DZone

THE 2018 DZONE GUIDE TO

Security

DEFENDING YOUR CODE

VOLUME IV

BROUGHT TO YOU IN PARTNERSHIP WITH

bmc

CYBERARK® CONJUR

edgescan

INSPEC BY CHEF

graylog

threat stack

Dear Reader,

Welcome to *DZone's 2018 Guide to Security: Defending Your Code*, our fourth Guide on the subject. In my year and half manning the desk as the editor for our Security Zone, I witnessed firsthand developers' interest in the field of security grow. But I also saw the frustrations that topics such as GDPR brought about and the, for lack of a better word, panic surrounding major breaches like Equifax.

In this Guide, the articles you'll read meet at the crossroads of these topics. We discuss how companies get hacked, and how these organizations can take proactive steps by making security a first-class citizen. But we also delve into tutorials for the more technically inclined reader, covering topics such as virtual patching, how to sure embedded systems, and adding security protocols to mobile applications. Finally, we look to the future, and how AI and blockchain promise to revolutionize the way we conceive of and implement cybersecurity.

In years past, software security was an often-ignored process, pushed to the corner of the room and ignored. Or, at best, it had a dedicated, but small, team of security professionals who were allotted little to no time to do any meaningful security testing. Even today, release dates and application performance tend to outweigh security concerns. But the world of security is changing. The shift-left methodology borrowed from DevOps is allowing security professionals and developers to implement security protocols earlier in the development of their software; more time gets set aside to perform penetration tests; and developers are actively learning about the best ways to increase security in their applications, whether that means integrating an authentication solution or conducting security code reviews.

Thank you for downloading the *2018 DZone Guide to Security*. As security continues to be treated more and more as a first-class citizen by developers, dev leads, and organizations as a whole, we hope you'll think of DZone as the place to stay informed on trends and learn new security skills. If this Guide has piqued your interest, be sure to head over to the Security Zone and check out the hundreds of articles we house on a variety of security topics, and thumb through our security-related Refcardz, like our card on REST API Security.

By Jordan Baker
CONTENT COORDINATOR, DZONE

Table of Contents

DZone is...

BUSINESS & PRODUCT
MATT TORMOLLEN, CEO
MATT SCHMIDT, PRESIDENT
JESSE DAVIS, EVP, TECHNOLOGY
KELLET ATKINSON, MEDIA PRODUCT MANAGER

PRODUCTION
CHRIS SMITH, DIRECTOR OF PRODUCTION
ANDRE POWELL, SR. PRODUCTION COORD.
ASHLEY SLATE, DESIGN DIRECTOR
G. RYAN SPAIN, PRODUCTION COORD.
BILLY DAVIS, PRODUCTION COORD.
NAOMI KROMER, SR. CAMPAIGN SPECIALIST
JASON BUDDAY, CAMPAIGN SPECIALIST
MICHAELA LICARI, CAMPAIGN SPECIALIST

EDITORIAL
CAITLIN CANDELMO, DIR. OF CONTENT & COMMUNITY
MATT WERNER, PUBLICATIONS COORDINATOR
SARAH DAVIS, PUBLICATIONS ASSOCIATE
MICHAEL THARRINGTON, CONTENT & COMMUNITY MANAGER II
KARA PHELPS, CONTENT & COMMUNITY MANAGER
TOM SMITH, RESEARCH ANALYST
MIKE GATES, CONTENT TEAM LEAD
JORDAN BAKER, CONTENT COORDINATOR
ANNE MARIE GLEN, CONTENT COORDINATOR
ANDRE LEE-MOYE, CONTENT COORDINATOR
LAUREN FERRELL, CONTENT COORDINATOR
LINDSAY SMITH, CONTENT COORDINATOR

SALES
CHRIS BRUMFIELD, SALES MANAGER
FERAS ABDEL, SALES MANAGER
JIM DYER, SR. ACCOUNT EXECUTIVE
ANDREW BARKER, SR. ACCOUNT EXECUTIVE
BRETT SAYRE, ACCOUNT EXECUTIVE
ALEX CRAFTS, KEY ACCOUNT MANAGER
BRIAN ANDERSON, KEY ACCOUNT MANAGER
SEAN BUSWELL, SALES DEVELOPMENT REPRESENTATIVE

MARKETING
SUSAN WALL, CMO
AARON TULL, DIRECTOR OF MARKETING
SARAH HUNTINGTON, DIRECTOR OF MARKETING
WAYNETTE TUBBS, DIRECTOR OF MARKETING COMM.
KRISTEN PAGÀN, MARKETING SPECIALIST
COLIN BISH, MARKETING SPECIALIST

Executive Summary

BY MATT WERNER - PUBLICATIONS COORDINATOR, DZONE

This is DZone's fourth research guide focused on application security, and as expected, it is still one of the biggest issues facing the entire software industry. In 2018 alone, we've seen major breaches at Exactis (a marketing firm), Ticketfly, and, perhaps most worryingly, Facebook — in which a political consultancy scraped details on over 87 million users of the social network (and has yet to be officially punished by any authority). In the wake of continued major breaches, DZone asked 426 readers to share with us their interest in application security, what challenges they faced, and how they adapt to new threats and breaches emerging every day.

FEWER DEVELOPERS INTERESTED IN BEING RESPONSIBLE FOR SECURITY

Data: In 2017, 53% of survey participants believed developers should be primarily responsible for security, compared to 42% in 2018. 31% of respondents in 2018 felt that security teams should be more responsible, compared to 28% in 2017. Respondents are less confident about putting responsibility on security frameworks, with 15% believing they should be relied on in 2018, a 5% drop from 2017. In 2018, 6% of respondents each believed that managers and senior leadership (i.e. C-suite) should be responsible for security.

Implications: The large drop in those who believe developers should be primarily responsible for security can be attributed to a few factors. The first is the addition of management and senior leadership options in our survey, which did not exist in 2017. Another is perhaps an issue of mixed messaging to developers. Application performance is seen as a higher priority than security (36% vs. 31%), and developers may feel like their priorities lie elsewhere.

Recommendations: There is still a majority of respondents who feel developers should be primarily responsible for security, and while organizations figure out a proper separation of work and duties between development and security teams, developers should learn all they can. Helpful resources such as OWASP and DZone's Security Zone can help keep developers up-to-date with what threats they need to defend against. It is also good practice to include automated security testing into the SDLC, and to keep documentation up-to-date with how code is secured.

DEVELOPERS RECEIVING MORE TRAINING THAN BEFORE

Data: In 2018, 4% fewer survey respondents reported that they were being trained on an ad-hoc basis (33% from 37%), and 2% fewer respondents reported that they were never getting training on security (25% from 27% in 2017). Quarterly trainings increased 6% from 9% in 2017, while there was no statistically significant different in yearly (15% in 2018 vs. 16% in 2017) or semi-annual (13% in 2018 vs. 12% in 2017) trainings.

Implications: There was a huge jump in quarterly security training for developers across all organizations, signifying that security is being seen more and more as a necessity for organized training — very different than learning-on-the-go when there's a security breach or if there is a nonspecific directive from managers.

Recommendations: The jump in quarterly security trainings is a huge step in the right direction for organizations to educate developers and tech professionals about proper security best practices. The goal is to see ad-hoc training, which can range from impromptu trainings to Googling a solution to a problem. Press managers and senior leadership can organize regular trainings catered to what developers are working on at your organization.

THE MOST COMMON VULNERABILITIES

Data: The five most common types of vulnerabilities for web applications encountered on a regular basis are weak authentication and session management (43%), cross-site scripting (40%), sensitive data exposure (39%), injection attacks (39%), and security misconfiguration (38%). Survey takers selected these vulnerabilities from the OWASP Top Ten.

Implications: Cross-site scripting and injection attacks are both very common, indicating that developers are either not properly sanitizing their inputs or that there are other issues present. The prevalence of weak authentication and session management suggest that users may not be assigned proper roles that only grant them access to what they need.

Recommendations: Developers should first become familiar with at least the OWASP Top Ten, as it is all but a standard resource for developers with an eye on security. Security testing will also help reveal several, if not all, of these vulnerabilities, such as cross-site scripting. Think about ways in which attackers could breach your application and defend against those avenues.

Key Research Findings

BY JORDAN BAKER - CONTENT COORDINATOR, DZONE

DEMOGRAPHICS

For this year's DZone Security Survey, we received 683 responses with a 62% completion percentage. Here's how some of the demographics for our respondents break down:

- 42% work for companies that are headquartered in the USA, 18% work for companies in South Central Asia, and 13% work for companies in Europe.
- 21% work for software vendors, 18% in finance/banking, and 10% in e-commerce.
- 24% work for organizations sized 10,000+ employees, 23% for organizations sized 100-999 employees, and 20% for organizations with 1,000-9,999 employees.
- 35% work as developers/engineers, 25% as developer team leads, and 17% as architects.
- 84% develop web applications/services (SaaS), 48% develop enterprise business applications, and 28% develop native mobile applications.
- 84% work for companies that use the Java ecosystem, 67% use the JavaScript (client-side) ecosystem, and 37% use the Node.js (server-side) ecosystem.

DEVELOPERS AS SECURITY PROFESSIONALS

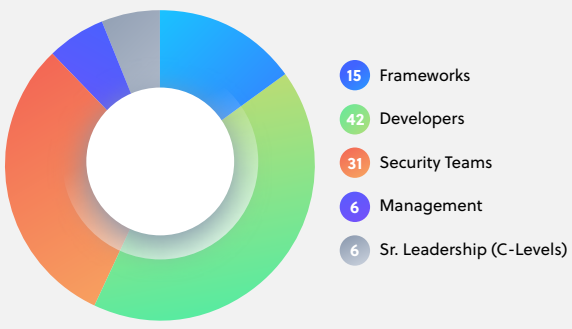
Over the last several years, we've witnessed a large push among the developer community for security to shift left in the SDLC. The statistics from this year's DZone Security Survey show the effectiveness of this trend. When asked who should take primary responsibility for security, 42% of respondents said developers, 31% said security teams, and 15% said the frameworks themselves. Of the respondents who answered this question, 35% currently work as developers/engineers. Of those 35% currently employed in developer roles, almost half (41%) told us they believe developers should be primarily responsible for security. Additionally, 53% of developer team leads reported that developers should be primarily responsible for security. These are both promising trends in the field of application security.

A core part of the shift left movement in application security is not only increasing concern for security among developers, but also providing developers with the necessary training and resources to learn secure coding practices. Taking a historical look at security training data, we can see some positive signs. In the 2017 and 2018 DZone Security Surveys, we asked how frequently developers at our respondents' organizations received security training. Here's how their answers broke down:

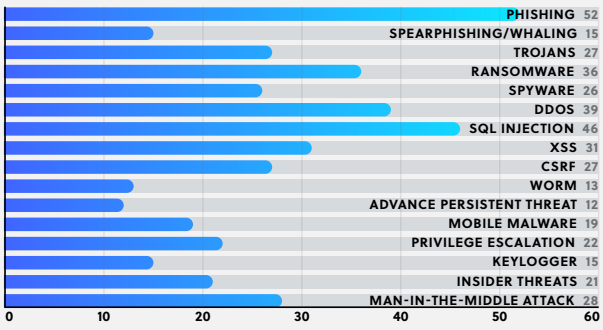
- Ad-hoc:
  - 2017: 37%
  - 2018: 33%
- Never:
  - 2017: 27%
  - 2018: 25%
- Yearly:
  - 2017: 16%
  - 2018: 15%
- Semi-annually:
  - 2017: 12%
  - 2018: 13%
- Quarterly:
  - 2017: 9%
  - 2018: 15%

The increase in quarterly security training proved quite substantial and is

WHO SHOULD BE PRIMARILY RESPONSIBLE FOR SECURITY?



WHICH OF THESE TRAITS HAVE YOU BEEN MOST CONCERNED ABOUT IN YOUR ORGANIZATION IN THE PAST YEAR?



inversely proportional to the percentage of developers reporting that they receive security training on an ad-hoc basis or no training at all. While ad-hoc or no security training remain the two largest categories reported by our respondents, the decrease in their instances over a year, and the marked jump in quarterly trainings, is a positive sign for the industry.

While security has certainly become a greater concern for developers in recent years, it continues to be outweighed by performance concerns. 37% of respondents said that their organization views performance as the largest priority, while 31% reported that security is their organization's most important concern. In addition to performance, releasing software on schedule often overrides security issues. Approximately half of this year's respondents (51%) reported allowing release schedules to interfere with security concerns on an at least semi-regular basis. To take a more granular look, 34% said release schedules "sometimes" interfere with security, 11% reported "often," and 6% reported it happens "all the time." In fact, only 10% of this year's survey takers told us that releasing on schedule never overrides security concerns; and an additional 25% said it rarely happens.

So, what happens when a vulnerability slips through the cracks? 83% of respondents told us they inform customers of potential known vulnerabilities that got shipped in their software. While we'd all like to see this number at 100%, it's still a marked increase from 2017, when only 67% of respondents reported informing customers of potential vulnerabilities in their solutions.

APPSEC

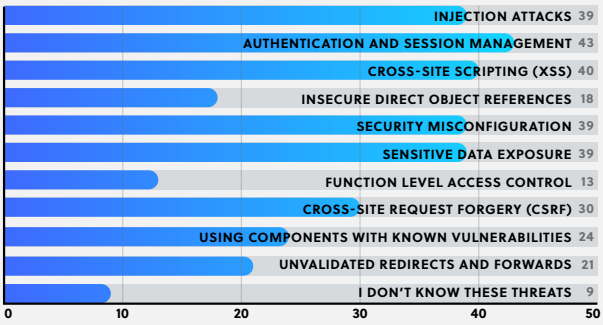
Now that we've established that security is becoming a greater concern among developers, let's explore how developers are approaching application security. When we asked at what stage of the SDLC respondents spend the most time implementing security measures, 33% said design, 28% said implementation, 14% said testing, and 13% said requirements gathering and analysis. The fact that a third of respondents implement security in their code in the design phase provides another indication that security is shifting left in the SDLC. This also means, however, that more effort is being put into securing new code than legacy code. In fact, 31% of respondents told us they spend more time per week securing new code, whereas 20% told us they spend more time securing legacy code. Another 31% reported spending equal amounts of time securing legacy and new code, but even with that demographic included, the security scales still tip toward new code.

When it comes to secure coding techniques, validating inputs proved the most popular option among respondents, with 74% of survey takers reporting to use this tactic. In 2017, however, 83% of our Security Survey respondents told us they validated inputs. Architecting and designing for security policies is a technique that seems to be on the rise, however. In 2017, 59% of respondents reported using this approach; this year, that number rose to 64%. Given the increased popularity of architecting and designing for security policies, let's quickly examine the architectural patterns developers use to secure their code. The two most popular responses were roles and sessions, though these too saw a slight year-over-year decrease. 70% of respondents reported using roles as their architectural pattern, falling from 75% in 2017, and 62% reported using sessions as an architectural pattern, whereas 67% had told us they used sessions in 2017. Secure access layers came in a close third, with 61% of respondents telling us they use this pattern.

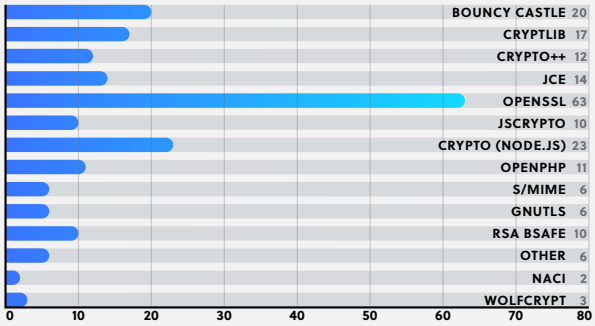
Within these larger architectural patterns, two means of securing code dominate. When we asked what APIs and implementations respondents use for encryption, 63% reported using OpenSSL. Interestingly, OpenSSL proved exceedingly more popular among respondents working as web application developers than those working as enterprise business application developers. Among web application developers, 87% reported using OpenSSL for their encryption purposes, whereas 52% of enterprise business application developers told us they use OpenSSL. Though OpenSSL dominated the responses, the Crypto package for Node.js saw an impressive year-over-year increase. 23% of this year's respondents reported using Crypto, up from 16% in 2017.

The second means of securing code, which dominated responses, was the use of authentication tokens (including digital signatures) for verifying message integrity. 72% of survey-takers said they use authentication tokens in their applications, while the second most popular option among respondents, the use of check sums, received only 34% of the responses. Unlike OpenSSL, however, the use of authentication tokens for securing application code proved relatively equal between web

WHICH OF THE FOLLOWING TYPES OF VULNERABILITIES IN YOUR WEB APPLICATIONS DO YOU ENCOUNTER MOST FREQUENTLY (FROM THE OWASP TOP 10)?



WHAT APIS AND IMPLEMENTATIONS DO YOU USE FOR ENCRYPTION?



application developers and enterprise business application developers. 78% of respondents working as web app developers and 74% of enterprise business app developers use authentication tokens to secure their code.

Unlike the means by which developers secure their code, there's no dominant form of testing. Among our respondents, 24% use penetration testing, 17% use security code review, 14% perform source code analysis, 13% do vulnerability assessments, and 12% have no formal security testing. Given that the difference between the most and least popular form of testing is only 12%, let's again use web app developers and enterprise business app developers as a point of comparison to get a more granular look at how respondents are using security testing. Penetration testing proved the most popular option among both web app and enterprise business app developers, receiving 27% of web app dev's responses and 28% of business app dev's responses. Interestingly, the second most popular option among web app developers was performing security code reviews (18%), whereas for enterprise business app developers it was source code analysis (17%).

Despite the use of these testing techniques, 48% of respondents reported that among the applications they test for security, sufficient test coverage is never achieved. While that number is a bit troubling, it dropped from last year, when 54% of survey respondents reported inadequate test coverage. Additionally, 33% reported that the threat model is handled in an acceptable way, up from 27% in last year's survey.

