aws marketplace

SANS

Securing a Modern Web Application in AWS

Explore threat modeling and learn how to create and support your web application security strategy with AWS Marketplace software seller solutions.

AWS Marketplace Introduction

As more organizations turn to distributed web applications to maintain high availability and reduce costs, many are choosing to store these applications in the AWS cloud for added elasticity, scalability, and ability to handle large workloads. Doing this securely, however, means addressing potential threats to multiple components, such as the front-end cloud application and corresponding databases.

In this whitepaper, SANS analyst and instructor, Shaun McCullough, will provide an introduction to exploring the vulnerabilities associated with modern web applications, web application firewalls, and DevSec operations that oversee security to continually update code. This process, known as threat modeling, is vital to the ability to prioritize vulnerabilities and security operations to meet those challenges.

Building on Shaun's perspective, AWS Marketplace shares how this process can be applied to your AWS Cloud environment with an introduction to relevant AWS security services and AWS Marketplace software sellers, such as Fortinet, Barracuda, and Imperva.

The featured Fortinet solutions for this use case can be accessed in AWS Marketplace

Fortinet Managed Rules for AWS WAF
AWS Quick Start for Fortinet FortiGate
Fortinet FortiWeb Cloud WAF-as-a-Service

SANS

A SANS Whitepaper

How to Protect a Modern Web Application in AWS

Written by **Shaun McCullough**

April 2019

Sponsored by:

AWS Marketplace

Introduction

As businesses move more assets to the cloud, having a security plan is essential, but nobody has the time or resources to do everything that is needed from the start. Instead, organizations need to prioritize their security plans based on the risks to which they are exposed. Too often, organizations start with securing the service they know best or have read about in a blog, or they try to buy their way out of the risks with multiple, expensive security appliances.

While the team is knee-deep in transitioning core services, security takes a back seat. It's confusing to understand where the cloud service provider's responsibility ends and the customer's responsibility begins, or how best to secure the services and leverage new tools properly.

Prioritizing the risks, and hence determining what should be secured first, can be simplified through *threat modeling*—the process of identifying and prioritizing the risks to infrastructure, applications and the services they provide. A proper threat model allows organizations to identify applicable risks, prioritize those risks and evaluate how to manage changes in risks over time.

Implementing threat modeling in the cloud is similar to implementing for a traditional infrastructure, but the cloud services, risk priority levels and potential solutions can be vastly different. A threat against a web application stack will be the same in the cloud as it is when deployed on premises. However, cloud providers offer new tools to address

the risks. Security teams can bring together cloud-native services, centralized logging, new identity access management processes and easy-to-implement third-party services to make applications and infrastructures safer.

This paper is a use case of modeling the threats against a web application server and how to address those risks in a cloud environment. We will cover the web app stack, including the web server, the application code, and the DevOps pipelines to manage it. Database threats will be covered in future papers in this series. We'll examine the tools and services that cloud providers offer to operate web applications at scale and integrate security services. The paper also breaks down the DevOps process, explains how it can be threat-modeled, and describes common security risks and improvements over traditional workflows.

Building a culture of threat modeling prepares organizations to address the most significant threats with limited resources.

A Threat Modeling Primer

As defined in a special publication by the National Institute of Standards and Technology (NIST), threat modeling is "a form of risk assessment that models aspects of the attack and defense sides of a particular logical entity."[1] By implementing a threat modeling process, organizations can improve their security posture, identify unrealized risks and provide their leadership with the proper tools to prioritize which risks to focus on first.

Threat Modeling Process and Frameworks

Most threat models start in one of two ways:

- Identifying a set of attacker techniques the organization is at risk from
- Identifying a set of deployed assets that are at risk

Organizations need to pick the approach that works best for them, but asset-focused threat modeling is usually the most straightforward.

