Security?

Comprehensive Coverage: Gain complete visibility across your software supply chain, from open-source components to third-party integrations.

Advanced Contextualization: Identify and prioritize vulnerabilities with the context needed to address the most critical risks first.

Proactive Risk Management: Move beyond detection to prevention, ensuring that vulnerabilities are addressed before they become threats.

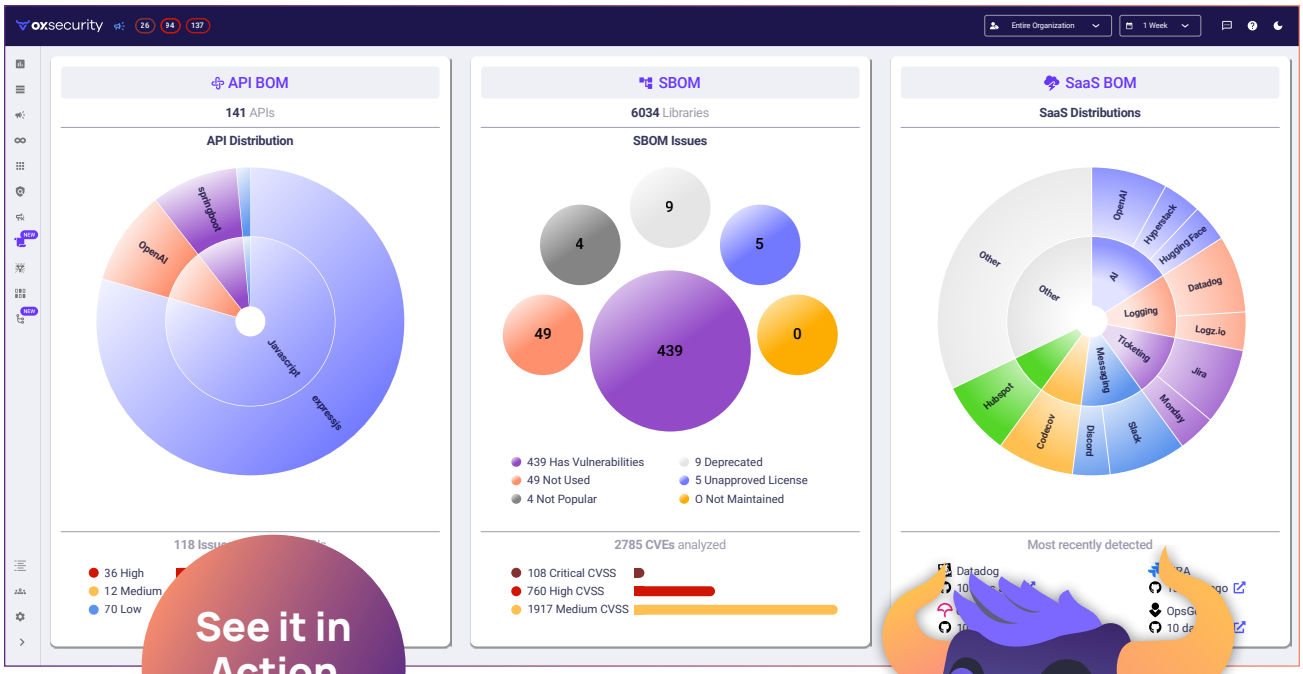
Seamless Integration: Work within your existing development and security workflows with tools that enhance, rather than disrupt, your processes.

Get started today

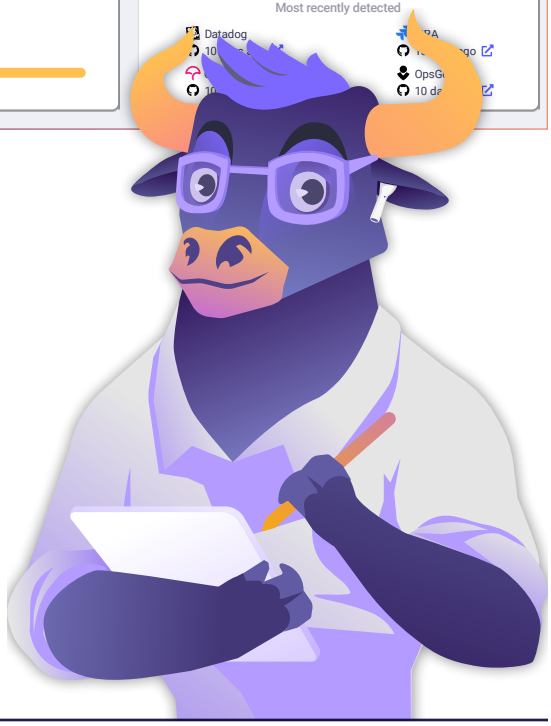
The future of AppSec is here, and it starts with understanding the unique challenges and opportunities of your environment. Let OX Security guide you on your journey to a more secure, resilient software supply chain. Visit our website to learn more about how we can help you cut out manual AppSec processes through simplification and automation while building scalable and secure development.

Don't wait—secure your software supply chain today with OX Security.

oxsecurity



See it in Action Today!



About OX Security

At OX Security, we're simplifying application security (AppSec) with the first-ever Active ASPM platform offering seamless visibility and traceability from code to cloud and cloud to code. Leveraging our proprietary AppSec Data Fabric, OX delivers comprehensive security coverage, contextualized prioritization, and automated response and remediation throughout the software development lifecycle. Recently recognized as a Gartner Cool Vendor and a SINET 16 Innovator, OX is trusted by dozens of global enterprises and tech-forward companies. Founded by industry leaders Neatsun Ziv, former VP of CheckPoint's Cyber Security business unit, and Lior Arzi from Check Point's Security Division, OX's Active ASPM platform is more than a platform; it empowers organizations to take the first step toward eliminating manual AppSec practices while enabling scalable and secure development.

INTERESTED IN LEARNING MORE VISIT: WWW.OX.SECURITY/BOOK-A-DEMO/

oxsecurity