VULNERABILITIES/ATTACKS

Over the past 12 months, the IT industry has witnessed several large-scale attacks, such as the hacking of Equifax and instances of ransomware like NotPetya, and vulnerabilities exploited, like the infamous struts vulnerability that eventually led to the Equifax hack. But do the realities on the ground match the sensational headlines?

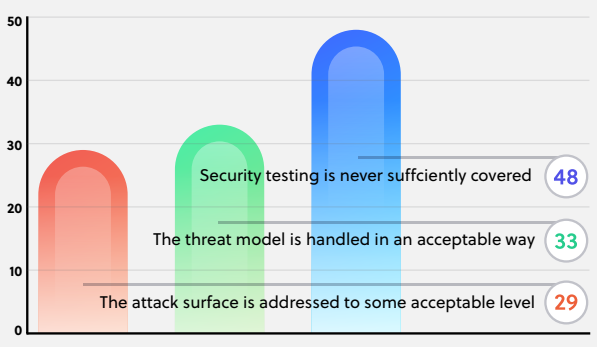
We asked our survey-takers what threats have most concerned their organization over the past year. Despite the flabergastingly bad past year for cybersecurity, our respondents' answers remained virtually identical to those reported in the 2017 DZone Security Survey. 52% reported phishing as their organization's biggest concern, 46% said SQL injection (SQLi), 39% said DDoS, 36% reported ransomware, and 31% said cross-site scripting (XSS) attacks. Even when we compare this data to to our two main developer verticals (web app and enterprise business app developers), the numbers regarding threats that concern their organizations don't undergo any statistically significant changes.

Something interesting does pop out, however, when we compare the threats that most concern organizations and the types of vulnerabilities developers encounter most often. While most vulnerabilites our respondents reported encountering were not that surprising, such as authentication + session mangagment (43%) and cross-site scripting (40%), unvalidated redirects + forwards were selected by a rather small number of respondents. Unvalidated redirects + forwards was the eighth most common vulnerability from the OWASP Top 10 faced by respodents, with 23% of survey-takers reporting to have had issues with this vulnerability. The low position of unvalidated redirects + forwards is surprising given the role this vulnerability plays in the spread of phishing attacks, which was the most prominent organizational security concern among our respondents. Unvalidated redirects and forwards are, in fact, the programmatic mechanism for driving users to a seemingly innocuous, but malicious site (Paul Ionescu, "The 10 Most Common Application Attacks in Action," SecurityIntelligence by IBM). Thus, despite a low instance of phishing attacks over the past year, it seems organizations are bracing for this type of cyberattack to increase in frequency.

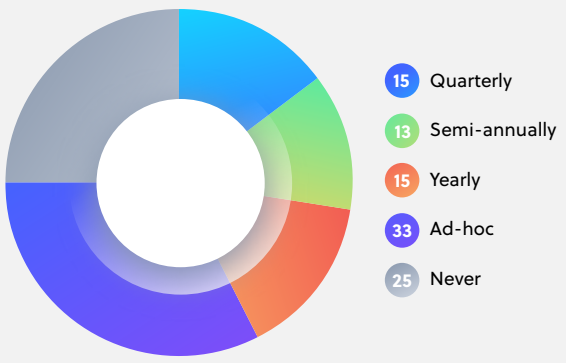
Given that 39% of respondents reported having faced issues with denial-of-service attacks, let's quickly go over the data regarding this common type of attack. Having to deal with many high-resource connections proved by far the most common instance of DDoS attacks faced by survey-takers, with 54% of respondents having faced this issue. The second most common DDoS faced was requests for large files (30%). No other form of DDoS attack registered more than 18% of respondents' votes.

So, how do these attacks and vulnerabilities affect respondents' ability to deploy their software? 43% reported that security analysis and vulnerability-fixing had a medium impact, 36% reported a low impact, and 13% reported a high impact. These numbers are, again, nearly identical to last year's DZone Security Survey results.

OF THE APPLICATIONS YOU TEST FOR SECURITY, HOW IS SUFFICIENT TESTING DETERMINED?



HOW FREQUENTLY ARE DEVELOPERS AT YOUR ORGANIZATION TRAINED IN SECURITY?



Security Through Blockchain

QUICK VIEW

01. Blockchain technology provides openness, immutability, and distributed topology that changes the way applications are secured.

02. Blockchains are becoming an essential facet of many sectors, including monetary, legal, and commerce.

03. The largest financial institutions and companies, like ICE, IBM, Microsoft, and Starbucks, are putting their trust in blockchain technology.

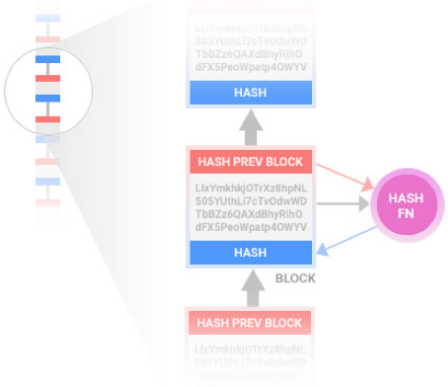
BY JUSTIN ALBANO
SOFTWARE ENGINEER, **CATALOGIC SOFTWARE**

Since its inception in 2008 by Satoshi Nakamoto, blockchain technology has spread across the globe at an astonishing pace and is positioned to be one of the greatest technological advancements in decades. From cashless digital currencies to food supply chains, blockchains are helping to radically change the way the world thinks and operates. Foremost among its benefits is the intrinsic security forged in its openness, immutability, and distributed topology, which places trust in a network of participants rather than a central authority.

Amid the whirlwind of excitement and flourishing of blockchain-based cryptocurrencies, such as Bitcoin, a great deal of conflation and misconception has arisen about what blockchain technology is and how it can be used to solve practical problems. In order to understand how it can be useful for securing applications — and how so many people and companies have placed their trust in this emerging technology — it is essential to take a detailed look at the embedded security of blockchains and how they are used in practice, beginning with what constitutes a blockchain.

WHAT IS A BLOCKCHAIN?

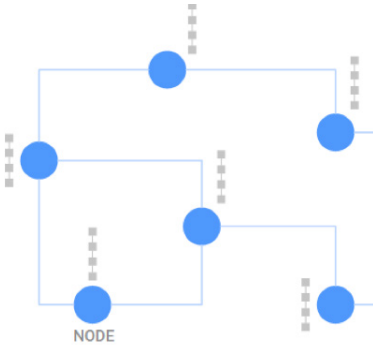
Blockchain technology often gets confused with its specific implementations (such as Bitcoin) but there are certain concepts all blockchains share apart from cryptocurrencies. The name **blockchain** is derived from the data structure that sits at the heart of this technology, consisting of an arbitrary number of blocks cryptographically chained together. A **block** is composed of three principal parts: (1) data, (2) a cryptographic hash of the previous block, and (3) a cryptographic hash of both the data and the hash of the previous block. Doing so not only creates an ordered dependency between subsequent blocks but it also institutes a data dependency between blocks.



If the data contained in any block is changed, the corresponding hash of that block will change, which in turn invalidates the hashes of the subsequent blocks in the chain that depend on it. For example, if the data in block *i* is changed, the hash of block *i* will need to be updated, as well. Updating the hash of *i* results in a change to the hash of block *i* + *1* (since the hash of *i* is an input to the hash of *i* + *1*), which in turn results in a change to the hash of *i* + *2*, and so on.

This allows any outside observer who has a copy of the blockchain to recognize that a block in the chain has been altered — either accidentally or nefariously. Additionally, this determination is nearly trivial, requiring only a single hash comparison.

The key to ensuring data already accepted to a chain remains immutable lies in a **blockchain network** — a peer-to-peer (P2P) distributed network consisting of devices called **nodes**. These nodes are computing devices, such as personal computers, supercomputers, phones, tablets, etc., taking part in the network. Conceptually, every node in the network contains a complete copy of the blockchain, although this is not always the case in practice.



If a node wishes to add a new block to the chain, it must broadcast the new block to all of the other nodes in the network — creating an open network where any node can see the entirety of the blockchain at any given time. Using a predetermined **consensus protocol**, each of the nodes either accepts or rejects the new block. If the new block is accepted, it is added to the front of the chain of each node. Given enough time, all of the nodes in the network will reach **consensus**, having a copy of the same chain as all of the other nodes in the network.

Generally, the protocol is based on the acceptance of the longest chain in the network and therefore, for a bad actor to change an existing block, it must recompute the hash for the changed block and all subsequent blocks and submit it to the network. Using a Proof of Work (PoW) or Proof of Stake (PoS) scheme, this becomes exponentially more difficult the further into the chain a block is altered. For example, in Bitcoin, gaining a controlling stake in the network requires an infeasible level of computing power.

This combination of fixed blocks and network consensus differs from existing technology in a few fundamental ways: It is open, immutable, and does not rely on a central authority. At any given time, any node can verify for itself the integrity of the data in the chain. The distributed nature of the consensus mechanism, combined with verifiability, ensures that no single node (or plurality of nodes) can feasibly dictate to other nodes the correct state of the chain.

Although security is built into the core of blockchain technology, a blockchain and its associated network are only as secure as the underlying technologies used.

This combination of fixed blocks and network consensus differs from existing technology in a few fundamental ways: It is open, immutable, and does not rely on a central authority.

IS A BLOCKCHAIN SECURE?

The security of blockchain technology hinges on a few crucial assumptions. The data structure of a blockchain is secure so long as the cryptographic hashing algorithm used for chaining blocks is sufficiently secure. In many cases, Secure Hash Algorithm (SHA) 256 is used, which is accepted by nearly all institutions, including the National Security Agency (NSA).

A more challenging criterion is the security of a blockchain network, which consists of nodes which cannot be trusted outright. In order to provide a sufficient level of security in a blockchain network, a consensus protocol must be devised such that it is statistically infeasible to gain a controlling stake in the network. The specifics will vary by application but as we will see shortly, the protocols devised for practical blockchain applications are secure enough for major financial institutions and companies to place their trust in them.

WHAT DO BLOCKCHAINS LOOK LIKE IN PRACTICE?

While blockchain technology is theoretically secure, there is no greater testament to the security of blockchains than their widespread use in critical applications and the trust they have earned by some of the largest financial institutions, originating with cryptocurrencies.

BITCOIN AND CRYPTOCURRENCY

In terms of both scale and notoriety, **Bitcoin** is the most recognizable blockchain system, with over 22 million Bitcoin wallets created as of July 2018 and nearly 210 million transactions per day at the time of writing. Bitcoin and other **cryptocurrencies** take the concept of a blockchain and propel it to an entirely different level of practically, allowing users to electronically transfer money without any trusted third party (such as a bank or financial clearing house).

This technological leap is made possible through the use of more sophisticated cryptographic tools, such as asymmetric encryption and digital signatures. Instead of storing arbitrary data in each block, the Bitcoin blockchain stores transactions, which are digitally signed by their originators and thus reflect the authenticity of the transaction. This stringency is made in an effort to thwart the possibility of a bad actor double-spending his or her coins.

With cryptocurrencies, the consensus protocol involves a node proving to the rest of the network it has a sufficient stake in the network (i.e. PoS) or that it has performed a sufficient level of work (i.e. PoW) to submit a new node to the blockchain. In order to solve the double-spend problem while quickly accepting transactions, a balance must be struck between the time required for a transaction to be considered valid and the burden that is placed on an attacker to succeed in fraudulently spending his or her money.

With the creation of innovative companies such as Bakkt (created by the Intercontinental Exchange, ICE, the owner of the New York Stock Exchange — in partnership with Microsoft and Starbucks), many of the largest financial institutions in the world are starting to solidify their trust not only in cryptocurrencies like Bitcoin, but also in the integrity of the blockchain technology that underpins them.

SMART CONTRACTS

A second breakthrough thanks to blockchain technology (first conceived in 1994) is the smart contract. A **smart contract** is a cryptographically signed agreement between an arbitrary number of parties. This contract is then publically submitted as a block in a blockchain network, ensuring that all of the nodes recognize the validity and obligation of the parties involved. Some contracts may involve the exchange of payment for some good or service, while others may require the forfeiture of assets if the contract is breached.

Numerous cryptocurrencies, including Bitcoin and Ethereum, allow users to embed contracts within a transaction. This enables a user to release funds when an objective, codified condition is satisfied. Improvements, such as the Bitcoin Lightning Network, utilize these underlying smart contracts to great advantage. Regardless of the specific criteria, smart contracts are starting to change the legal landscape in many countries around the world and at its current pace, will likely drastically change the way we deal with binding contracts in the future.

FOOD SUPPLY CHAIN

Along with financial and legal breakthroughs, blockchains are helping to improve the way we bring food from the farm to the market. IBM — partnered with Walmart, Dole, Nestlé, and a host of other major food companies — have created a **permissioned blockchain** (data is shared only to those entities given permission by the chain owner) called IBM Food Trust to help trace the origin and state of food from the farm to the shelf.

Although this product is in its infancy, its acceptance has been accelerated by the support of many of the largest growers, suppliers, and regulators in the United States. While there are many secure alternatives to blockchain currently available, these major partners have placed enough trust in blockchain technology to trace the complex journey of our daily food products. It will be interesting to watch how similar permissioned blockchains can enable security while still maintaining control over sensitive data.

CONCLUSION

Blockchain technology is poised on becoming one of the most creatively disruptive inventions in recent decades and its combination of openness, immutability, and distributed network makes it a very promising technology for securing a host of applications. Regardless of the application, blockchain has cemented its place in the world of software security and can enable countless advancements, some of which have not been fully conceived.

JUSTIN ALBANO is a Software Engineer at Catalogic Software, Inc. responsible for building distributed catalog, backup, and recovery solutions for Fortune 500 clients, focusing on Spring-based REST API and MongoDB development. When not working or writing, he can be found at the ice rink, watching hockey, drawing, or reading.

www.edgescan.com

Full-stack Vulnerability Management & Intelligence

"The expertise and delivery of this service has been outstanding..."

— Security and Risk Management, Media Industry 30B + USD

Gartner peerinsights™

4.9 out of 5 Stars

Golden Rules for Vulnerability Management

From visibility to API integration, from result validation to developer support, the items below should all be considered when choosing a vulnerability management SaaS.

1. Web application assessment coverage is king. Depth and breadth of assessment is key to ensuring all risks are detected. Assessment technology needs to be "tuned" to ensure production-safe testing without sacrificing rigor. Poor coverage can result in "false negatives," undiscovered vulnerabilities residing in your system just waiting to be exploited.

2. Full stack security is key, as "hackers don't give a s*#t" where your vulnerability resides. Web and infrastructure security combined results in full stack vulnerability intelligence.

3. A SaaS with DevSecOps pipeline integration capabilities is well-suited to early vulnerability detection. Keep pace with development as change occurs.

4. False positives are an evil waste of time. Your chosen SaaS should deliver validated and actionable vulnerability intelligence. Risk rating and prioritization of discovered risks are also integral features.

5. Choosing a SaaS with a continuous assessment model solves the issue of securing a system in continuous flux. As new vulnerabilities are discovered in the industry, continuous assessment helps detect if you're exposed.

6. Situational awareness is required. Alerting and custom events are key to knowing what matters and when.

7. Your chosen SaaS should have strong API capabilities, enabling you to automate, invoke tests, and consume vulnerability intel. "If I can do it on the UX, I should be able to do it via an API."

8. Visibility is key and asset profiling is paramount. What's my attack surface? If it changes, do I get notified about what matters to me? "We can't secure what we don't know about."

9. Developer support is crucial. Vulnerability management and intelligence is as much about mitigation as it is discovery.

10. A SaaS with a strong metrics capability helps measure improvement: Metrics such as time-to-fix, vulnerabilities by type, layer, and risk all help. "We can't improve what we can't measure."

11. Asset categorization, tagging, vulnerability risk acceptance, and retesting on demand are all important features to aid prioritization and realistic metrics.

WRITTEN BY EOIN KEARY
FOUNDER, **EDGESCAN.COM**

PARTNER SPOTLIGHT

edgescan

Continuous Full-stack Vulnerability Management SaaS. Deep cybersecurity testing, on demand, validated including developer support.

edgescan

CATEGORY
Cloud-based SaaS focusing on vulnerability detection and mitigation.

RELEASE SCHEDULE
Monthly releases

OPEN SOURCE?
No

CASE STUDY

The edgescan SaaS delivers vulnerability management, detection, asset profiling, and support to some of the world's largest organizations as well as SMEs.

A leading global pharma organization uses edgescan to protect their global estate. Using edgescan's continuous vulnerability management model, they keep pace with change by constantly assessing web applications, API's, and cloud hosting systems for vulnerabilities.

When systems are newly deployed, edgescan automatically assesses them, which ensures the security team are made aware of any risk related issues.

When vulnerabilities are discovered, they are validated and rated for risk by expert security analysts who also supply developer/technical support.

With our SaaS technology, scale and efficiency are a given. Integration via the API with Icon's risk dashboards and GRC was simple, delivering near-real-time visibility of their security posture.

STRENGTHS

1. Full-stack security: Web apps, infrastructure, cloud, internal, and Internet-facing systems.
2. Continuous assessment: Continuous and on-demand assessments included.
3. Vulnerabilities validated and risk-rated by experts: False-positive-free Intel.
4. Developer support: Upskilling of technical staff with support.
5. Continuous Asset Profiling: Attack surface profiling - detection of exposed services and systems.

NOTABLE USERS

- Paddy Power Betfair
- An Post
- Multinational mass media & entertainment conglomerate
- UK Government
- Illumio

WEBSITE edgescan.com **TWITTER** @edgescan **BLOG** edgescan.com/company/blog

Why Are So Many Companies Getting Hacked, and What Can Be Done About It?

BY BRIAN KELLY
HEAD OF CONJUR ENGINEERING, **CYBERARK**

QUICK VIEW

01. What are the root causes of the enormous number of data breaches at the enterprise level?

02. Developing secure software is rarely, if ever, taught in schools, and very few companies provide training for developers to create secure software.