Threat modeling is a process, not a one-time whiteboard session on a Monday afternoon. As the threats evolve, so do an organization's risk appetite and security implementations, along with the experience of the team. Organizations must create a culture of threat modeling, where the model is evaluated, implemented, tested, reviewed and re-evaluated regularly.

The first threat model an organization builds could take time and even be painful. As the team gains experience, the process becomes more natural and standardized. Security teams should hold quarterly reviews to make updates, question assumptions and adjust risks. Teams should also perform a yearly re-evaluation of the whole

Drivers of Threat Prioritization

Prioritizing threats is often tricky and likely influenced by the expertise or culture of the organization. If the network team is seasoned, runs a stable environment and has the time to research new threats, it can create the most detailed plan for reducing security risks in the team's responsibility area. In contrast, a host team caught in the middle of a complicated operating system upgrade has no time to think of next week's risks, much less next year's. The organizational culture, workloads, expertise and maturity drive how organizations respond to threats. A threat model process helps level the playing field by giving the appropriate team members the space, tools and support to think about risks and threats across the organization.

[1] Draft NIST SP 800-154, Guide to Data-Centric System Threat Modeling, https://csrc.nist.gov/publications/detail/sp/800-154/draft

threat model, with all the experts available. Regular reviews of the threat model help organizations understand whether the risk-reduction plans are working.

Among the various threat modeling frameworks, the DREAD risk assessment model works well. Used at OpenStack, DREAD helps teams evaluate the potential results of an attack. DREAD helps the team walk through how a system is at risk, what the attack vector looks like, how likely the attack is to occur and how to prioritize which risks to focus on.

Threat modeling is a process, not a one-time whiteboard session on a Monday afternoon.

The IANS Pragmatic Threat Modeling Toolkit is a spreadsheet that helps organizations walk through the DREAD framework. Users can identify assets at risk, work through DREAD rankings and graph results for easier understanding.[2]

Risk Assessment and Prioritization

Every risk in an environment is addressed in one of four ways, as illustrated in Figure 1.

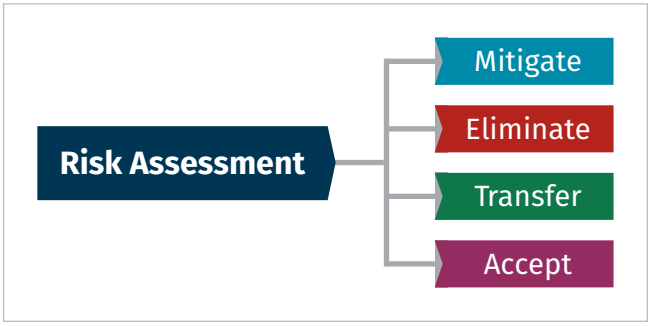


Figure 1. Risk Management Strategies

Mitigate—Putting a firewall in front of your web server will mitigate some attacks, but not all of them. Most security controls focus on mitigating risks.

Eliminate—Eliminating a risk will likely require changing the nature of the asset at risk in such a way that the risk fundamentally goes away. A firewall cannot eliminate all scripting attacks against a web application, but removing all data entry fields and making the website completely static will certainly eliminate whole categories of attacks. Eliminating risks is ideal, but difficult—and usually means re-architecting.

Transfer—When an organization decides to move on-premises infrastructure to a cloud provider, it is effectively transferring asset risks to the service provider. The organization is making a business decision to pay for the provider to manage, secure, provision or operate the service. Cloud providers operate on a shared responsibility model. From a security perspective, that means that parts of the infrastructure stack have been transferred to the cloud provider. It is now responsible for operating, security and managing the assets.

Serverless technology is a good example of transferring risk and taking advantage of this shared responsibility model. A customer could spin up virtual machines in the cloud, managing the full stack from operating system to application. The customer is responsible for the patching, configuration and security monitoring of that virtual machine operating system, while the cloud provider is responsible for the virtualization infrastructure, storage and network. Serverless offerings allow the customer to execute a bundle of code, yet have no direct interaction with the executing operating system. The service provider manages the servers in a serverless offering. The risk of operating system vulnerabilities is now transferred to the cloud provider.