03. At the same time, developers shouldn't be expected to be security experts. We have experts for that.

04. It can be difficult to make a company — especially a startup — concentrate on security issues instead of business goals.

If you think you're hearing about a company getting hacked almost every day, that's because you're correct — there were over 1,300 significantly damaging breaches of large businesses last year. That's more than *three per day* on average, and that's only counting the ones that were reported publicly. Unfortunately, hacks are occurring at an ever-increasing rate.

Whether you work in the technology sector or not, you may be rightly wondering when the software profession will fix this problem and, finally, start to secure your personal data. It's an understandable expectation, and fixing it will first require a hard look at the root causes of this rising problem.

INSUFFICIENT EDUCATION

Computing degrees offer students the chance to learn many of the skills necessary to be a professional software engineer. Topics like algorithms, data structures, and discrete mathematics are found in almost every software-focused college course.

However, almost all students in these kinds of courses will graduate without any knowledge of how to write *secure* software. It's simply not taught in most third-level schools. If it is taught, it's only at an introductory level.

Shortly after a computer science student leaves school, they'll likely get a job in a firm where they could be responsible for building software that handles your personal information. Making matters worse, very few companies invest in the training required for their employees to practice secure software development.

Colleges teaching software development should make security a core part of their syllabi. Computer science professors could even perform basic security checks on student programming assignments. If students saw their grades drop if they submitted work that was vulnerable to injection attacks or buffer overflow, we might see good security habits formed before graduation and not after.

RISK MITIGATION IS A TOUGH SELL

Almost all technology companies are trying to accelerate their growth all the time. An early-stage startup will spend its time building new features in its search for a product/market fit before it runs out of capital. Larger publicly traded companies have to meet their quarterly revenue goals or watch their share price dip.

When companies are so focused on earnings and market success, it can be tough to convince them to invest in things that don't directly contribute to increased revenue. It can be a hard sell and convince companies to deal with basic operational inefficiencies — let alone to make significant investments in preventative security.

The potential impact of a hypothetical breach is always debatable ahead of time — will it be just some hard-to-quantify damage to the company's reputation, or will it be real dollars stolen from their accounts? Or, will it be something trivial that doesn't even have to be disclosed to the public? Overconfidence and other human biases can cause the best-case outcomes to be favored and worst case outcomes to be ignored.

This is an economic dilemma, and so it has an actuarial solution that is based on the risk of potential breaches, the negative impact such breaches would have, and the cost of preventative measures. For example, if a company is insecurely storing lots of personally identifiable information (PII) and could go quickly out of business if that PII was stolen, it should be easy to justify investing in security to prevent a breach.

More companies should start analyzing their risk around the economics

of potential breaches and plan accordingly. It can't always be the loudest customer or the prospect of driving a product's feature set.

INCENTIVE MISMATCHES

If someone is successful in penetrating a system, they stand to make much more money through extortion or theft rather than money spent on the hack itself. If they can't hack into a system, they stand to make nothing. The difference is stark — a great reward for a successful hack, a net loss for a failure. Incentives are very strong for a hacker to make even small investments in their attempts.

If a company invests no money or effort in preventing hacks of their systems and nobody ends up hacking them, they will continue to make profits off their regular business endeavors. That's not much of an incentive for them to change course.

With these incentive mismatches, it's no surprise that hackers are far more motivated to penetrate systems rather than companies that are trying to protect themselves. A solid and objective risk analysis will help any company match the right security controls appropriate for them.

TECHNOLOGIES AND TECHNIQUES CHANGE RAPIDLY

For as long as locks have existed, lock pickers and lock inventors have been in an arms race.

Not much is different in the world of software, other than the pace. There seems to be a new JavaScript framework coming out each week, and original hacking techniques that can penetrate websites are appearing just as fast.

Reading any popular security blog (such as Krebs on Security or the CyberArk Threat Research blog) should give you a sense of how difficult it is for software engineers to keep up with the changes in their toolkits or defend against attacks on the innumerable technologies their applications rely on.

The solution to this problem is, counterintuitively, *not* to stop adopting new technology. Rather, a careful and judicious adoption of new technology can help defend against threats. Hackers always prefer going after old technology with well-established vulnerabilities available for them to exploit.

SOFTWARE ENGINEERS ARE WRONGLY EXPECTED TO BE SECURITY EXPERTS

Speaking of the woes of software engineers, the software industry is moving in a worrying direction by expecting application engineers to be security experts, too.

Programmers and engineers already have more than enough difficult technical subjects to master just to be able to create applications that are usable, accessible, responsive, and resilient. Expecting them to become experts in yet another domain is a recipe for failure.

This isn't meant to absolve engineers of their responsibilities in security — they should be aware of common security risks (such as the OWASP Top 10) and how to avoid them, but expecting them to be leaders in the field of security is unfair. True security experts train for years in their field and have a depth of knowledge in that area that few programmers share.

Engineers and security experts should, instead, work together to build tools that allow applications to be built more safely, with less chance of things being done the wrong way. For example, no reasonably skilled engineer will try to create their own version of the TLS security protocols, which are used for secure communications across the Internet. Those protocols are difficult enough to understand, let alone interact with using code. That's why almost all engineers will use an off-the-shelf, trusted implementation of those protocols rather than try to build their own.

The software industry just needs to figure out a similar approach for engineers to use when storing passwords and their users' private information. Many solutions exist, but there are no standards. The best approach will be the one that takes this task off the engineer's plate entirely, allowing them to build application features without worrying about common security issues.

IF A SYSTEM CAN BE USED, IT CAN BE HACKED

The only safe computer system is one that is disconnected from the network, unplugged, thrown in a dumpster, put through an industrial shredder, loaded into an incinerator, and turned into ash.

As glib as that statement may seem, it is, sadly, quite true. If data exists on a hard drive, there's a chance that it can be accessed by an attacker. That chance increases greatly if the system is powered on and is reachable on a network. Even if a computer system is completely powered off and physically disconnected, a human could still be bribed or blackmailed into accessing it by a determined hacker.

There is no magic solution to this problem, unfortunately. As a society, we have accepted this risk because of the benefits afforded to us by turning on all of our computers and making them work for us. All we can do is work to keep them secure and train our fellow humans how not to get duped or tricked into helping hackers gain access.

So, let's get our software engineering students well-trained in the basics of security, improve the technology that makes application development safer, and teach companies to better analyze their risks. The safety of our data depends on it.

BRIAN KELLY is Head of Conjur Engineering at CyberArk, where he focuses on creating products that add much-needed security and identity management to the landscape of DevOps tools and cloud systems. Brian is passionate about software, building teams, cybersecurity, and DevOps.

graylog

GRAYLOG LEADS IN SECURITY INCIDENT AND EVENT MANAGEMENT

Leader of this year's Info-Tech Data Quadrant, Graylog provides the complete software experience to navigate the threat landscape.

See why at www.graylog.org

Best Practices in Web App Security Logging

With log frameworks not including a "security" severity level, it's no wonder custom web apps often don't collect security-related data or aren't configured to meet the needs of threat hunters or compliance officers.

We get it. Collecting, storing, searching, and analyzing logs can be resource intensive.

In fact, it's difficult enough that 'insufficient logging and monitoring' makes OWASP's Top 10 Web App Security Risks. So what should you be logging? At a minimum: a system-wide timestamp, account/user ID, source IP, event outcome (success or failure), and description. But simply collecting this data isn't enough.

Conveniently, DevOps is the perfect bridge to the security team. Start by separating security logs. That way you can focus on making them tamper-proof, tightly controlled, and actively monitored while minimizing performance impacts. Next, make sure security events are closely associated with the user performing the action, not the object class. Finally, tune log collection to reduce signal-to-noise ratio, keep sensitive data like passwords or PII out, and set alerts for any time a security log stops.

We get it. Collecting, storing, searching, and analyzing logs can be resource intensive.

To accomplish all this, you really need a centralized log management tool. We, of course, think Graylog is the best. But, no matter who you choose, you should be able to collect all the data from all the logs and store that data efficiently. It should scale up to meet peak volume. It should be easy to manage and configure any type of log collector, analyze the data, search for specific information, and set up alerts and reports.

And, most importantly, it should be a great company to work with, making your life easier and adding value to the business.

WRITTEN BY LENNART KOOPMANN
CHIEF TECHNOLOGY OFFICER, GRAYLOG

PARTNER SPOTLIGHT

Graylog Enterprise

Graylog is a leading centralized log management solution built to open standards for capturing, storing, and enabling real-time analysis against terabytes of machine data.

CATEGORY
Centralized Log Management

RELEASE SCHEDULE
Minor releases approximately 3 times per year, one major release per year

OPEN SOURCE?
Yes

CASE STUDY
Critical Start is the largest cybersecurity integrator based in Texas and the fastest-growing in North America. They were looking for a way to provide log collection and analysis for their customers at a lower total cost of ownership when they selected Graylog. Must-have requirements were multi-tenancy, ability to deliver Critical Start's 20 core security use cases, and deployment to AWS in minutes with immediate ingestion of standard log data. WE are excited to have met this challenge and continue to build on this relationship for many years to come.

STRENGTHS
1. Faster analysis
2. Easy-to-use platform
3. Simple administration
4. Lower cost
5. Full data capture

WEBSITE graylog.org **TWITTER** @graylog2 **BLOG** graylog.org/blog

Securing Embedded Systems

QUICK VIEW

01. Risk changes based on your development strategy.

02. More control isn't necessarily better.

03. You can pay to lower your risk.

BY CHRIS LAMB
CYBER-SECURITY RESEARCH SCIENTIST, **SANDIA NATIONAL LABORATORIES** AND ZONE LEADER, **DZONE**

Embedded systems come in a wide range of flavors and can be surprisingly complicated and difficult to secure. I'm going to look at three types of basic designs, describe why they're difficult to secure, and what you can do to secure these systems.

One of the reasons we have so many different kinds of embedded system designs is the wide range of embedded systems people use. We have devices that are considered embedded systems running on hardware ranging from small PIC processors to ARM systems to fully featured systems-on-a-chip, not that different from what we see in cell phones. The scale and feature sets of these chips directly impact the complexity of the controlling software and leads us to a fairly straightforward taxonomy of possible software designs.

One way to look at the kinds of software needed to run these types of devices is to divide the options into categories related to the amount of software you need to write to make the device do, well, whatever it is that you need it to do. Simpler systems may not use any external code, or you may be in a position to write all the code that runs on the device, though this is becoming more rare today as embedded systems become more powerful. Other systems may use third-party open source components, like embedded Linux, or Das U-Boot, or something similar. These kinds of systems are becoming the standard in the Internet of Things space today, as they're easier to manage from a revenue and licensing perspective, they're less expensive to build, and they require less training for engineers to be effective. Finally, we see high-end products using either proprietary or fully featured third-party real-time operating systems. These kinds of systems tend to use software like VxWorks. In these cases, engineers are still responsible for the final system, but the operating system vendor shoulders much of the risk of delivering insecure systems, and usually provides extensive support for their products as well. So, let's take a look at these systems to see where the risk lies and how you can deal with it.

BARE METAL

Let's call the first category of systems bare metal, as that's what you're pretty close to when developing on this kind of hardware. In these cases, you're responsible for everything — memory management, peripheral management, thread management, and so on. Fortunately, these kinds of systems are relatively simple, so there's not too much to get hung up on. But trust me, there's enough.

These kinds of programs usually have a single entry point at a well-known memory location (or something similar to this). They also have nothing in the way of pre-existing code. The good news is that nobody else's mistakes may make your system less secure. Unfortunately, that's also the bad news: you take on all the risk.

These projects lean very heavily on software development best practices, and don't depend on system knowledge at all (as there's really no system in place other than the one you've designed). In order to ameliorate your project risk, you need to implement strong code practices to ensure secure code production (you'll need to do the same for other attributes, like performance and usability, of course). This is pure software engineering practice — you'll need to implement code reviews, understand secure coding practices, use automated analysis tools to ensure appropriate memory handling, and have a clear understanding of your attack surface.

There's a wide variety of secure code auditing tools out there, and two of the most common tools (for the C programming language, at least) are ITS4 and RATS. Both of these tools are fairly simple. ITS4 can generate a

fair number of false positives, and neither tool will detect design flaws, but they will find things like buffer overflows and race conditions. You will also need to avoid, as much as possible, unsafe standard C library functions.

There's a wide variety of dynamic analysis tools you can use, as well. Valgrind is still one of the strongest dynamic analysis tools available today. Another trick that more experienced engineers will use is to build a set of macros that will allow you to compile your code for x86-based processors. Embedded processors very rarely use the x86 architecture as it's just too expensive for these kinds of applications, so you'll usually be targeting ARM or MIPS architectures (or others, like Microchip PIC microcontrollers). Some of these architectures are remarkably different from x86 and may in fact use extensions or alterations to the C language (especially for memory management). Being able to compile this code for x86 makes dynamic analysis something you can integrate into your usual build system rather than something you need run on development boards.

Overall, you have the most control over what happens on your system in bare metal configurations. You also have the most responsibility.

OPEN EMBEDDED OPERATING SYSTEMS

Open embedded options are becoming more and more common today, especially on mass-produced embedded products like smart bulbs or programmable power outlets. And they're becoming more popular for good reason: they're less expensive to distribute, licensing costs are non-existent, they are simple to manage, and they can offload much of the complexity of embedded programming. On these kinds of systems, you can develop your application code in a familiar environment and run that code, in production, in familiar ways. Embedded Linuxes handle all of the more complex system management tasks for you. All you need to do is build and install a couple of user space programs, usually one that runs as a daemon process and one or two for system debugging.

Clearly, you're still writing code, so the caveats around developing for bare metal systems still apply. But now you are also responsible for securing the operating system you're using, which can end up requiring a significant amount of know-how.

First, you need to understand the attack surface of the entire system, not just the code you're writing. You also want to minimize this as much as possible. In situations where you have complete control over the code running on the system, this can be significantly easier. When you're configuring a true embedded OS, however, things can get a bit fuzzier. You need to understand exactly what is installed in the OS image you're using, and you need to ensure that it only contains services you will need and use.

This means that you need to ensure that SSH is removed, as well as FTP, TELNET, and the like (unless you have a specific reason to include it). You may need to embed some kind of web server, but if you do, you need to know exactly how it works and install the most recent (and hopefully bug-free) version you can. You need to be very deliberate in how you configure the system, and not blindly accept any system defaults. You absolutely need to know how the system works and what all of the components are doing. You are, after all, responsible for the entire product if you use this approach, not just the code you wrote.

PROPRIETARY EMBEDDED OPERATING SYSTEMS

Proprietary embedded systems have the same characteristics as open systems, with some slight twists. To begin with, the operating system and application configuration can be significantly different than what most engineers are familiar with. VxWorks, for example, uses this kind of configuration: the operating system treats application code with the same priority as kernel code, and VxWorks firmware images are distributed as a single binary blob, without the same kinds of familiar filesystem and kernel constructs we see in embedded Linux. This makes sense — after all, VxWorks is a real-time OS, and is generally used with non-commodity devices like higher-end industrial and manufacturing equipment.

That said, these kinds of operating systems do have significant advantages. Although they can be more troubling to distribute from a licensing perspective, they generally have much better support, even though you still shoulder the risk from a security perspective. The companies that support these kinds of OSes are large and generally trustworthy (e.g. WindRiver, the company that sells VxWorks, is owned by Intel).

These systems usually have high-quality training available as well. While the same concerns with open OSes exist in these kinds of systems with respect to system hardening, code auditing, and review, you can expect much more support in your efforts to secure your systems.

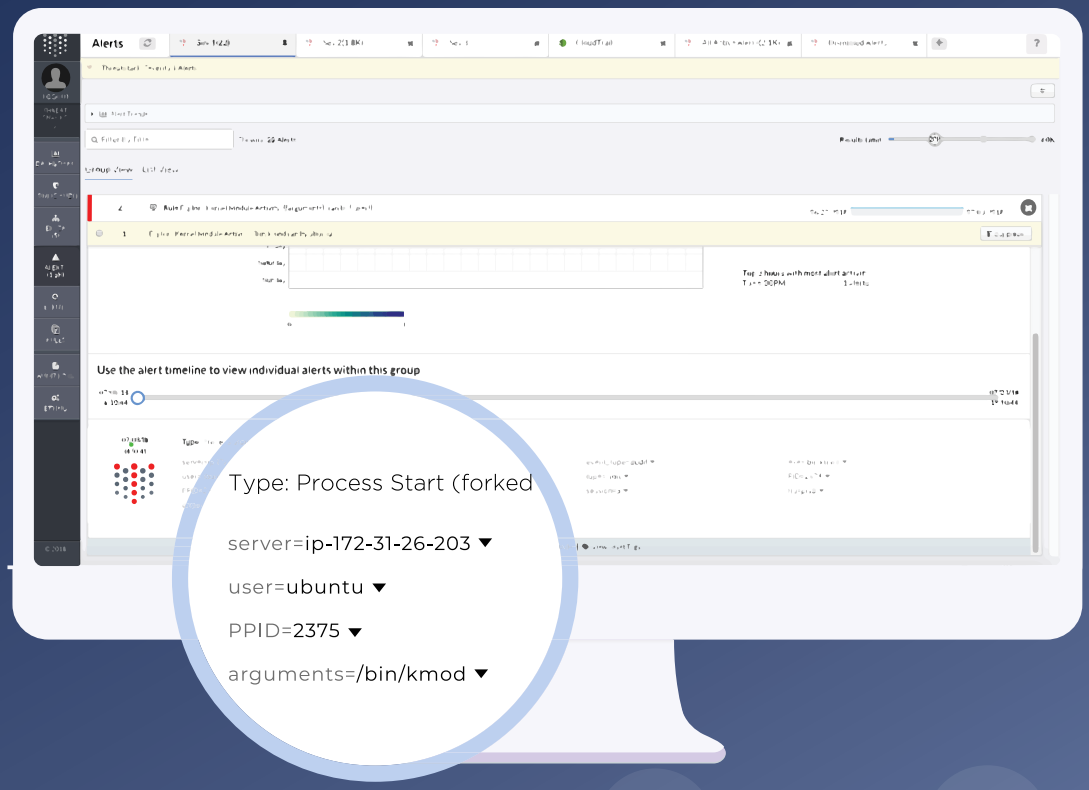
WHAT DOES THIS MEAN FOR ME?

We've reviewed a taxonomy of embedded systems and covered how you can secure them from an engineering perspective. While it might not capture all the systems you might need to build, it'll cover most of them. So, what should you do?

Well, you'll likely need to make those decisions based on more than securability. The device you're building, the market you're targeting, and your team's expertise are more important drivers in selecting a particular approach. For example, you're not likely to use VxWorks with a commodity IoT device. Likewise, you shouldn't build your own operating system for an expensive real-time programmable logic controller either. With that in mind, these guidelines can help you secure your device no matter the hardware you've chosen.

CHRIS LAMB currently serves as a Cybersecurity Research Scientist with Sandia National Laboratories. He is also a Research Assistant Professor affiliated with the Electrical and Computer Engineering department at the University of New Mexico. Currently, his research interests center around industrial control system cybersecurity, machine learning, artificial intelligence, and their intersections. He is a TOGAF 9 Certified Enterprise Architect and a Certified Information Systems Security Professional (CISSP) through the International Information Systems Security Certification Consortium.

threat stack



Ask Your Security Person.

THREATSTACK.COM

How AI Takes Cybersecurity to the Next Level

BY FRED JACQUET
TECHNOLOGY EVANGELIST

QUICK VIEW

01. Companies take 100-200 days to detect a data breach — how do we evaluate the compromising, stealing, spying, and vandalizing that could occur during that time?

02. Business leaders are recognizing the financial impact that cyber threats place on them; it is forecasted that worldwide spending on security hardware, software, and services will reach up to $120 billion in 2021.

03. At the cutting edge of innovation, cybercriminals implement social and technical engineering that benefits from the weaknesses of both human and IT infrastructures to steal sensitive data.

In April 2018, a social media giant had to notify nearly 90 million users that their personal data might have been *"improperly shared."* It was a colossal security breach.

In October 2017, several ten of thousands of harmless cameras were used to produce a massive distributed denial-of-service (DDoS) attack that dramatically decreased websites' function or even took them down by sending them billions of requests. Among the renowned companies involved were Amazon, Netflix, and Twitter.

"Cyber crime is the greatest threat to every company in the world." — ***Ginni Rometty, CEO, IBM***

Back in 2016, there was a new malware every 4.6 seconds — in 2017, this evolved to 4.2 seconds. Symantec reports that 1 in 13 URLs they analyzed were malicious in 2017, while this number was 1 in 20 in 2016. Cybersecurity Venture predicts that cyber criminality will cost the world $6 trillion annually by 2021, which will make it *"more profitable than the global trade of all major illegal drugs combined."*

From a global perspective, the IT market that covers every aspect of cybersecurity is estimated to reach $170 billion by 2020 — more than twice the 2015 number.

Companies typically take 100-200 days to detect a data breach. Imagine the extent of the effects in terms of compromising, stealing, spying, and vandalizing that could be operated in that timeframe, causing total damages costs to grow to historic heights. That's why more investment is needed in security tools to help companies understand, anticipate, detect, and fight cyber attacks.

All of this culminates to Warren Buffett's claims that cyber attacks are the biggest threat to humanity.

WHAT ARE CYBER THREATS?

Cyber criminals are leveraging technology components including networks, operating systems, and hardware to operate malevolent code and algorithms for the purposes of breaking into company systems and data — possibly to compromise the business, steal money, hold the company ransom, or threaten the business.

We can group cybercrime into three main categories:

1. Infractions that threaten data integrity, including hacking, breaches of automated data processing systems, unauthorized processing of personal data, and credit card fraud.
2. Infractions using information system contents for malicious purposes like child pornography, incitement to terrorism, and racism.
3. Infractions facilitated by information and communication technologies, e.g. online scams, money laundering, online fraud, and ransoming.

Additionally, the term "cyber attack" covers a variety of other categories, including:

- System infiltration
- Identity or intellectual property theft
- Unauthorized access
- Fraud
- Extortion
- Distributed denial-of-service attacks (DDoS)
- Viruses, phishing, spamming, spyware, trojans, etc.

When cyber criminality appeared at the end of the 90s, it was mainly about violating privacy or confidentiality. Nowadays, cyberpirates make up organizations that might even be supported by some governments. Far from bored teenagers surrendering to the sheer pleasure of cyber vandalism, cybercriminals, hacktivists, and even nation states are now seriously threatening to extort money at the international level.

See, for example, the massive computer attack WannaCry that hit 300,000+ computers in 150 countries in May 2017. The purpose of this virus was to take personal data hostage by encrypting data and making the data inaccessible. The victims who didn't have backups faced two choices: pay the ransom in Bitcoins for decryption or lose their data.

MARKET ANALYSIS AND IMPORTANCE

In their *Predictions 2018: Cybersecurity* report, Forrester set the stage for the international security situation. They predicted that companies will have to cope with four major trends:

1. Geopolitical tension
2. Ubiquitous connectivity
3. Digital transformation initiatives
4. The continuing shift toward the data economy

According to them, all these phenomena have given rise to global cybersecurity concerns.

The IDC forecasts that worldwide spending on hardware, software, and services aimed at security will reach $120 billion in 2021 (it was $83.5 billion in 2017).

Exploring security and risk management trends for 2018, Gartner spotlights some dominant tendencies including:

- **Awareness**: Business leaders are starting to fully recognize the financial risk that cyber threats place on them and are now taking a closer look at their security strategies.
- **Regulation**: To minimize risks, companies are increasingly complying to regulations. They are embracing security policies not only as a way to protect their assets but also as a business differentiator.
- **Cloud**: By developing an enterprise cloud strategy, companies are establishing increased guidance strategies that allow for better requirement analysis, more sophisticated architectural planning, and more flexible risk acceptance processes.

THE SCOPE OF CYBERSECURITY

Part of the cyber crime problem is that technological innovations and developments — such as the cloud, IoT, mobile networks, smart objects, autonomous devices, and sensors — are weakening the security of companies that are typically already inadequately protected (even big guys such as Facebook or Sony Pictures, for example) by opening wide access to their data and systems.

Cybercriminals implement social and technical engineering that benefit from both human and IT infrastructure weaknesses to steal sensitive data for commercial espionage and to ransom individuals.

"We cannot escape the immutable fact that humans and machines complement each other."
— Peter Firstbrook, Research Vice President, Gartner

Although the list of threats is wide and is rapidly changing, some risks are spectacular according to different perspectives — outstanding technical ingenuity, impressive amount of victims, wide geographic spread, or dramatic data or money volumes involved.

Let's see two examples.

1. RANSOMWARE

The principle of ransoming, as the name suggests, is to proceed to extortion. Pirates use encryption methods to block critical data or to lock or compromise entire IT systems in order to extort money. Without going into the technicalities of it, the pirates get into targeted systems via email links, social network messages, and malicious software such as worms or trojans.

There are loads of means to protect yourself from threats of this kind, including backing up data on a regular basis, configuring firewalls properly, providing collaborators with training programs, leveraging up-to-date cybersecurity technologies, or conducting recurring intrusion tests and vulnerability assessments.

2. DDOS

A DDoS attack is a type of denial-of-service attack where a compromised network of systems is used to target a single system, application, or network so that it can be made unavailable to the intended user. The principle is pretty simple. A web server is overwhelmed with illegitimate requests from a single system, causing the server to crash or slow down. As a result, legitimate requests are not processed by the web server, which prevents authorized users from accessing a website or computer.

They can be basically divided into three categories:

1. **Volume-based attacks**: Flooding the bandwidth of a targeted site.
2. **Protocol attacks**: Exploiting server resources or intermediate communications equipment (e.g. load balancers and firewalls).
3. **Application layer attacks**: Overexerting specific functions in order to disable them.

To evaluate the business impact (from the victim's perspective) of a DDoS attack, we should take into consideration that up to 50% of these attacks last less than 24 hours. With an estimed cost of $40,000/hour, an ordinary attack would cost about $500,000. And let's keep in mind

that costs are not just a matter of money directly stated; attacks might also cost the company organization, reliability, and reputation.

THE WEAPONIZATION OF AI

Unfortunately, it is feared that even though security firms are using machine learning, neural networks, advanced analytics, and AI technologies to prevent, anticipate, and fight cyber attacks, there is a real danger that hackers will also use these technologies to engage in the AI-driven arms race.

In February, the University of Oxford and University of Cambridge team voiced their concerns that cyber pirates could use AI tools to take control of autonomous vehicles and turn them into weapons.

"In cybersecurity, AI can automatically identify potentially malicious software behavior, attack vectors, and related anomalies in real-time, allowing a continuously adaptive defense mechanism to identify and shut down intrusions faster and easier than ever before."
— ***Chris Ganje, CEO, Amplyfi***

AI TAKES CYBERSECURITY TO THE NEXT LEVEL

Just like predictive analytics, cybersecurity will derive the maximum possible benefit from artificial intelligence. Companies can't afford to rely only on their manpower and human knowledge to resist the growing flood of cyber attacks. Companies are needing to spend more time on security, and staying up-to-date with the evolving technologies and methods is becoming difficult.

The risk, then, is that cyber pirates will increasingly compromise companies and individuals.

AI can help save time, increase production efficiency, and improve processes to fight against cyber attacks.

A Ponemon Institute study found that human supervision will remain a requirement, though they did establishe some key findings demonstrating the value of AI. AI could:

- Help cut costs
- Minimize data breaches
- Improve productivity
- Provide deeper security
- Support identification and authentication technologies
- Help identify application security vulnerabilities
- Save investigation and detection time

Especially with respect to reliability and time gains, AI brings genuine added value in domains such as:

- Organizing cyber strategy approaches
- Capturing actionable intelligence
- Investigating and detecting application vulnerabilities
- Investigating actionable intelligence
- Cleaning and fixing damaged/infected networks, applications, and devices

Security companies are leveraging the value that machine learning could bring to security tools for various industries. For example, ML helps break down the complexity of automating complex processes for detecting attacks and reacting to breaches. The challenge now is to deliver tangible results in processes including detection, analysis, and anticipation of attacks — to be highly reactive or responsive or, even better, proactive. While traditional computing requires some knowledge of the attack characteristics, machine learning implements technological resources that can discover, adapt, learn, and improve themselves to identify and counter attacks.

Automation, machine learning, and AI are empowering the combination of human and technical methods engaged in cyber defense. Harnessing the real potential of artificial intelligence implies that the methodologies and tools with which we address security challenges must be reassessed.

Cyber criminals are mainly taking advantage of software, network, and infrastructure weaknesses. AI-based security tools should be able to search out and fix these flaws and vulnerabilities. Also, they should be designed to defend against incoming attacks when a breach is opened in a targeted system.

MIT's Computer Science and Artificial Intelligence Laboratory (CSAIL) and the machine learning startup PatternEx are working on the predictive perspective of cyber defense. They have developed AI2, an artificial intelligence platform aiming to predict cyber attacks far better than current systems by incorporating input from human experts in a continuous stream. They designed Active Contextual Modeling, a continuous loop of feedback between the human analyst and AI system. It has a recall rate over 85%. This unique combination of machine learning modeling and human expertise enrichment is 10x more efficient than learning-only solutions.

CONCLUSION

According to Symantec, up to 980 million people across 20 countries were affected by cyber crime in 2017. Victims of cybercrime lost a total of $172 billion. Cyber criminality is very lucrative. Nicole Eagan, CEO of cybersecurity firm Darktrace, points out that although we are in the early stages of hackers using AI themselves as offensive tools, that day will eventually come.

But thankfully, today, the biggest companies in the cyber defense field are always right there at the forefront when it comes to efficiency, technology, and innovation.

FRED JACQUET has been working in the Information Technology industry for over two decades. His experience results from great years within IT companies leading the data intelligence market, as well as assuming roles including Solution Architect, CTO, and Evangelist. His preferred playground is data-oriented, i.e. analytics, IoT, and AI.

bmc
Multi-Cloud

Automated security for all your clouds.

Learn more at **bmc.com/cloudsecurity**

Herding Cats in the Cloud – Transforming and Automating Cloud Security

Enterprise spend on public cloud, projected to eclipse $160 billion in 2018 [IDC], continues to expand at breakneck speed. Cost efficiency, scalability, and flexibility are simply too attractive to ignore. Yet, last year, over 2 billion public cloud records were exposed. Why is public cloud security a struggle for many? Many factors combine to make public cloud security a perfect storm.

The velocity of change which the cloud enables creates security problems which exceed human ability to keep pace. The sheer number of distributed Dev teams spread across the enterprise, while accelerating innovation, presents a cloud governance challenge. Hundreds of accounts, across multiple cloud platforms, and thousands of microservices lead to a "Wild West" mode of cloud operations. While the shortage of cybersecurity talent persists, we have accelerated our ability to break glass without anyone to pick up the pieces. Intelligent automation is required if we are to make cloud security as agile as the software developers driving digital transformation.

Containers have caught on, as the promise of "build once, deploy anywhere" is oh so tempting, even if container security expertise is even more scarce than cloud security talent. SecOps and Cloud Operations alike know that consistency, governance, and especially automation are critical if security is to be as agile as the business.

And this is all only the beginning. By some estimates, only 10% of workloads which will be migrated to the cloud have done so. While enterprise usage of public cloud services from the likes of Google Cloud, AWS, and Azure have skyrocketed in recent years, much more growth is yet to come. Security in the cloud is no less important than on-premises. In fact, the speed of change, scale, and limited visibility arguably make it even more critical. The enterprise must find a way to securely manage its public cloud resources in a way which increases visibility and consistency, leverages automated assessment and remediation, and does not hamstring accelerated innovation.

WRITTEN BY RICK BOSWORTH
DIRECTOR, SOLUTIONS MARKETING, BMC

PARTNER SPOTLIGHT

TrueSight Cloud Security

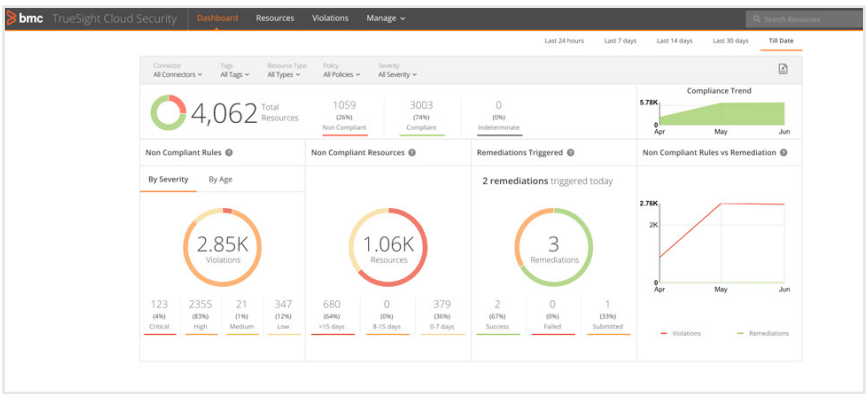
TrueSight Cloud Security is a cloud security and compliance SaaS solution which automates the assessment and remediation of cloud resources and containers.

CATEGORY
Cloud Security Posture Management

RELEASE SCHEDULE
Continuous

OPEN SOURCE?
No

TRUESIGHT CLOUD SECURITY DASHBOARD, VISUALIZING ENTERPRISE-WIDE CLOUD SECURITY POSTURE



STRENGTHS

1. Automated security assessment and remediation of cloud resources
2. Full-stack container security, including Kubernetes, Docker, and host OS
3. Enterprise-wide visibility of cloud security posture
4. Extensive library of ready-to-use security and compliance policies
5. Customizable and extensible

WEBSITE bmc.com/cloudsecurity

TWITTER @BMCSoftware

BLOG bmc.com/blogs

Beating the Cost, Time, and Quality Equation With OWASP ZAP Automation

BY JAMES MCDERMOTT
INFRASTRUCTURE ARCHITECT, **DROPSOURCE**

QUICK VIEW

01. The project management triangle is fundamentally applicable to managing security for a project.

02. Calculating the depth you should implement to be cost-effective can be calculated using the triangle and the risk equation. While there are other ways which are more granular, its a great starting point.

03. Calculating every risk can be very expensive, but there are a number of places that are already well defined to get started complying with best practices, such as the OWASP Top 10 for web security.

The project management triangle has been around since at least the 1950s and represents the idea that quality is the result of three elements:

- **Budget (cost)**: What resources do you have to build the product?
- **Deadlines (time)**: When does the product need to go to market?
- **Scope**: What are all the required features?

It's a way of showing that changes in any fundamental element will directly impact the resulting quality for better or worse. The triangle shows the trade-offs that can result in the same quality with different elements... like more money, more time, or fewer features.

The relationship between the contributing factors of the triangle explains why tools like Jira, Crucible, Jenkins, Ansible, and other various forms of automation are so valuable. Each one of these tools provides a modifier to the elements of the triangle that then alter the quality equation for more time, lower costs, or faster completion, and result in higher quality.

SECURITY IS TREATED AS A SECOND-CLASS CITIZEN

Traditionally, security isn't considered outside of the most basic of requirements unless you have a product specifically geared towards it, yet any modern developer will laugh if you say you don't expect unit tests to be part of their development cycle.

Kent Beck stated, "Make it work, make it right, make it fast," which has been both the basis for development of many fine software projects and the subject of much conjecture as to the interpretation of "make it right" and "make it fast," as there is no explicit "make is secure" step.

If you ask the development group to ensure there is a secure review of their code, first, they will raise eyebrows; then, they will likely push back and assume it costs too much. So, let's take a look at why that's simply not true.