[2] IANS Pragmatic Threat Modeling Toolkit, https://portal.iansresearch.com/media/739278/ians_pragmatic_threat_modeling_toolkit.xlsm

Accept—If an organization is unable to mitigate, eliminate or transfer the risk, then it is accepting that risk. It might be a temporary acceptance to be re-evaluated later. In the threat model process, it is healthy for the organization to understand that accepting risk is a valid option that frees it to plan, prioritize, and dive into the other risks.

As an organization gets more comfortable with its threat model process, it should start incorporating the model into the beginning of the development cycle, helping to identify risks that need to be mitigated or eliminated before the organization has invested the time in creating and deploying it. Security teams that work separately from those who create the systems are fighting an uphill battle that will impair effectiveness while raising costs. Include the whole team when modeling a set of services. The developers likely can suggest and implement ways to significantly reduce the risk scores.

Building threat models for IT-operated application services will help with prioritizing and accepting risks.

Building threat models for IT-operated application services will help with prioritizing and accepting risks. Cloud services offer new opportunities for customers to mitigate, eliminate or transfer those risks for traditional IT service applications and to establish new workflows for developing and deploying those systems through DevOps.

DevOps with Security

DevOps is a process that enables close coordination between development and operation teams.[3] That integration enables organizations to develop and quickly deploy new services with zero downtime and improved reliability. The process is especially beneficial for organizations that deploy new versions of software multiple times a day.

To incorporate DevOps, organizations rework testing and deployment processes to be safe, automated and executable at any time. Continuous Integration is the process by which software changes from multiple developers are integrated into a single stack, likely multiple times a day. With *Continuous Integration*, security teams can avoid the big end-of-a-sprint integration sessions that cause delays and waste resources. *Continuous Deployment* is the process of building software to be releasable into production at any time, with an easy push of the button.

Companies using on-premises environments have been leveraging DevOps processes to create close coordination between the developers, who create new applications, and operations, which provides the virtual machines they run on. The cloud brings a whole host of services to automate all aspects of the infrastructure deployment and management that on-premises services are unable to match.

Continuous Integration and Continuous Deployment (CI/CD) require organizations to rethink their planning, development and deployment pipelines to be highly automated. See Figure 2.

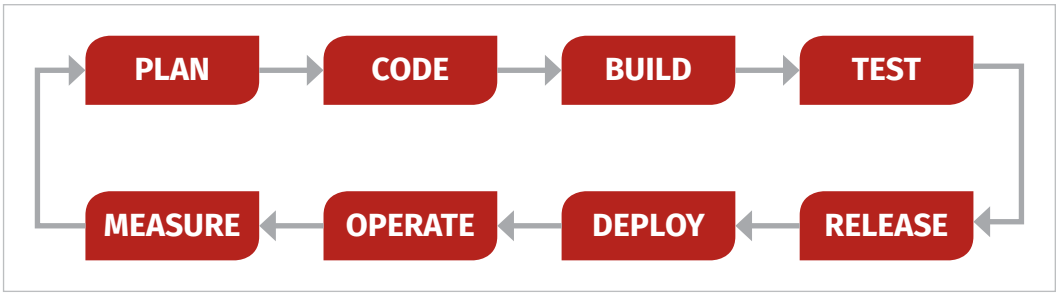


Figure 2. Continuous Integration and Continuous Deployment (CI/CD)

[3] NIST SP800-190, https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-190.pdf

With CI/CD, every evaluation, decision, configuration or security test that can be automated is automated. If these processes cannot be automated, then the development team must rework the architecture.