"They are going to get in — get over it." — Michael Hayden, former head of the NSA

Nearly daily, we see breaches impacting companies from identity loss, to potentially altering world democratic elections. Memorable breaches from 2017 include:

2017 BRANCHES	CONSEQUENCES
Equifax	143 million accounts 200,000 credit card numbers
Verizon	14 million users' account information
Uber	57 million users' personal data
RNC	200 million users' votion data

The list continues to grow. Given the impact breaches have on public and corporate trust, we must determine if anything can be reasonably done to prevent these issues.

In most cases, there is. Equifax had terrible password and access policies as well as unpatched software; Verizon and RNC had misconfigured AWS issues. Other exploits included misconfigured email servers and other common hacks.

There are plenty of ingenious hacks out there that are very hard to protect against and take serious consideration... but largely, the biggest hacks seem to fall into the same categories of **known and preventable threats**.

Public applications *will* suffer attacks. The resulting costs of a successful exploit can be reputation or cost, or can extend into ramifications that ripple through the organization. We must consider the costs imposed on the quality triangle. Saying it's too expensive isn't realistic anymore. Therefore, in today's environment, the definition of **"make it right" is simply not satisfied without a security audit**.

The triangle can identify the consequences to the project, but to identify where to invest resources, we need to quantify the threats. Identifying which threats to address requires establishing a method to quantify the risk consequence versus the cost to implement it.

QUANTIFY THE COST TO THE QUALITY TRIANGLE

Consider two potential public applications with external APIs. The first is a public WordPress server providing a blog of recipes online. The second is a fitness app with medical history, as well as access to medical records that would make it trivial to impersonate a user's identity.

The fitness app will require extensive security testing, including all the bare minimum requirements for PCI and HIPAA compliance. The recipe blog would only require a solid assurance that the data is not lost or corrupted.

Identifying the relevant threats to the application is the first step. There are a few databases that collect various types of risks, including a database that is specifically geared towards web applications called the OWASP Top 10.

The OWASP itself is a worldwide not-for-profit charitable organization focused on improving the security of software. Their mission is making software security visible so that individuals and organizations are able to make informed decisions.

The importance of protecting each risk can be derived by identifying the cost of resources by using the Top 10 list to identify the top vulnerabilities and then quantifying the impact to the quality triangle.

For each threat:
risk value = threat * vulnerability * consequence (ref).

A simple example using the Top 10 list applied to the above examples of the medical records database server vs. the public recipe database gives an idea of how the value of risk changes with the formula.

Medical records database:

THREAT	VULNERABILITY	CONSEQUENCE
Network attacker	Buffer overflow	Escalated privileges — stolen records
Network attacker	Injection	Corrupted data or escalated privileges — stolen records
Any system or non-system user	Unlogged breach	Untraceable attack, possible loss of all records
Network attacker	XML entities	Privilege excalation — stolen records

Breaking down the risks and consequences shows there is an obvious first place to fix: the unlogged breach. The consequences could result in a possible loss of all records, each of which carries a required fine for loss. The other threats are at least more limited — not to mention the loss of consumer confidence when the press asks how the breach occurred and you shrug and say, "Dunno, sorry."

Let's take a look at the same exploits against the recipe database.

THREAT	VULNERABILITY	CONSEQUENCE
Network attacker	Buffer overflow	Escalated privileges stolen recipes! (Which are public)
Network attacker	Injection	Corrupted data or escalated privileges, e.g. people making bad food, reputation
Any system of non-system user	Unlogged breach	Untraceable attack, possible loss of all no backend up public recipes
Network attacker	XML entities	Privilege escalation — stolen recipes!

Obviously, the consequences are far less severe. The log system would still be very high on the to-do list but the severity of the consequences is substantially lower. In fact, injection or blocking buffer overflows might even be deemed more pertinent, as recipes showing up with small changes or something like wrong ingredients would be far more damaging to the site's brand and value. Prevention versus mitigation.

In the example of the recipe database, understanding the risk/consequence relationship might allow the mitigation of sufficient risk by simply setting up a solid remote backup system —- whereas, with the medical records database, each risk is attached directly to a real dollar value (fines) and therefore each potential threat will need to be identified and quantified.

Identifying every possible threat, especially in a large organization, is a daunting task. There are more granular methods of identifying specific risk and criticality, which might be relevant depending on the application. Formulas for identifying attack costs as well as likelihood

vectors are available which focus more closely on things like attack trees, fuzzy logic, CVSS (common vulnerability scoring system), and organizationally dependent impact. Realistically, this job is getting more complex every day.

IT'S CLEAR "WHY," AND NOW THE "WHAT" HAS BEEN ESTABLISHED, BUT THERE IS ALSO A BETTER "HOW"

Tools like Jenkins and Ansible make it possible to maintain or lower the cost of portions of the quality triangle by automating repeatable tasks. Many languages are developing along a similar paradigm, such as PHP with composer or Node with npm, by providing systems that leverage the effort of multitudes of developers to make it easy to create, review, and implement new functionality.

Low-code platforms (like the one I work on) can extend this paradigm to generate full source code, instead of just libraries, and can in some cases be built using the common interface of a web-based IDE. Cool!

These tools and mechanisms are all rooted in the quality triangle, enhancing the quality equation by modifying one or more of the contributing factors to create a better product faster and more feature-rich for lower overall cost. It should come as no surprise that tools are evolving in the security space to provide the same benefits.

The same organization that provides the top ten threat list project (OWASP) also provides a project called the ZAP (Zed Attack Proxy) server. Among other things, ZAP provides the means to test for these top ten threats, as well as many others. ZAP also creates a vector that can be included in the automated components, such as Jenkins, of your development cycle.

By applying the open-source principle to security and automation, everyone benefits. (Expressed as Linus Law by Eric Raymond as, "Given enough eyeballs, all bugs are shallow.") Exploits will be exposed and fixes can be automated into tools that help everyone learn how to protect against them.

ZAP then provides the ability to apply the same benefits Jenkins or Ansible does to the development process to the security aspects of the quality triangle, automating the regular testing of numerous vectors of exploits with little cost to the development team. It's higher-quality and comes at the same cost, and is arguably even cheaper due to less after-the-fact mitigation.

ZAP has a number of modes to run in, including as a desktop utility or a server, and also supports a robusst scripting mechanism the includes support for any scripting language that supports JSR 223 such as Python, JavaScript, or Groovy. This makes ZAP perfect for doing everything from lightweight passive scanning to granularly defined scripted testing.

The ZAP server can also run active or passive scans. This provides the ability to test your site without actively attacking it, which is important if you do not own the website in question. This also makes it possible to integrate multiple environments, from dev to staging, before reaching production on machines, and makes it easy to re-provision to ensure the active scans won't do any serious damage (though it would still be better to be found before someone else does).

A REALLY GOOD TOOL ISN'T JUST A TESTER, BUT ALSO A TEACHER

Tools such as Findbugs, Sonar, and PMD help teach developers by exposing bad patterns in design, logic, or structure. This is important as on-the-job training for developers that actually evolves as it is being used (assuming it is continually updated). The ZAP server fulfills this requirement as it gives details about what happened and helps developers recognize what types of activity will cause a potential threat.

In summary, security needs to be a first-tier concern, and if we leverage automation tools and risk assessment practices that are already established, we can develop a strategy to provide for the new requirements with minimal impact to the end quality of the project while managing reasonable costs and scope. Additionally, we can provide ongoing training in the clearly growing and important area of security with minimal overhead.

THAT IS HOW WE ZAP THE TRIANGLE

There are many other tools worth investigating based on your application's needs. For example, there are more ways of assessing risk that provide a much more granular level of risk assessment. One such list worth investing some time into is the CVSS, which provides tools and a training system on managing assessments. Keep in mind that there is no "silver bullet," and we need to be able to learn and adapt. The universe of development is one that has just undergone its "big bang" relatively recently and is rapidly expanding.

RESOURCES

- fortune.com/2016/05/14/cybersecurity-risk-calculation
- azquotes.com/author/31849-Kent_Beck
- isaca.org/Journal/archives/2014/Volume-4/Pages/JOnline-An-Enhanced-Risk-Formula-for-Software-Security-Vulnerabilities
- checkmarx.com/2017/12/31/recap-biggest-data-breaches-2017
- owasp.org
- owasp.org/index.php/ZAPpingTheTop10
- jcp.org/en/jsr/detail?id=223
- owasp.org/index.php/Category:OWASP_Top_Ten_Project
- Startup Security 101
- Improving Build Times With Jenkins Pipelines
- Ethereum Scripting – The Future of the Web (new.0)

JAMES MCDERMOTT is a 20+ year high-tech veteran helping notable startups such as Red Hat and Lulu in their early days as well as tech giants such as Teradata with their global data, analytics, and global infrastructure/services. James joined Dropsource early on in 2015 after quickly recognizing the incredible potential of the platform as a disruptive technology destined to make positive changes.

diving deeper

INTO SECURITY

twitter

 **@malwareunicorn**

 **@jeremiahg**

 **@campuscodi**

 **@Lisa_CyberSN**

 **@dannyjpalmer**

 **@LukasStefanko**

 **@aprilwright**

 **@anton_chuvakin**

 **@window**

 **@e_kaspersky**

podcasts

Southern-Fried Security

All of your information security needs are addressed in this this podcast, which addresses topics ranging from phishing to building a security strategy to evaluating security product vendors.

Risky Business

This monthly podcast features interviews with security aficionados, along with recent security-related news. It's self-described as a "security podcast without the waffle."

Brakeing Down Security

In this weekly podcast, get the latest in the world of security, privacy, compliance, and regulatory issues in the workplace.

zones

Security dzone.com/security

The Security Zone covers topics related to securing applications and the machines that run them. The goal of the Zone is to help prepare the people who make and support those applications stay up to date on industry best practices, remain aware of new and omnipresent threats, and help them to think "security-first."

Performance dzone.com/performance

Scalability and optimization are constant concerns for the developer and operations manager. The Performance Zone focuses on all things performance, covering everything from database optimization to garbage collection, tool and technique comparisons, and tweaks to keep your code as efficient as possible.

DevOps dzone.com/devops

DevOps is a cultural movement, supported by exciting new tools, that is aimed at encouraging close cooperation within cross-disciplinary teams of dvelopers and IT operations. The DevOps Zone is your hot spot for news and resources about Continuous Delivery, Puppet, Chef, Jenkins, and much more.

refcardz

Introduction to DevSecOps

This Refcard will show you how to get started with DevSecOps with key themes, crucial steps to begin your journey, and a guide to choosing security tools and technologies to build your DevSecOps pipeline.

Docker Security

This Refcard will lay out the basics of the container security challenge, give you hands-on experience with basic security options, and also spell out some more advanced workflows. We split container security into three sections covering what to do at each step of your container security lifecycle.

Java EE Security Essentials

This newly updated Refcard begins by introducing some common terms and concepts related to Java EE security such as identity stores and authentication mechanisms. We then explore authentication authorization, web module security, EJB module security, and application client security with in-depth examples.

books

Cybersecurity & CyberWar: What Everyone Needs to Know

This easy-to-read, deeply informative resource book explores how cyberspace security works, why it matters, and what we can do.

Threat Modeling: Designing for Security

Get the tools you need for structured thinking about what can go wrong when it comes to security, and learn how to adopt a structured approach to threat modeling.

Iron-Clad Java: Building Secure Web Applications

Learn about developing secure Java applications, eliminating existing security bugs, and defending against cross-site scripting.

Application Security Blanket

Developers and architects are expected more and more to understand and implement secure code and best practices, but only 25% of developers get any sort of training on the subject — and 33% get "ad-hoc training," which is likely a nice way of saying "searching online for the answer." So, let's start with a few basic security design patterns. These should all look familiar to you, because you've probably used applications in your work and personal lives that follow several of them. So, grab a nice cup of coffee and let's begin.

Roles

Utilizing roles is a fairly simple concept: system access is only granted to people in specific roles at the organization. For example, if you want someone to make changes to the frontend of an application, you may only grant access to developers and architects, and automatically include other roles such as marketing, sales, or support staff.

Sessions

Anyone who logs into a system has begun a "session" of time they're spending with that system, and to make sessions more secure, they are monitored so that any suspicious activity that happens can be traced back to the user or attacker who accessed the system.

Secure Access Layers

Secure access layers, or "layered security," is the practice of combining multiple security protocols and tools to protect a system. For each common attack or vulnerability, there should be at least one protection against it.

Single Access Points

Secure access layers, or "layered security," is the practice of combining multiple security protocols and tools to protect a system. For each common attack or vulnerability, there should be at least one protection against it.

Limited Views

In addition to limiting access to specific roles, developers can introduce limited views to each role so that those roles can only see options that concern them. For example, an admin logged into a web page may see options to edit that page, but a standard user would not.

Checkpoints

Developers can utilize checkpoints to organize security checks and their consequences. A checkpoint is more like a strategy or plan that can be changed to prioritize security threats or alerts. For example, forgetting your password is common, so someone typing in the wrong password a few times should not trigger a major warning, but checkpoints can lock someone out of their account after trying more than a reasonable number of times.

Security and compliance don't need to slow you down

Understand the state of your applications and detect and correct issues for a confident migration to the cloud with InSpec from Chef.

InSpec is Chef's open-source language for describing security & compliance rules that can be shared between software engineers, operations, and security engineers. Your compliance, security, and other policy requirements become automated tests that can be run against traditional servers, containers, and cloud APIs alike, ensuring consistent standards are enforced in every environment you manage, at every stage of development.

How InSpec makes security and compliance easier:

- Easily Assess Security and Compliance.
- Detect Issues and Prioritize Remediation.
- Inspect Machines, Data, and new SaaS APIs.
- Keep up with Rapidly Changing Threat and Compliance Landscapes.

Ready to see how compliance as code helps you manage security across your entire infrastructure? Learn more at http://chef.io/inspec

> "InSpec has helped us unify our compliance, security and DevOps teams and streamlined audits, reducing the thousands of staff hours usually required by as much as 95 percent."
>
> **Jon Williams, CTO, Niu Solutions**

INSPEC

CHEF™

Automate the validation of compliance and security in any environment, on any platform with Chef InSpec.

InSpec turns compliance, security, and other policy requirements into automated tests so you can stop treating information security as an afterthought and build security directly into the software delivery lifecycle.

An open-source language, InSpec lets you describe security and compliance rules that can be shared between software engineers, operations, and security engineers. Unlike other products, InSpec is designed to be used at all stages of the software delivery process, from a developer's workstation all the way to production, with no performance impact or side-effects, allowing companies to achieve continuous compliance. And, in contrast to other compliance languages, InSpec is designed to be easy to use, even by users with no background in programming.

The traditional approach for ensuring enterprise IT security and compliance is to bolt it on near the end of the product delivery process, right before work products go to production. This not only slows down the speed of delivery, but it is clearly not working, as evidenced by the continued drumbeat of major security breaches attributed to previously-disclosed vulnerabilities. Delivery of non-compliant environments is a direct result of onerous, manual compliance processes involving the exchange of written documents, spreadsheets or PDFs, causing compliance to frequently be ignored in the rush to production. The answer isn't more control: the answer is more collaboration.

InSpec lets you define compliance as code, using industry standard profiles such as Center for Internet Security (CIS), widely available profiles for standards such as HIPAA, GDPR, and custom security and compliance profiles you create for your own specific needs. And InSpec works across all of your systems, from on-premises and VM based apps to all of the major Cloud platforms. Learn more and try it yourself today at chef.io/inspec.

WRITTEN BY JULIAN DUNN
DIRECTOR OF PRODUCT MARKETING, CHEF SOFTWARE

PARTNER SPOTLIGHT

Chef InSpec

Automate the validation of compliance and security in any environment, on any platform with Chef InSpec.

CATEGORY
Security and Compliance Automation

RELEASE SCHEDULE
Monthly releases

OPEN SOURCE?
Yes

CASE STUDY

Niu Solutions is a UK-based consultancy whose challenge is to help these high-velocity organizations deliver value fast, while enforcing the appropriate level of regulatory checks to ensure compliance of the final solution. Niu used Chef InSpec to allow them to continuously and automatically monitor projects for compliance based on specific rules, freeing up resources so teams can shift their focus to create and deliver core value rather than dedicating cycles to audit processes. This reduces the time Niu spends on compliance checks by 93%.

According to Jon Williams of Niu, "With InSpec, you have a real-time view of how you're performing. When you come to that audit exam you already know if you're passing or not. In fact, the event of the audit is a simple step of printing the output." Handing off compliance work to InSpec also created a blueprint for Niu to adopt Continuous Automation for their infrastructure configuration, deployment and management. Combining InSpec and Chef Automate allows Jon's team to detect and correct in the same process, and then automate the ongoing configuration and deployment based on lessons learned.

STRENGTHS

1. Easily assess security and compliance
2. Detect issues and prioritize remediation
3. Inspect machines, data, and new SaaS APIs
4. Satisfy audits any time and make them painless
5. Reduce ambiguity and miscommunication around rules
6. Keep up with rapidly changing threats and compliance landscapes

NOTABLE USERS

- Alaska Airlines
- GE
- Carfax
- CapitalOne
- Optum

WEBSITE chef.io **TWITTER** @chef **BLOG** blog.chef.io

"Shift Up"

New Security Considerations for Containers-as-a-Service and Serverless Architectures

QUICK VIEW

01. The concept of "shift left" engages security earlier in the development cycle for cloud-native applications, accelerating application cycles while reducing risk.

02. Containers and serverless environments bring unique security challenges since the network, host and enforcement points are no longer within your full control.

03. Forward-thinking security organizations must also "shift up" to focus on protecting the application tier with controls at the development, container, and orchestration levels.

BY TSVI KORREN
CISSP CHIEF SOLUTIONS ARCHITECT, **AQUA SECURITY**

While containers can create more secure application development environments and dramatically speed up development, they can also introduce new security challenges. Containers are applications vulnerable to privilege escalation, zero-day attacks, hostile takeovers, and data exfiltration, and the later you discover a problem, the more difficult it is to mitigate the damage. This compels DevOps and information security teams to facilitate security's "shift left" to the beginning of the development cycle. However, shifting left does not address the fact that deploying containers and migrating to cloud-native environments also necessitates the security team to "shift up" to focus on its new priority: protecting the application tier.

The world is quickly moving away from the on-premises environment to cloud-based and serverless environments where control of the middle — i.e. the network, host, and enforcement points — is no longer in the hands of the organization that's producing the workload. This creates a gap between what security organizations are able to do with the tools they have, and their ability (or rather, inability) to execute those controls in the computing environment.

SHIFT LEFT, THEN SHIFT UP

The security team works with DevOps to shift left in order to identify and remedy potential security issues with applications before they move into production and throughout the development cycle. Yet, while shifting left is absolutely necessary, that alone is not sufficient to address the new security issues containers can create.

For example, shifting left does not provide for effective incident response because that has everything to do with what controls the security team has over the runtime environment. Effective incident response requires identifying the incident, understanding its causes and potential effects, then deciding on whether or not to take action.

However, shifting left does not address the fact that deploying containers and migrating to cloud-native environments also necessitates the security team to "shift up" to focus on its new priority: protecting the application tier.

Consider just one aspect of how security is managed in the runtime environment. In a traditional server infrastructure on-premises or in the cloud, where you have an application running on an operating system, the standard approach is to install anti-malware on the operating system of a VM so that if the application

is compromised and the malware tries to impact the OS, the anti-malware stops it.

If your organization is using AWS Fargate or Microsoft ACI, where do you install anti-malware? You don't, because the traditional location for executing security policies in the middle layers are no longer in your purview. More security organizations are realizing those controls are important to address even after they have worked with DevOps to facilitate the shift left. This represents a sea change from the days when security had to concern itself primarily with securing networks, hosts, and endpoints.

Effective incident response requires identifying the incident, understanding its causes and potential effects, then deciding on whether or not to take action.

HOW TO SHIFT UP?

The first step is to come to the realization that you can't execute controls and policies in the traditional way. There has to be more thought put into what controls need to be executed and where. Some tasks will shift left, like understanding what potential vulnerabilities or deficiencies exist in somebody's code and how the configuration of the image could create a potential takeover situation. Others remain in the runtime, such as monitoring what containers are doing and understanding what software is running in them.

Another major change is the loss of the enforcement point on the underlying operating system. Of course, it has to go somewhere — ideally inside the container where you will execute the controls, manage incident response controls, etc. A different solution is required because you're operating in a different environment.

Consider all controls that were once executed in the operating system: preventing rogue deployments and malicious code injections, securing user credentials, guarding network connections, and thwarting zero-day attacks. Shifting up requires you to spread these controls among the container, orchestration, and development environments. You can't execute that with your existing security tools because all of those tools were based on being embedded into the underlying operating system, which is no longer available.

STRENGTHENING YOUR SECURITY POSTURE

One primary benefit of shifting left is that you reduce the surface area for an attack. Containers and cloud-native applications enable the use of thin, hardened hosts that only require those resources needed for the container engine. Their smaller perimeters make identifying anomalies and suspicious activities easier because you're analyzing data from a much smaller pool of permitted actions.

Shifting up further minimizes the attack surface by allowing only what the application needs, and preventing users from adding any other processes, benign or not. This requires implementing a solution that enables you to maintain the container's single purpose and user, thereby ensuring that even a compromised container is useless to attackers.

Shifting up further minimizes the attack surface by allowing only what the application needs, and preventing users from adding any other processes, benign or not.

Another key similarity between shifting left and shifting up: don't leave security to the very end. The first step to solving any problem is realizing one exists. Disappearing enforcement points, changing areas of responsibilities, and the increased speed of application development are all forces driving the need for the security organization to shift up. Security can't ignore the upper layers of the deployment stack. Once confronted with how much control DevOps has over the entire stack, it becomes clear that security needs to shift up to focus on protecting the application layer.

TSVI KORREN CISSP has been an IT security professional for more than 20 years. In previous positions at DEC and CA Inc., he consulted with customers across various industries, helping them define and optimize the processes and organizational aspects of their information security. Tsvi is currently the Chief Solutions Architect at Aqua, where he leads the effort of enabling organizations to use containers and improve their security through DevSecOps.

Apply These Best Practices to Secure Android and iOS Mobile Apps

BY KATIE STRZEMPKA
VP OF CUSTOMER SUCCESS AND SERVICES, **NOWSECURE**

QUICK VIEW

01. Insecure data storage is a major factor in mobile device security breaches.

02. Best practices include properly protecting user credentials, third-party encryption, using keychain carefully, and choosing the most restrictive data protection.

03. Another major attack vector is the insecure transmission of data.

04. Best practices to protect against insecure communication include certificate pinning, certificate validation, and hostname verification.

In the race to develop and evolve mobile apps with rich user experience, security can sometimes be left in the dust. A staggering 85% of some 45,000 mobile apps available in the Google Play™ and the Apple® App Store® violate at least one or more of the OWASP Mobile Top 10, according to a NowSecure benchmark analysis. That figure attests to the importance of incorporating mobile appsec best practices and automated testing into the development process.

Mobile apps provide several potential attack vectors for hackers to exploit. The most prevalent and common vulnerabilities arise from failing to secure data at rest (DAR) or data in motion (DIM) over the air. This can lead to stolen credentials, leaked company secrets, privacy violations, and more. What follows are some recommendations for how developers can create more secure Android and iOS mobile apps by safeguarding stored data and properly validating SSL/TLS certificates.

INSECURE DATA STORAGE

In our experience, user and application data is frequently stored insecurely. The best way to avoid data compromise is to refrain from storing or caching data on the mobile device. However, that practice isn't always feasible. One option for minimizing the storage of user or app information is to transmit and display data, but not to persist to memory. This requires developers to ensure that a leak doesn't occur when screenshots are written to disk. Another technique for reducing risk is to only store data in RAM temporarily.

Consider adopting some of the following additional best practices for storing data.

PROPERLY PROTECT USER CREDENTIALS

Never, ever store a password in plain text on a device. If it must be stored, create a hash value using a unique device token stored on registration. The reason to use a hash and device token is to avoid storing the actual username or password locally or loaded into memory unprotected. Even if it was copied to another device or posted on the web, it wouldn't be usable. A malicious threat actor would need to uncover more information in order to successfully access the authenticated user name.

ADD THIRD-PARTY ENCRYPTION

If storing sensitive data on the device is an application requirement, add an additional layer of verified, third-party encryption such as SQLCipher to the data. Device encryption alone is not sufficient. Adding another layer of encryption provides more control over the implementation and attacks focused on the main OS encryption classes.

Encrypt highly sensitive values in a SQLite database using SQLCipher, which encrypts the entire database using a PRAGMA key. Generate the PRAGMA key at runtime when the user installs/launches the application for the first time. Generate a unique PRAGMA key for each user or device. The source of the key material must have sufficient entropy. For instance, don't choose something that's easily predictable such as a username or something else that can be guessed.

WATCH EXTERNAL STORAGE

Don't ever store user or app data in an Android external storage (SD) card. Prior to API 19, any app can read external storage. Post-API 19 (KitKat), apps require READ_EXTERNAL_STORAGE permission.

Apps with this permission can access any data on external storage without further permission management. If you need to store app data in this manner, protect it by setting proper file permissions to avoid unauthorized read/write access.

If an app requests data from external storage, developers must use scoped directory access via the `StorageManager` class. This enables users to only grant access to a specific directory from external storage that's specified by the developer. For example, an app may only need photos taken with the native camera app. A user could grant the app with access to the pictures directory, but not the entire SD card.

```
StorageVolume volume = (StorageVolume)
    mediaMountedIntent.getParcelableExtra(StorageVolume
EXTRA_STORAGE_VOLUME);
volume.createAccessIntent(Environment
DIRECTORY_PICTURES);
startActivityForResult(intent, request_code);
```

USE KEYCHAIN CAREFULLY

iOS provides the Keychain for secure data storage. But in several scenarios, such as when a device has been jailbroken, the keychain can be compromised and subsequently decrypted. And in iOS 10.3 API and below, items added to the Keychain will remain in the Keychain even after the app has been uninstalled.

CHOOSE THE MOST RESTRICTIVE PROTECTION

When storing data in the Keychain, use the most restrictive protection class as defined by the `kSecAttrAccessible` attribute that still allows your application to function properly. For example, if your application is not designed to be running in the background, use `kSecAttrAccessibleWhenUnlocked` or `kSecAttrAccessibleWhenUnlockedThisDeviceOnly`.

- To prevent the exposure of Keychain data via iTunes backup, use the "...`ThisDeviceOnly`" protection classes whenever possible.
- For highly sensitive data, supplement Keychain protections with application-level encryption. For example, require the user to enter a passphrase to authenticate within the application, then use that passphrase to encrypt data before storing it in the Keychain.

INSECURE DATA TRANSMISSION

The second major attack vector comes from data in motion that hasn't been fully secured. In particular, potential problems arise from the lack of TLS/SSL certificate validation. Many apps don't properly validate SSL/TLS certificates, leaving them vulnerable to man-in-the-middle attacks.

Two common mistakes around certificate validation can lead to man-in-the-middle (MiTM) attacks. The first stems from accepting self-signed certificates. Developers may disable certificate validation in apps for a variety of reasons. For example, a developer may need to test code on the production server, but lacks a domain certificate for the test environment. In this situation, the developer may add code to the networking library to accept all certificates as valid.

Accepting all certificates as valid, however, allows an attacker to execute an MiTM attack on the app by simply using a self-signed certificate. This approach to developing an app nullifies the effect of SSL/TLS and provides no value over an unencrypted, plaintext connection (other than requiring an active MITM attack to view and modify traffic whereas a plaintext connection can be monitored passively).

Another common mistake in implementing SSL/TLS is setting a permissive hostname verifier. In this case, the app won't accept self-signed certificates because the certificate is still validated. But if an app "allows all hostnames," a certificate issued by any valid certificate authority (CA) for any domain name can be used to execute an MITM attack and sign traffic.

IMPLEMENT CERTIFICATE PINNING

For any app that handles data considered to be highly sensitive, implement certificate pinning to reduce vulnerability to MitM attacks. If a user can be tricked into a malicious self-signed certificate on a mobile device, certificate pinning can prevent network traffic interception. With certificate pinning, a client is pre-configured to know what server certificate it should expect to connect to. If they don't match, the client will prevent that session from occurring.

CONDUCT PROPER CERTIFICATE VALIDATION

For apps that handle data considered to be highly sensitive but for which certificates can't be pinned, developers need to employ certificate validation. Certificates presented to the app must be fully validated by the app and signed by a trusted root certificate authority.

EMPLOY HOSTNAME VERIFICATION

The app must check and verify that the hostname (Common Name or CN) extracted from the certificate matches that of the host the app intends to communicate with.

In addition to following best practices for developing secure mobile apps, all organizations can benefit from integrating security testing into the dev lifecycle to find and fix flaws faster. Learn more about how to embark on the journey to secure DevOps for mobile apps in a new ebook.

KATIE STRZEMPKA is the VP of Customer Success and Services at NowSecure, oversees the NowSecure mobile app security testing methodology, manages a team of expert mobile app penetration testers, co-authored the book iPhone and iOS Forensics, and is a recognized expert in mobile security. Katie holds a Master's in Cyber Forensics and a Bachelor of Science in Computer Technology from Purdue University.

You Know You Want to Secure Your DevOps Environment

Whether in dev, DevOps, IT or security, we all know that our apps and data need to be protected. We've all heard the horror stories of access keys or other credentials inadvertently left in code available on GitHub – ready for truffleHog to scoop up. OK, so it's a problem, but we've got apps and new features to deploy.

SECURITY THAT YOU CAN KISS

If we're going to secure our DevOps environment, it better not be complicated or impact the team's flow or velocity. Basically, "keep it simple, stupid." If the existing code uses environmental variables, why not automatically remove the hardcoded credentials and automatically onboard them into the secrets management system?

SECURITY ISLANDS AREN'T SECURE

Almost all CI/CD tools and container platforms offer a secrets management solution: Puppet has Hiera; Ansible, Vault; Kubernetes, Secrets; AWS, Secrets Manager; etc. So why not use them? First, most CI/CD pipelines have multiple tools, and you'll need to securely share secrets and credentials among them. Typically, they're not great at securely sharing secrets, so at best you have islands of security.

Is managing secrets in multiple tools keeping it simple? No!

USE A VENDOR FOCUSED ON SECURITY

Second, security is hard, and attackers only need to be right once. Security isn't the primary focus for the CI/CD tool vendors, or even the cloud vendors, so secrets management is an add-on. Some have basic credential rotation, but not robust audit, segregation of duties or role-based access control. The best practice is to simplify security and use a single platform to manage all your secrets. Ideally the platform has native integrations with CI/CD tools to seamlessly and securely manage credentials, and it is from a vendor focused on security.

MAKE IT EASY FOR DEVS AND LET SECURITY WORRY ABOUT SECRETS MANAGEMENT

To protect your DevOps environment, your secrets management platform should be fully functioned, easy to use and from a vendor 100% focused on security. Start today with CyberArk Conjur Open-Source or Enterprise.

WRITTEN BY CHRIS SMITH
DIRECTOR, DEVOPS PRODUCT MARKETING, **CYBERARK SOFTWARE**

PARTNER SPOTLIGHT

CyberArk Conjur

CYBERARK® CONJUR

The most comprehensive solution for securing secrets and credentials across DevOps, containers, and cloud native applications.

CATEGORY	RELEASE SCHEDULE	OPEN SOURCE?
Software Security	4-6 per year	Yes

USE CASES

CyberArk Conjur is a powerful secrets management solution, designed for the unique requirements of cloud-native applications, container orchestration and DevOps. Conjur seamlessly integrates with your PaaS platform and DevOps tool chain to easily and consistently protect secrets, keys, certificates and authentication data.

Example use cases include:

1. Securing secrets – Remove, secure, and manage (rotation, audit, etc.) secrets and other credentials used by applications, containers, and other non-human users.
2. Securing containers – Leverage the container platform's native capabilities to authenticate requests for secrets, and then manage these secrets.
3. Securing the CI/CD pipeline – Integrate with other CyberArk solutions to secure the DevOps tool admin consoles (Jenkins, Ansible, etc.), and the scripts used to run the pipeline.

STRENGTHS

1. **Developer friendly:** Remove developer friction with automated onboarding of secrets from code.
2. **CI/CD and cloud ready:** Integrate seamlessly with leading CI/CD tools, containers and cloud platforms including, Jenkins, Ansible, Kubernetes, Pivotal, AWS and GCP.
3. **Centralized management:** Apply consistent security access policies and audit across the enterprise.
4. **Enterprise class:** Achieve high scalability and high performance. Leverage global support.

ENTERPRISE CUSTOMERS

CyberArk solutions are used by organizations across the globe, including more than half of the Fortune 100.

WEBSITE Conjur.org | **TWITTER** @CyberArk | slack | **BLOG** Conjur.org/blog

Introduction to Virtual Patching

QUICK VIEW

01. Learn why patching is still a problem for enterprise software.

02. Discover the virtues of virtual patching.

03. Learn about the evolution of virtual patching and emergence of runtime virtual patching.

BY APOSTOLOS GIANNAKIDIS
SECURITY ARCHITECT, **WARATEK**

Equifax estimates the costs associated with a 2017 security breach will exceed $600 million. And it is not just Equifax that has suffered a tremendous financial hit. Dozens of other organizations have lost more than $100 million in revenue over the past year due to outdated software.

According to Gartner, 99% of vulnerabilities exploited are not zero-days. Instead, they are vulnerabilities that were known for at least one year. In fact, most IT infrastructures include legacy platforms, unpatched software, outdated third-party components, and poor patch management processes. It is much easier for bad guys to find unpatched applications than to discover a zero-day vulnerability.

SO, WHY IS PATCHING STILL A PROBLEM?

Patching for end-user applications may be simple to trivial most of the time. However, it is an entirely different story for enterprise software. This is because enterprise software is inherently more complex. Installing a patch for enterprise software is, in most cases, an expensive and manual process that consists of several rigorous steps that cannot always be done during regular business hours and typically requires critical systems to be offline.

Downtime is a precious commodity for enterprises and to get around it there are strategies and techniques such as rolling updates, blue-green deployments, and canary releases. Such strategies are very complex, highly error-prone, and time-consuming, and require careful planning as well as platform and infrastructure changes. Additionally, new releases of software oftentimes come with new functionality that is not backwards-compatible and has the potential to disrupt the normal operation of the application.

PATCHING IS A LOSING BATTLE

According to Sonatype, more than 10,000 new versions of open-source components are released daily, offering new features, bug fixes, and security patches. What is more astonishing though is that, according to Mitre, almost 15,000 new vulnerabilities surfaced in 2017. That is an average of 41 new vulnerabilities disclosed per day or a new vulnerability every 30 minutes.

There simply is not enough time or resources to address these flaws in a timely fashion before attackers use their inherent advantages to find a way in. Therefore, patching is a significant challenge for IT departments. Organizations value meeting deadlines and not slowing productivity more than the risk of unpatched software. What commonly happens is teams accept the risk and patch at a more “convenient” time, usually once per quarter or longer. However, this practice creates a window of opportunity for the bad guys.

Any time a new patch is released to fix a vulnerability, it starts a race against the clock for malicious actors to find organizations with unpatched systems and exploit the flaws. It has become a common case to see malicious activity just hours after security alerts are publicly released. According to recent Ponemon study, it takes, on average, three to six days for an attacker to successfully exploit a vulnerability, more than 250 days to discover an attack is underway, and an additional 82 days to contain the attack. This raises a critical question: How can organizations protect their assets while deferring the actual patching of their systems?

VIRTUAL PATCHING TO THE RESCUE

Virtual patching refers to establishing an immediate security policy enforcement layer which prevents the exploitation of a known vulnerability, without modifying the application's source code, binary changes, or restarting the application. Using a virtual patching strategy, organizations can achieve a nice balance between substantially reducing the cost, time, and effort needed to patch while maintaining their availability SLAs and normal patching cycles.