DevSecOps takes the DevOps process and builds in automated security evaluation gates. The "Sec" of DevSecOps requires the organization to establish security policies for the product before development starts, implementing them in the testing and deployment pipelines. Automated tests are security policies that become reality, not just words in a binder. The best CI/CD processes incorporating DevSecOps give developers the tools to test the security of their code at their workstations—at the beginning of the process rather than waiting until the end of development and being surprised.[4]

CI/CD is usually focused on deploying applications automatically and continuously. However, the cloud opens a whole new area, allowing the automatic provisioning and deployment of core infrastructure itself. The cloud provides APIs, development kits and specialized services that let customers control every aspect of the infrastructure with DevOps-like processes and tooling.

Imagine creating an infrastructure pipeline where a configuration file is used to build a web application stack. And say that a new version of the web server is released with a software patch, and you want to deploy it. After testing it locally, the team updates the configuration file and checks it into version control, and a CI/CD pipeline kicks in and replaces all deployed web servers with the updated versions—automatically.

CI/CD comes with risks, however. Automating processes traditionally done by humans can reduce errors, but it also hides unforeseen problems. The platforms that implement DevSecOps and CI/CD pipelines are new attack vectors. The CI/CD platform must become part of the threat modeling process for an organization to ensure that the entire infrastructure is evaluated.

Threat Modeling a Web Application

As previously discussed, the threat model process starts with identifying deployed assets that are at risk—assets that are well understood and vital to the business. As part of our use case, let's model the threat to the web application itself and investigate a threat model for the web application.

Risk of Web Application Attacks

Web applications are usually at risk—they live on the internet, with the sole purpose of capturing and providing information to all their users living on untrusted networks. Complex web applications with user access controls, database-backed pages and free-form input fields are notorious for their vulnerabilities.

The Open Web Application Security Project (OWASP) Top 10[5] is the best starting place when analyzing threats against web applications. Top attack techniques are prioritized, researched and documented, with details of how the attack works and suggested best practices for stopping the attacks.

---

[4] *Accelerate: Building and Scaling High Performing Technology Organizations*, by Nicole Forsgren, Jez Humble and Gene Kim (IT Revolution, 2018)

[5] OWASP Top Ten Project, www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

Cross-site scripting (XSS) is a common attack on web applications that the OWASP Top 10 – 2017 report describes:

> XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.[6]

Use Case: Spoofing an Identity

Web applications require data inputs and dynamically display information back to users. XSS could result in many different threat categories. For this use case, an XSS attack that exposes other users' browser session credentials can be used to spoof an identity.

After categorizing the threat, a team can evaluate the risk using the DREAD model. Each DREAD risk-rating category is given a value from 1 to 10. Figure 2 describes the ratings.

Damage Potential—How much damage will occur if this vulnerability is compromised?

- 0 = None
- 3 = Individual user data is compromised or affected, or availability is denied
- 5 = All individual tenant data is compromised or affected, or availability is denied
- 7 = All tenant data is compromised or affected, or availability is denied
- 7 = Denied availability of a component/service
- 8 = Denied availability of all components/services
- 9 = Compromised underlying management and infrastructure data
- 10 = Complete system or data destruction, failure or compromise

Reproducibility—How reliably can the vulnerability be exploited?

- 0 = Very hard or impossible, even for administrators; the vulnerability is unstable and statistically unlikely to be reliably exploited
- 5 = One or two steps required; tooling/scripting readily available
- 10 = Unauthenticated users can trivially and reliably exploit using only a web browser

Exploitability—How difficult is the vulnerability to exploit?

- 0 = N/A We assert that every vulnerability is exploitable, given time and effort; all scores should be 1-10
- 1 = Even with direct knowledge of the vulnerability, we do not see a viable path for exploitation
- 2 = Advanced techniques required, custom tooling; only exploitable by authenticated users
- 5 = Exploit is available/understood, usable with only moderate skill by authenticated users
- 7 = Exploit is available/understood, usable by non-authenticated users
- 10 = Trivial—just a web browser

Affected Users—How many users will be affected?

- 0 = None
- 5 = Specific to a given project
- 10 = All users

Discoverability—How easy is it to discover the threat, to learn of the vulnerability? (By convention this is set to 10 even for privately reported vulnerabilities.)

- 0 = Very hard to impossible to detect even given access to source code and privileged access to running systems
- 5 = Can figure it out by guessing or by monitoring network traces
- 9 = Details of faults like this are already in the public domain and can be easily discovered using a search engine
- 10 = The information is visible in the web browser address bar or in a form

Figure 2. DREAD Risk Ratings[7]

[6] The 10 Most Critical Web Application Security Risks, www.owasp.org/images/7/72/OWASP_Top_10-2017_(en).pdf.pdf

[7] Adapted from DREAD Rating, https://wiki.openstack.org/wiki/Security/OSSA-Metrics#DREAD

The rating of a single threat does not provide a full picture of the organization's vulnerable landscape. DREAD ratings of multiple risks should be viewed in tandem to get a complete picture of the risks that need to be prioritized. While informed by the DREAD rating guidance, organizations will arrive at their final rating number/prioritization through a combination of the ratings and their own experiences, knowledge and biases. Table 1 shows the DREAD rating for our use case.

Table 1. DREAD Rating for Web Application

Category	Rating	Spoofing Identity
Damage Potential	2	The business unit is a significant driver of the risk rating for an application. What data does the application hold? How far-reaching would the attack be? How important is the asset itself? In this example, an XSS attack to gain credentials does not do any damage itself.
Reproducibility	7	Once identified, an XSS attack is easy to reproduce through scripts. Only common application access is necessary, rather than special access privileges.
Exploitability	4	Depending on the vulnerability of the application, an XSS could be easy or hard to exploit. Discoverability rates how easy it is to determine if there is potential for an XSS; however, making the exploit perform the desired identity spoofing can be tricky, so we will rate this lower.
Affected Users	4	An XSS attack affects the users logged into the application at the time of the attack, and potentially any users who view the corrupted data. Some users will be affected, but not all.
Discoverability	7	Entering JavaScript into a webpage and reviewing the results gives an attacker a good idea if there is an XSS vulnerability, even if they cannot complete the exploit.
DREAD Average	**4.8**	

Because XSS is a well-known and well-researched attack method, security teams have multiple ways to mitigate the risk of an XSS attack on a web server. A popular security control is incorporating a web application firewall (WAF) to monitor and block any suspicious traffic before it reaches the web server.[8] Large cloud service providers make it easy to implement a WAF right from the console. AWS's WAF service allows you to customize rules and access control lists to fit your business and risk models.

Larger cloud service providers may offer WAF assets that can be integrated into their service offerings. They are easy to set up, are relatively inexpensive, and should be able to block OWASP Top 10 and other common attacks. If the DREAD risk is higher and more protection is needed, the cloud service provider often has a variety of top-tier third-party products with WAF offerings available for installation (for example, Impreva SecureSphere and Fortinet FortiGate).[9] One way to eliminate the risk of XSS is to remove data entry fields altogether. It requires rethinking the web application architecture and possibly removing functionality for the sake of security. If eliminating the data entry fields is not viable, you can transfer that ownership to a third party. For instance, if the data input fields are for user authentication, leverage a third-party single sign-on service. Eliminating and transferring risks tends to be more costly, but will help decrease DREAD risk scores. The bottom line is that the threat modeling process should drive prioritization of assets and financial commitments.

[8] Web Application Firewall, www.owasp.org/index.php/Web_Application_Firewall

[9] This paper mentions product names to provide real-life examples of how varying classes of tools can be used. The use of these examples is not an endorsement of any product.

Use Case: SQL Injection Attack

Modern web applications are driven by databases that can contain a wealth of knowledge that attackers want. A SQL injection tricks the database into returning unintended data.[10] One outcome of a SQL injection attack is *information disclosure*. The DREAD rating determines the severity of this attack in the environment. See Table 2.