Virtual patching can be used to add protection temporarily, giving DevSecOps teams time to deploy physical patches according to their

own update schedules. Virtual patches can also be used permanently for legacy systems that may not patchable.

Additionally, there is less likelihood of introducing conflicts or incompatibilities as no application files are changed on disk.

Moreover, it is common for software vendors to release their security fixes in bundles that are installed in an all-or-nothing approach. When one of the fixes causes functional issues to the application, then the whole security bundle must be uninstalled.

Virtual patching allows organizations to cherry-pick which patches are important to their environments and lets them rollback specific patches that are not functionally compatible with the application.

TYPES OF VIRTUAL PATCHING

Virtual patching was initially pioneered by the IPS/IDS community quite a few years ago. Later, virtual patching was brought into the WAF space and, most recently, RASP products offer similar capabilities.

It is important to clarify, though, that not all virtual patching solutions are equal. In general, we could classify virtual patching according the quality of the patching that can be delivered.

The first type of virtual patching is purely based on analyzing network traffic. The system that offers this type of virtual patching uses signatures, regular expressions, and pattern matching to identify malicious activity and block the corresponding requests.

The second type of virtual patching is based on the same principles; however, it offers a more robust way of specifying the criteria to block requests, using a rules language and capabilities such as state management. ModSecurity is an example of such a virtual patching solution.

Virtual patching of these types has several disadvantages. Because they lack context-awareness, these virtual patching techniques are notorious for generating false positives and false negatives. Signatures and pattern-matching are heuristic approaches that cannot offer the accuracy that is required to be highly effective. A poorly written virtual patch of this type risks blocking legitimate traffic, disrupting the normal execution of the application, and adding extra workload to security teams who must filter out the false alarms.

Additionally, patches of this type are very fragile to change. In the case of web applications, any time there is a change to the vulnerable URL or HTTP parameters, then the virtual patch is invalidated and needs to be re-configured.

More importantly, these approaches safeguard specific inputs but leave the vulnerable components unprotected. If there are more entry points that lead to the same vulnerable component, then the virtual patch will fail to protect.

One could claim that it is an overstatement to say that the above approaches provide "patching" capabilities, since the code remains vulnerable even after the virtual patch is applied. This is why some refer to these approaches as "vulnerability shielding" instead of patching.

The third type is the natural evolution of virtual patching and leverages the Just-in-Time compilers of contemporary runtime platforms (such as the Java Virtual Machine and the Common Language Runtime). Being inside the application and being able to control every instruction before it gets executed allows true virtual patching to be achieved. It is now feasible to deliver a virtual patch that is functionally equivalent to the vendor's physical patch without modifying the application's source code or its binary files while ensuring the component is properly patched and is no longer vulnerable.

Runtime virtual patching is the type of virtual patching that the community has aspired to achieve since its inception. WAF/IDS engineers envisioned the attraction of virtual patching, but the lack of control of the runtime platform and the architectural limitations that place such solutions outside of the application have forced virtual patching to be more or less an elaborate input/output validation solution.

This does not mean that these types of virtual patching have no use. On the contrary, there are cases where applying an input validation virtual patch makes sense. Consider the cases of legacy and end-of-life systems or when the vendor does not release a security fix. Recently, Sonatype found that 84% of open-source projects do not fix known security defects. For such cases, a compensating control based on input validation type virtual patching may be the only way to protect such systems.

Virtual patching has advanced to the point it can largely be an automated process that requires little manual intervention and is just as effective and perhaps more robust than physical patches. With compliance regimens requiring more rigid patch management processes, the time has come for security teams to adopt a virtual patching strategy into their vulnerability and patch management programs.

Virtual patching comes in various ways each having its own advantages. When teams asses their patching needs, they must be sure to have a clear goal in mind about why they are patching and what they are trying to achieve. Does a shield against known signatures that does not fix the underlying software flaw suffice, or is a virtual patch that is functionally equivalent to a physical patch and remediates the flaw in the software component required?

APOSTOLOS GIANNAKIDIS is the Security Architect at Waratek driving the R&D of runtime security features. Before joining Waratek, Apostolos worked at Oracle focusing on destructive and security testing. Apostolos has more than 15 years of experience in the software industry and holds an MSc in Computer Science. Apostolos has received an honorable mention by the Ministry of Finance of Greece, has been acknowledged by Oracle for submitting Java vulnerabilities and is featured on Google's Vulnerability Hall of Fame.

Executive Insights on the Current and Future State of Security

QUICK VIEW

01. The most important elements of application and data security are: 1) access; 2) encryption; 3) depth; 4) education; 5) data; and, 6) security by design.

02. The most significant changes to the cybersecurity landscape are: 1) expanded threat vectors; 2) the speed of change; and, 3) legislation taking place.

03. The most common issues causing poor security are: 1) lack of compliance; 2) human error; and, 3) lack of skills and knowledge.

BY TOM SMITH
RESEARCH ANALYST AT **DZONE**

To gather insights on the current and future state of security, we talked to 48 executives from 42 companies about security in their own organizations and for the clients with whom they are working. Given all of the breaches that have appeared in the news and the enforcement of GDPR, response to this topic was unlike any we have seen for previous security research guides. Here's who we talked to:

- Jim Souders, CEO, Adaptiva
- Murali Palanisamy, CTO, AppViewX
- Amir Jerbi, Co-Founder and CTO of Aqua Security
- Andreas Pettersson, CEO, Arcules
- Andrew Avanessian, COO, Avecto
- Dave Mariani, CEO and Co-Founder AtScale
- Bruno Aziza, CMO, AtScale
- Nitsan Miron, Vice President Product Management, Barracuda Networks
- Mo Rosen, GM, CA Security
- Sam King, GM, CA Veracode
- Mark Crumphey, CA SourceClear
- Stuart Scott, AWS Trainer /Cybersecurity Expert, Cloud Academy
- Cliff Turner, Senior Solutions Architect, CloudPassage
- Mark Forrest, CEO, Cryptshare
- Antonio Challita, Director of Product Management, CyberSight
- Doug Dooley, COO, Data Theorem
- Patrick Lightbody, SVP Product Management, Delphix
- OJ Ngo, CTO, DH2i
- Reid Tatoris, Vice President Product and Outreach Marketing, Distil Networks
- Paul Kraus, CEO, Eastwind Networks
- Don Lewis, Senior Marketing Manager, EdgeWave
- Anders Wallgren, CTO, Electric Cloud
- Venkat Ramasamy, COO, FileCloud
- Jesse Endahl, CPO, CSO and Co-Founder, Fleetsmith
- Tom Sela, Head of Security Research, illusive
- Matan Kubovsky, Vice President R&D, Illusive
- Roy Halevi, CTO and Co-founder, Intezer
- Darren Guccione, CEO, Keeper Security
- Andrew Howard, Chief Technology Officer, Kudelski Security
- Rajesh Ganesan, VP Product Development, ManageEngine
- John Omernik, Distinguished Technologist, MapR
- James Willet, Vice President of Engineering, Neustar
- Gary Duan, CTO, NeuVector

- Randall Degges, Head of Developer Advocacy, Okta
- Dan Koloski, Vice President, Security and Systems Management, Oracle
- Heather Howland, CEO, Preempt
- Randy Battat, CEO, PreVeil
- Arkadiy Miteiko, CEO, QbitLogic
- Linus Chang, Founder , Scram Software
- Altaz Valani, Research Director, Security Compass
- Ed Adams, CEO, Security Innovation
- Neill Feather, CEO, SiteLock
- Oded Moshe, VP Products, SysAid
- Gaurav Deshpande, Vice President of Marketing, TigerGraph
- Todd Blaschka, COO, TigerGraph
- Matthew Vernhout, Director of Privacy and Industry Relations, 250ok
- Setu Kulkarni, Vice President of Product and Corporate Strategy, Whitehat Security
- Erik Nordmark, Co-founder and Chief Architect, Zededa

---

1. Several elements were mentioned as being important to application and data security: **1) access; 2) encryption; 3) depth; 4) education; 5) data; and, 6) security by design**.

Know who has access to critical mainframe resources and how. Have visibility into who has privileged access in your organization. Have flow-based access control and federated access. Have the ability to set-up fine-grained access control. As well as the ability to give people access to data without exposing PII.

Encryption should be standard for all data, as well as for every CISO and CIO. Have an encryption key and certificate, as well as assignment and key management.

Security is about having multiple layers and asking, "What if this mechanism is breached?" Have multiple solutions and layers to block threats.

It comes down to people. Developers, engineers, and staff need to understand the importance of security. Once organizational commitment is in place, you need to train your teams and provide teaching through a center of excellence so continuous learning happens, given how fast the landscape is changing.

Figure out where all of your data is and assess its importance and risk. Classify data based on sensitivity level for compliance purposes. Follow a security by design mindset from day one. Integrate security into the SDLC and follow a DevSecOps methodologies.

2. The three key ways the cybersecurity landscape is changing are: **1) expanded threat vectors; 2) the speed of change; and, 3) legislation taking place.**

The landscape, terrain, and targets are growing exponentially. There are more threat vectors including third-party applications, insecure IoT devices, mobile, and cloud technologies. 95% of code is being built with open source software, and reusable code equals reusable vulnerabilities.

The landscape is changing on a daily basis with more devices and device types connected to the internet. One of the greatest challenges impacting organizations today is simply velocity. More things are connected to the network with poor security and plenty of bandwidth.

Legislation has amped up with GDPR and Australia's data breach disclosure requirements. There is a lot of legislative impact on security protocols and processes. This is the first time law enforcement is enforcing protection and data security around the world. Fines and mandatory reporting are causing organizations to take cybersecurity more seriously.

3. The most effective security techniques involve **multi-level and company-wide engagement and ownership.** Security involves prevention, detection, remediation, and response. Compliance and best practices are very important, as are modern application security principles, a DevSecOps methodologies, and security analytics tools. Organizations are the most effective when people, processes, and technology are all integrated into a continuous cycle that can adapt readily to change.

Employees need to become part of a security culture. Educate your staff on the need to follow the principle of least privilege, granting access to only those who absolutely need it. Security teams need to act more as coaches and consultants to the teams deploying code.

4. There is a tremendous breadth of use cases spanning 12 industries and 23 applications across the professionals with whom we spoke. As expected, **financial services and healthcare were the most frequently mentioned industries** while the **most frequently mentioned applications were fraud detection and cloud security**. Following are a few specific examples:

Alipay processes 100,000 payments every second. China telecom has 450 million clients making billions of calls. All payments

and calls are added to a graph database in real time, fed into a machine learning tool, and encrypted as they are sent out for fraud detection.

A security company performing background checks needed to collect 100 points of identification including tax returns, passports, and other information which was stored and encrypted as long as the data was needed for verification.

VMware is building microservices and is using a solution that could scale, provide visibility into the threat surface, have an inventory of the code, and check the security of the open source code on the frontend without slowing down the development process.

5. The most common issues negatively affecting security are: **1) lack of compliance; 2) human error; and, 3) lack of skills and knowledge.**

While many organizations have a well-defined security policy, enforcement and compliance are a challenge. Weak credentials or credentials reuse have been responsible for breaches of websites like LinkedIn. It is rare for an organization to adhere to best security practices.

Human error and poor habits are the most common issues affecting security in the digitally connected world. People need to be trained and notified when they are doing something that jeopardizes the security of their network or device.

Awareness and security training is lacking. There is a shortage of qualified security professionals, and there is disagreement on whether developers need to be educated and if they should be expected to be responsible for the security of their code or the applications they are building. While many organizations are ready to embrace DevSecOps, others are not.

6. Everyone interviewed had concerns regarding the current state of security. The most frequently mentioned concerns were **lack of knowledge and governance as well as the speed and efficacy of the attackers.**

There's a lack of knowledge, education, and understanding. There needs to be a cultural shift so everyone in the company realizes security is their responsibility. Organizations are focusing too heavily on technology solutions and not enough on improving the basics of training and recruiting the right mix of personnel. Microservices are new to virtually everyone, and we see many more vulnerabilities per line of code than traditional applications.

Very few organizations or developers follow a security by design methodology from day one. There is a logical need for a well-developed and engineered solution. We need better standards, methodologies, and governance policies as well as more security oversight.

Hackers are getting smarter. The magnitude, frequency, and efficacy of security hacks are increasing. We're seeing a lot more automated and sophisticated attacks that combine several pieces of malware from different pieces of the kill chain.

7. The future of security is **automation supported by artificial intelligence (AI) and machine learning (ML)**. Ability to automate removing much of the repeated tasks and using a platform to complete the task is critical to keep pace with all of the changes and different types of attacks. Likewise, organizations need the ability to automate the response to a threat hence the need to adopt intelligent automation powered by AI/ML.

Identify how we can adapt to changing scenarios using AI/ML and understand AI/ML so we can optimize and implement it well and understand how an adversary might use AI/ML against us. The future belongs to intelligent machines that will augment security functions while working alongside humans.

8. Regarding security, developers need to **think about security by design in mind along with a myriad of best practices**. Embrace security early in the process during design and have it fully automated and integrated into the SDLC. Treat security like the nucleus and most important feature of the product. Grab DevSecOps with both hands and automate. The more developers know about security and the more secure code they develop, the more valuable they become.

Make secure development principles part of the way you think and work. Know the three key security frameworks: 1) Microsoft's Secure Development Lifecycle (SDL); 2) Open Web Application Security Project (OWASP); and, 3) Industrial Internet Consortium (IIC). Avoid basic mistakes like injection and overruns. Seek security awareness training. Follow secure development best practices, Interact with security engineers. Understand security is a team sport and everyone needs to be thinking about it. However, ultimately it's the developer's application and they are responsible for its security.

TOM SMITH is a Research Analyst at DZone who excels at gathering insights from analytics—both quantitative and qualitative—to drive business results. His passion is sharing information of value to help people succeed. In his spare time, you can find him either eating at Chipotle or working out at the gym.

Solutions Directory

This directory contains anti-tamper software, authentication, cloud access security, DDoS protection, endpoint security, and penetration testing tools, as well as many other tools to assist your application security. It provides free trial data and product category information gathered from vendor websites and project pages. Solutions are selected for inclusion based on several impartial criteria, including solution maturity, technical innovativeness, relevance, and data availability.

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
A10 Networks	Thunder TPS	DDoS protection	30 days	a10networks.com/products/ddos-protection-thunder-tps
Acunetix	Acunetix Web Vulnerability Scanner	DAST, IAST	Demo available by request	acunetix.com/vulnerability-scanner
Ad Novum	Nevis Security Suite	WAF, authentication, identity management	N/A	adnovum.sg/en/sg/solutions/products/nevis
Akamai	Akamai	CDN, DDoS protection, WAF	Available by request	akamai.com
Akamai	Kona Site Defender	WAF, DDoS protection	Available by request	akamai.com/us/en/solutions/products/cloud-security/kona-site-defender
Alert Logic, Inc.	Alert Logic Web Security Manager	WAF	Available by request	alertlogic.com/solutions
AlienVault	Unified Security Management	Security monitoring	Demo available by request	alienvault.com
Amazon	AWS WAF	WAF	Free tier available	aws.amazon.com/waf
Amazon	CloudFront	CDN, DDoS protection	Free tier available	aws.amazon.com/cloudfront
Anomali	ThreatStream Platform	Security monitoring	Demo available by request	anomali.com/platform

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
Appmobi	Appmobi	Apache Cordova app encryption & authentication	N/A	appmobi.com
AppRiver	SecureSurf	WAF	Available by request	appriver.com/services/web-protection
Appthroity	Appthority	Mobile AST	Demo available by request	appthority.com
Armor	Armor Anywhere	Cloud security platform	Demo available by request	armor.com/armor-anywhere-security
Arxan	Arxan Application Protection	Intrusion prevention system, code analysis	N/A	arxan.com/products/product-overview
Attivo	ThreatDefend Platform	Threat detection & monitoring	Demo available by request	attivonetworks.com/product/deception-technology
Auth0	Auth0	SSO & identity management	Free tier available	auth0.com
Avecto	Defendpoint	Privilege management, applicaton control	Demo available by request	avecto.com/defendpoint
Barracuda Networks	Barracuda NextGen Firewall	WAF	Available by request	barracuda.com/products/ngfirewall
Bay Dynamics	Risk Fabric	Predictive security analytics	N/A	baydynamics.com/risk-fabric-enterprises
Bitdefender	Bitdefender	Endpoint security, app security scanning	N/A	bitdefender.com
BlackDuck	Black Duck Hub	Open source scanning	Demo available by request	blackducksoftware.com/products/hub
BMC	SecOps Response Service	Security management & analytics	Available by request	bmc.com/it-solutions/truesight-vulnerability-management
Bricata	Bricata	Threat detection & analytics	Demo available by request	bricata.com/product
Bromium	Bromium Secure Platform	Virtualization-based endpoint security	Demo available by request	bromium.com/platform
Browser Exploitation Framework Project	BeEF	Web browser penetration testing	Open source	beefproject.com
CA Technologies	Advanced Authentication	Authentication	Available by request	ca.com/us/products/ca-advanced-authentication