Table 2. DREAD Rating for Database

Category	Rating	Information Disclosure
Damage Potential	7	A SQL injection, if successful, will likely affect all the data in the database, not just specific users. The actual damage done in information disclosure is another measure that requires the business units to weigh in.
Reproducibility	7	Once a SQL injection attack is identified, it is repeatable.
Exploitability	5	SQL injection (or NoSQL) tends to be easier to accomplish than XSS.
Affected Users	2	Other users may not even notice if a SQL injection attack is happening unless it is damaging the data. For an information disclosure categorized attack, the user effect is nominal.
Discoverability	6	Like XSS, the SQL injection vulnerability is easier to identify than actually to exploit.
DREAD Average	**5.4**	

The processes for mitigating a SQL injection and XSS attacks are similar. The SQL injection attack comes through the web application itself; thus the WAF is in a position to identify and block potential SQL injection attacks. Not all SQL injection attacks will be detected, and significant research has gone into countering a WAF.[11] When deciding on a WAF product, look at the entire threat model process and ensure that the WAF covers all the threats at the same time.

Another option is to leverage secure coding practices to develop safer code that neutralizes invalid text field inputs before being run in the SQL query on the database. Depending on the programming languages, a number of libraries, design patterns and tools can do this. The security team will need to ensure that all code is following these standards or incorporating the right tools. Today, CI/CD platforms provide opportunities to continuously scan, evaluate or test code as it is being developed.

Now that we've looked at modeling the threat to the web application, let us look at the threat to the development and deployment platform that is used in cloud operations.

Threat Modeling the DevSecOps Platform

We have looked at threat models for a well-known architecture like the web application. Now let's walk through a practical threat model of a CI/CD platform. Again, DREAD helps to prioritize the risks.

A CI/CD process is all about safely automating workflows. The Continuous Integration process kicks off when a developer checks code into the designated source code repository. Distributed version control systems (DVCSs) will mirror an entire copy of the codebase, including all history, on every developer's computer.[12] Git is the most

[10] SQL Injection: Modes of Attack, Defence, and Why It Matters, www.sans.org/reading-room/whitepapers/securecode/sql-injection-modes-attack-defence-matters-23

[11] SQL Injection Bypassing WAF, www.owasp.org/index.php/SQL_Injection_Bypassing_WAF

[12] Getting Started—About Version Control, https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control

popular DVCS in use today, used with a central Git repository management system like GitHub, GitLab or AWS CodeCommit. When developers request to check their code into the designated central repository, the Continuous Integration system kicks off to test the integration to ensure that it does not break the application. See Figure 3.

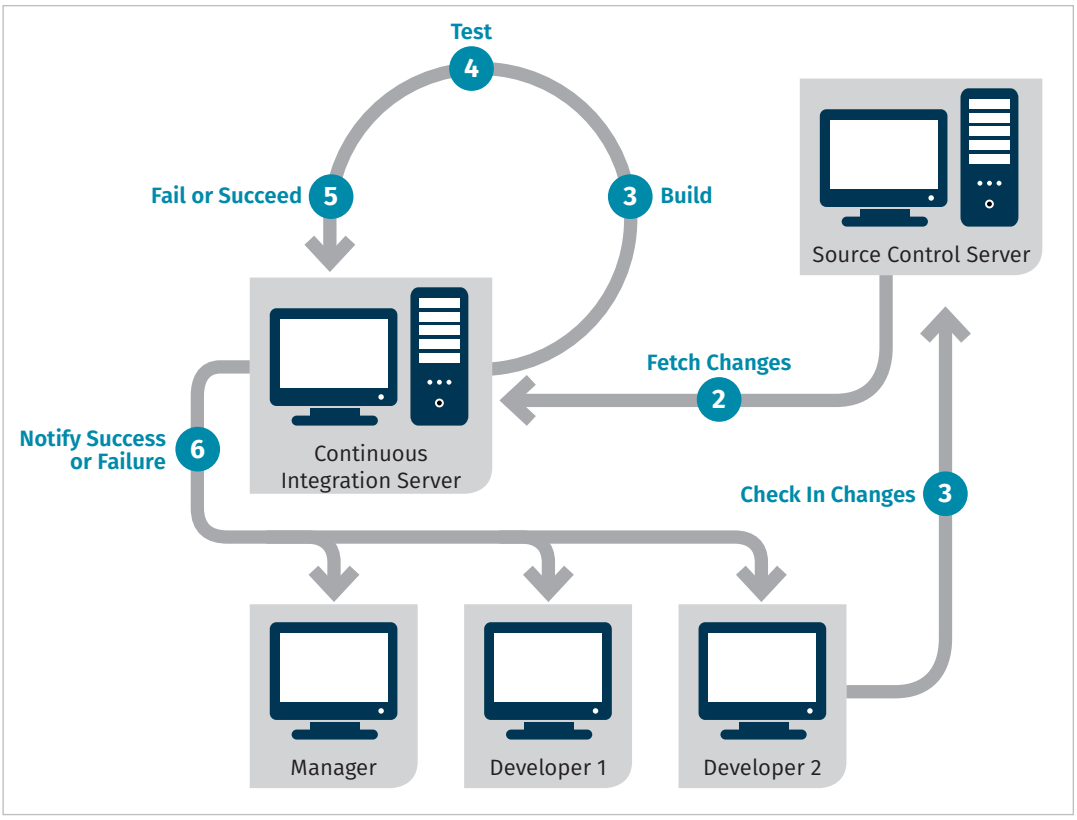


Figure 3. Continuous Integration Process

Use Case: Credential Disclosure

Web applications can make database connections directly to query for data. Many times, the web application connects to the database through credentials stored in a configuration file on the application's server. The developers have an instance of the database in their environment for testing, which may include a small copy of production data to test code changes properly.

If that credential file is accidentally checked into the source control system, that configuration file could become visible to unauthorized users—especially with open source software where the DVCS is accessible to the public. Disclosure of credentials can lead to an unauthorized login to the database, called "identity spoofing." Using the spoofed identity can then lead to additional information disclosure, tampering of data or even denial of service. Identifying each step and categorizing the actions along the way is building up the attack tree.[13] See Table 3.

Table 3. DREAD Rating of Credential Disclosure

Category	Rating	Credential Disclosure
Damage Potential	5	The damage from information disclosure varies depending on the value of the credentials themselves. In this use case, the credentials at risk are for the development environment and reside on the developer's machine. Because this test database contains a snapshot of production data for testing, customer data is at risk.
Reproducibility	8	The threat exploited is highly reproducible because the attacker can log into the at-risk asset.
Exploitability	8	Logging in with unauthorized credentials is easy when you have the credentials.
Affected Users	5	The database at risk in this particular threat model is a developer's test environment with limited production data.
Discoverability	9	The software is continuously scanning source code repositories looking for credential-like data, thus discovering the data could take mere minutes.
DREAD Average	**7**	

[13] Attack Trees, www.schneier.com/academic/archives/1999/12/attack_trees.html

As the developer is checking in new code in a Continuous Integration process, it is possible that the developer will accidentally check in that credential file and risk disclosure. If undetected, exposure is guaranteed.[14]

In CI/CD, the automated test platform could be used to evaluate the code to look for strings that resemble credentials and reject the merge. These tools are inexpensive and are easy to configure and execute; they fit perfectly with the CI/CD process and will mitigate the credential disclosure risks.

To eliminate the risk of credentials being checked in, eliminate the credential file. Secrets management systems, which are available from cloud service providers or through the marketplace, can be used to programmatically store credentials and only provide them to applications that are authorized. Although this risk-reduction will be harder to implement and can cause changes to the asset, eliminating a risk versus mitigating that risk might be worth the cost.