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
CA Technologies	Veracode Cloud Platform	SAST, DAST, mobile AST, penetration testing	Demo available by request	veracode.com/products/application-security-platform
Carbon Black	CB Protection	Enpoint security	Demo available by request	carbonblack.com/products/cb-protection
Cavirin	Cavirin Platform	Cloud security scanning	Available by request	cavirin.com/cavirin-platform
CDNetworks	DDoS Protection	CDN, WAF, DDoS protection	N/A	cdnetworks.com/us/en/products/ddos-protection
Century Link	Level 3 Content Delivery Network	CDN, DDoS protection	N/A	level3.com/en/products/content-delivery-network
Check Point Software Technologies Ltd	Check Point CloudGuard	Infrastructure, WAN, PaaS security	N/A	checkpoint.com/products/cloud-security
Checkmarx	Checkmarx Static Code Analysis	SAST	Demo available by request	checkmarx.com/technology/static-code-analysis-sca
Chef Software	Chef	Infrastructure automation, configuration management	Open source	chef.io/chef
CipherCloud	CipherCloud	Cloud access security broker	Available by request	ciphercloud.com
CIRT.net	Nikto2	Web server scanner	Open source	cirt.net/Nikto2
Cisco	CloudLock Security Fabric	Cloud access security broker	Demo available by request	cloudlock.com/platform
Cloudentity	Cloudentity	Developer self-service for identity & API security	Available by request	cloudentity.com
Citrix	NetScaler AppFirewall	WAF	90 days	citrix.com/products/netscaler-appfirewall
Cloudflare, Inc.	Cloudflare	CDN, DDoS protection, WAF	Free tier available	cloudflare.com
CloudPassage	Halo Cloud Secure	Cloud access security broker	Demo available by request	cloudpassage.com/why-halo
Code42 Software	Code42	Endpoint security	Available by request	code42.com
Cofense	Cofense PhishMe	Phishing simulation software	Demo available by request	cofense.com/product-services/simulator-2
Cofense	Cofense Intelligence	Threat detection & analytics	Demo available by request	cofense.com/product-services/phishing-intelligence

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
Core Security	Core Impact	Penetration testing	Demo available by request	coresecurity.com/core-impact
Core Security	Actionable Insight & Response Platform	Security monitoring	Demo available by request	coresecurity.com/actionable-insight-platform
Covata	Covata	RBAC, cloud security	Demo available by request	covata.com
CrowdStrike	Falcon	Endpoint security	15 days	crowdstrike.com/products
CyberArk Software	CyberArk	Authentication	30 days	cyberark.com
Cybereason	Deep Hunting Platform	Endpoint security	Demo available by request	cybereason.com/hunting-platform
Cylance	CylancePROTECT	AI-based endpoint security	Demo available by request	cylance.com/en_us/products/our-products/enterprise-products/protect
Darktrace	Enterprise Immune System	AI-based endpoint security	30 days	darktrace.com/technology/#enterprise-immune-system
Datiphy	Datiphy Enterprise Cloud Platform	Real-time threat detection	Demo available by request	datiphy.com
DenyAll	DenyAll WAF	WAF	Demo available by request	denyall.com/products/web-application-firewall
Digital Guardian	Digital Guardian Platform	Endpoint security, threat detection	Demo available by request	digitalguardian.com/products/threat-aware-data-protection-platform
DigitalStakeout	DigitalStakeout	Real-time threat intelligence & security analytics	Demo available by request	vdigitalstakeout.com
DXC Technology	DXC Security	Endpoint, application, & data security	N/A	dxc.technology/security
Edgescan	Edgescan	Continuous vulnerability management	Available by request	edgescan.com
Endgame	Engame Platform	Endpoint security, security automation	Demo available by request	endgame.com/platform
Ergon Informatik AG	Airlock Suite	WAF, authentication, identity management	Demo available by request	airlock.com/en/home
ESET	ESET Endpoint Protection	Endpoint security	N/A	eset.com/us/business/endpoint-protection
Evident.io	Evident Security Platform	Cloud security & compliance automation	Available by request	evident.io/what-is-esp

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
F5 Networks	Herculon	DDoS protection, SSL orchestration	N/A	f5.com/products/herculon/ddos-hybrid-defender
F5 Networks	F5 Big-IP ADC platform	WAF, DDoS protection	90 days	f5.com/products/big-ip
Fidelis Cybersecurity	Fidelis Platform	Threat detection & monitoring, endpoint security	Demo available by request	fidelissecurity.com
FireEye	Managed Defense	Threat detection & forensics	N/A	fireeye.com/solutions/managed-defense
FireEye	FireEye Helix	Endpoint & network security	N/A	fireeye.com/products/helix
Forcepoint	Forcepoint Next Generation Firewall	WAF	Demo available by request	forcepoint.com/product/network-security/forcepoint-ngfw
Fortinet	Next-Generation Firewall	WAF	Demo available by request	fortinet.com/products-services/products/firewall
Fortinet	FortiClient Next-Generation Endpoint Security	Endpoint security	Demo available by request	fortinet.com/products/endpoint-security/forticlient
Forum Systems	Forum Sentry	Authentication, API security gateway	Demo available by request	forumsys.com/product-solutions/forum-sentry-api-security-gateway
Gemalto	Authentification Management Platforms	Authentication	N/A	gemalto.com/enterprise-security/identity-access-management
Gigamon	GigaSECURE	Network security, load balancer	90 days	gigamon.com/solutions/gigasecure-security-delivery-platform
GrammaTech	CodeSonar	SAST for IoT	30 days	grammatech.com/products/codesonar
Graylog	Graylog	Log management	Open source	graylog.org
Hillstone Networks	Intelligent Next-Gen T-Series Firewall	WAF	Demo available by request	hillstonenet.com/our-products/intelligent-next-gen-firewalls-t-series
IBM	IBM AppScan	SAST, DAST, IAST	Available by request	ibm.com/security/application-security/appscan
IBM Bluemix	Bluebox	Mobile access security broker	Demo available by request	ibm.com/cloud-computing/bluemix/bluemix-private-cloud

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
iBoss	iBoss	WAF	Demo available by request	iboss.com/why-iboss
Imperva	Imperva Incapsula	WAF, DDoS protection	Demo available by request	incapsula.com
Infoblox	InfoBlox DNS Firewall	WAF	N/A	infoblox.com/products/dns-firewall
Intel	McAfee Small Business Security	Antivirus & monitoring	30 days	mcafee.com/consumer/en-us/store/m0
Ixia	Breakingpoint VE	Penetration testing	Demo available by request	ixiacom.com/products/breakingpoint-ve
Janrain	Social Login	Identity management & authentication	N/A	janrain.com/product
Juniper Networks	SRX Series Firewall	WAF	N/A	juniper.net/us/en/products-services/security/srx-series
Kali Linux	Kali Linux	Penetration testing	Open source	kali.org
Lastline	Lastline Breach Defender	Network security & monitoring	Demo available by request	lastline.com/products/defender
Lieberman Software	Lieberman RED Suite	Identity management & authentication	Demo available by request	liebsoft.com/red
LogRhythm	Threat Lifecycle Management Platform	Predictive security analytics	Demo available by request	logrhythm.com/products/threat-lifecycle-management-platform
Malwarebytes	Malwarebytes Endpoint Security	Endpoint security	Available by request	malwarebytes.com/business/endpointsecurity
MetaFlows	MetaFlows	Cloud security scanning	N/A	metaflows.com
Micro Focus	Fortify Static Code Analyzer	SAST, DAST, IAST, RASP	N/A	software.microfocus.com/en-us/products/static-code-analysis-sast/overview
Microsoft	Cloud App Security	Cloud access security broker	Available by request	microsoft.com/en-us/cloud-platform/cloud-app-security
N-Stalker	N-Stalker Web Application Security Scanner X	SAST, DAST	Free tier available	nstalker.com/products/editions
Netscout	Arbor	DDoS protection	N/A	netscout.com/product/arbor-availability-protection-system

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
Neustar, Inc.	Neustar	DDoS protection	N/A	home.neustar
NGINX	NGINX Plus	Authentication, DDoS protection	30 days	nginx.com/products
nmap.org	Nmap	Penetration testing & network mapping	Open source	nmap.org
Nokia	Deepfield	DDoS protection	N/A	deepfield.com/products/deepfield-defender
NowSecure	NowSecure	Penetration testing, compliance, security scanning services	Demo available by request	nowsecure.com
NSFOCUS	NSFOCUS ADS	DDoS protection	N/A	nsfocusglobal.com/products-overview
NSS Labs	CAWS Continuous Security Validation Platform	Security monitoring	N/A	nsslabs.com/caws-continuous-security-validation-platform/overview
Okta	Okta Platform	Authentication	Free tier available	okta.com/products/api-products
OPSWAT	MetaDefender	SAST, endpoint security	15 days	opswat.com/metadefender-core
OWASP	OWASP Projects	Various open-source security projects	Open source	owasp.org/index.php/Main_Page
Palo Alto Networks, Inc.	PA-7000 Series Firewall by Palo Alto Networks	WAF	Available by request	paloaltonetworks.com/products/secure-the-network/next-generation-firewall/pa-7000-series
Palo Alto Networks, Inc.	Palo Alto Enterprise Security Platform	RASP WAF	Available by request	paloaltonetworks.com/products/designing-for-prevention/security-platform
Peach Technology	Peach Fuzzer	Automated penetration testing	Available by request	peach.tech
Peach Technology	Peach API Security	Automated security scanning	Available by request	peach.tech/products/peach-api-security
Proofpoint	Cloudmark Insight Server	Real-time threat detection	Available by request	cloudmark.com/en/s/products/cloudmark-insight-server
PortSwigger LLC	Burp Suite	SAST, DAST, penetration testing	Free tier available	portswigger.net/burp

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
Pradeo	Pradeo	Mobile AST	Available by request	pradeo.com/en-US/application-security-testing
Prevoty	Prevoty RASP	RASP	Demo available by request	prevoty.com/products
ProtectWise	ProtectWise Grid	App security scanning	Demo available by request	protectwise.com/platform
Qualys, Inc.	Qualys Cloud Platform	DAST, WAF, monitoring	30 days	qualys.com/cloud-platform
Radware	AppWall	WAF, DDoS protection	N/A	radware.com/products/appwall
Radware	Alteon VA	Load balancing, DDoS protection	Free solution	radware.com/Resources/SoftwareDownloads
Rapid7	AppSpider Pro	DAST	Available by request	rapid7.com/products/appspider
Rapid7	Metasploit	Penetration testing	Free tier available	rapid7.com/products/metasploit
Rogue Wave Software	Klocwork	Code quality scanning	Available by request	roguewave.com/products-services/klocwork
RSA	RSA SecurID Suite	Identity management	Demo available by request	rsa.com/en-us/products/rsa-securid-suite
RSA	Threat Detection and Response	DAST	Demo available by request	rsa.com/en-us/products/threat-detection-and-response
Security Compass	SD Elements	Secure coding platform	Demo available by request	securitycompass.com/sdelements/how-it-works/#!/step/1
Securonix	Securonix Security Analytics Platform	Real-time threat detection	Demo available by request	securonix.com/products/securonix-security-analytics-platform
SentinelOne	SentinelOne	Endpoint security, DAST	Demo available by request	sentinelone.com/platform
ServiceNow	IT Service Management	Security monitoring	Demo available by request	servicenow.com/products/it-service-management.html

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
Signal Sciences	Signal Sciences Dashboard	RASP, WAF, real-time threat detection	Demo available by request	signalsciences.com/product
SiteLock	SiteLock TrueCode SAST	SAST, DAST	N/A	sitelock.com/truecode.php
SonicWall	SonicWall Cyber Security Products	WAF	N/A	sonicwall.com/en-us/products
Sophos	Sophos XG Firewall	WAF	30 days	sophos.com/en-us/products/next-gen-firewall.aspx
Splunk	Splunk Enterprise Security	Security monitoring & analytics	Demo available by request	splunk.com/en_us/products/premium-solutions/splunk-enterprise-security.html
StackPath	StackPath	WAF, DDoS protection	30 days	stackpath.com
Sucuri	Sucuri Website Firewall	WAF, DDoS protection, app security scanning	N/A	sucuri.net/website-firewall
Symantec	Cloud Access Security Broker	Cloud security testing & scanning	N/A	symantec.com/products/cloud-application-security-cloudsoc
Symantec Corporation	Symantec Advanced Threat Protection	IAST, RASP	60 days	symantec.com/products/threat-protection/advanced-threat-protection
Synopsys	Synopsys Static Analysis	SAST	N/A	synopsys.com/software-integrity/products/static-code-analysis.html
Tanium	Tanium Endpoint Platform	Endpoint security	Demo available by request	tanium.com/products
Tenable	SecurityCenter	Security monitoring & analytics, DAST	Available by request	tenable.com/products
ThinAir	ThinAir	Security monitoring and analytics	N/A	thinair.com/product
Threat Stack	Threat Stack	Cloud infrastructure security	Demo available by request	threatstack.com
Trend Micro	Trend Micro Deep Security Platform	SAST, DAST	N/A	trendmicro.com/us/enterprise/cloud-solutions/deep-security
Tripwire, Inc.	Tripwire Enterprise	IAST, RASP	Available by request	tripwire.com/it-security-software/scm/tripwire-enterprise

COMPANY	PRODUCT	PRODUCT TYPE	FREE TRIAL	WEBSITE
Trustwave	ModSecurity	WAF	Open source	modsecurity.org
Trustwave	Trustwave Secure Web Gateway	CDN, DAST	Demo available by request	trustwave.com/Products/Content-Security/Secure-Web-Gateway
Trustwave	Trustwave Web Application Firewall	WAF, penetration testing	Demo available by request	trustwave.com/Products/Application-Security/Web-Application-Firewall
UpGuard	UpGuard	Penetration testing, analytics	Demo available by request	upguard.com
Varonis	DatAdvantage	Insider threat detection/prevention	Demo available by request	varonis.com/products/datadvantage
Venafi	Venafi Platform	Device security identity management	N/A	venafi.com/platform
Vera	Vera Platform	Information rights managament	Demo available by request	vera.com/products/vera-for-files
Virtual Forge GmbH	CodeProfiler for ABAP	Code quality scanning	N/A	virtualforge.com/en/portfolio/codeprofiler.html
Waratek	Waratek	Application security for Java & .NET	Demo available by request	waratek.com
Webroot	BrightCloud Threat Intelligence by Webroot	DAST, endpoint security	30 days	webroot.com/us/en/business/threat-intelligence
WhiteHat Security	WhiteHat Sentinel	DAST	30 days	whitehatsec.com/products/dynamic-application-security-testing
WhiteHat Security	WhiteHat Sentinel	SAST	30 days	whitehatsec.com/products/static-application-security-testing
WhiteSource	Open Source Security	Open-source scanning	Open-source version available	whitesourcesoftware.com/open-source-security
Wireshark Foundation	Wireshark	Penetration testing & packet-level monitoring	Open source	wireshark.org
Ziften	Ziften Zenith	Endpoint security	Demo available by request	ziften.com/product-overview
Zscaler	Zscaler	Cloud security	Demo available by request	zscaler.com/products/zscaler-overview

GLOSSARY

APP TRANSPORT SECURITY (ATS)
A feature in the iOS operating system that enforces a minimum security level for communications (HTTPS vs. HTTP) between a mobile app and web services that support its functionality — enabled by default in SDKs for iOS 9.0 or later.

AUTHENTICATION
A mechanism that confirms a user's identity when they are requesting access to a resource in a system. This is generally handled by granting users an access token when they confirm their identity through a mechanism such as a password.

BRUTE-FORCE ATTACK
An attack that uses software to automate the guessing of numerous values very quickly in order to identify the correct value that will grant access to a protected resource.

CROSS-SITE REQUEST FORGERY (CSRF)
A malicious web exploit in which an attacking program forces a user's browser to perform an unwanted action on a site where the user is currently authenticated.

DATA EXFILTRATION
An unauthorized transfer of data. It can be carried out manually or through a malicious automated program.

DENIAL OF SERVICE ATTACK (DDOS)
A type of attack that uses multiple compromised systems that are forced to visit a website or system and overload its bandwidth in order to cause an outage.

DEVSECOPS
The integration of security into the DevOps methodology.

DYNAMIC APPLICATIONS SECURITY TESTING (DAST)
An analysis of an application's security that only monitors the runtime environment and the code that is executed in it. It simulates potential attacks and analyzes the results.

INJECTION ATTACK
A scenario where attackers relay malicious code through an application to another system for malicious manipulation of the application. These attacks can target an operating system via system calls, external programs via shell commands, or databases via query language (SQL) injection.

INTERACTIVE APPLICATION SECURITY TESTING (IAST)
A combination of SAST and DAST that is usually implemented in the form of an agent that monitors attacks and identifies vulnerabilities within the test runtime environment.

KEYCHAIN
Within the iOS operating system, a keychain encrypts and securely stores data used by apps and services. When the keychain is unlocked via a password, only trusted applications can read that data.

MAN-IN-THE-MIDDLE ATTACK
An attack whereby an adversary inserts themselves between an application and its backend services to eavesdrop on the communications between them.

PASSWORD-BASED KEY DERIVATION FUNCTION (PBKDF2)
An algorithm that makes passwords stronger by encrypting a password along with a "salt" value (i.e. random data) numerous times to make the password harder to guess.

RUNTIME APPLICATION SELF-PROTECTION (RASP)
A feature that is built into an application in order to detect and halt attacks in real-time, automatically.

RUNTIME PACKER
A tool that compresses software that will execute once the compressed file is opened. This is frequently used as a way to distribute malware.

SECURE SOCKETS LAYER (SSL)
An encrypted link that serves means to keep information that is passed between the web server and browsers private.

STATIC APPLICATION SECURITY TESTING (SAST)
An analysis of an application's security that looks at an application's source code, bytecode, or binary code to determine if there are parts that could allow security exploits by attackers.

WEB APPLICATION FIREWALL (WAF)
An appliance or application that monitors, filters, and blocks HTTP transmissions to a website based on customizable rules.

INTRODUCING THE

Open Source Zone

Start Contributing to OSS Communities and Discover Practical Use Cases for Open-Source Software

Whether you are transitioning from a closed to an open community or building an OSS project from the ground up, this Zone will help you solve real-world problems with open-source software.

Learn how to make your first OSS contribution, discover best practices for maintaining and securing your community, and explore the nitty-gritty licensing and legal aspects of OSS projects.

COMMITTERS & MAINTAINERS

COMMUNITY GUIDELINES

LICENSES & GOVERNANCE

TECHNICAL DEBT

Visit the Zone

BROUGHT TO YOU IN PARTNERSHIP WITH Flexera