Use Case: Software Vulnerability to Denial of Service

Humans write software, and humans are experts at making mistakes. Security professionals are continually patching, monitoring and managing software updates. To make matters worse, developers are increasingly reliant on software packages distributed by other developers. Code actually written by the development team may be a small percentage of the entire code base for the application. For this threat model, teams must evaluate the risk of a vulnerable third-party NodeJS module making its way into the software stack.

Node Package Manager (NPM) is the most widely used NodeJS package delivery tool, and is likely what organizations are using for JavaScript-based frameworks. A vulnerable NodeJS module can cause information disclosure, escalation of privileges or denial of service.[15] Let's look at denial of service and rate the DREAD risks, as shown in Table 4.

Table 4. DREAD Rating of Software Vulnerability

Category	Rating	Denial of Service
Damage Potential	7	The amount of damage caused by a denial of service is a business-unit-led decision. Is this a core part of the organization's business? Could it go down for a day and see no real effects? Business drivers are just as important as security risks in the threat model process. Knowing how vital each service is to the business helps define these values. For this use case, the product is a core part of the business and could not go down for any length of time.
Reproducibility	5	Reproducibility can be difficult because the exploit in the NodeJS module could be easy or hard to implement depending on what it is. Predicting future vulnerabilities is impractical. The threat modeling team will have to decide how to handle these ambiguous ratings and be consistent.
Exploitability	5	Similarly, exploitability is hard to assess.
Affected Users	8	The number of affected users can be significant. Denial of service attacks against production systems may slow down or even stop customers from using the application.
Discoverability	3	Because this use case is not an open source application, it will be difficult for an attacker to discover that an application has a particularly vulnerable NodeJS package.
DREAD Average	**5.6**	

[14] I accidently pushed sensitive info, https://github.community/t5/How-to-use-Git-and-GitHub/I-accidentally-pushed-sensitive-info/td-p/225

[15] NPM security advisories, www.npmjs.com/advisories

It can be difficult to know if a vulnerability exists in any included NodeJS packages. Although the vulnerability may not exist in the packages themselves, each of those packages could rely on other packages, which could be vulnerable. The CI/CD platform must continually analyze deployed modules for vulnerabilities discovered post-deployment.

Some code scanner products are available, usually as scriptable software applications that can be run by any CI/CD platform. Commercial versions provide a wealth of threat intelligence and software analysis and are able to not only identify reported vulnerabilities but also scan deep into the code itself and identify risky functions or statements. The code scanners should be easy to run with the CI/CD platform. When developers integrate their code, third-party vulnerability scanners could scan before acceptance. After deployment, the entire code base should be tested daily for newly discovered vulnerabilities that can flag to the security team.

Expanding on this idea, the entire deployment system can be scanned before deployment. In a cloud service environment, the configuration of the infrastructure itself can be managed by code, using tools such as AWS CloudFormation or HashiCorp's Terraform. When a configuration is changed, a sample virtual machine can be automatically built, then scanned by vulnerability scanning tools to ensure that no known vulnerabilities exist in the packages. Third-party scanners have cloud-ready services that can be initiated by CI/CD in the cloud. The results can be used by the CI/CD to determine if a deployment should continue—all automatically.

The risk model can help inform decision makers on whether to use free or commercial solutions. Investigate what additional services and intelligence the commercial products provide, whether they will be easier to implement and operate, and how they might work in the build process. Remember, the risk scores from the threat modeling process and the priorities they uncover can help direct where to focus time and money.

Summary

Start building a threat model process as part of the security culture of your organization and reap the benefits throughout the life of your infrastructure. Focus on identifying the threats, the risks they pose, and the relative business importance to help the organization prioritize where to focus attention and resources. The automation of the integration and deployment processes of applications means security policies need to be identified and implemented at the beginning of the development cycle, not the end.

Threat modeling is a great process for identifying risks. We recommend that any threat modeling process do the following:

- Prioritize risks so organizations know where to focus investment.
- Produce concrete plans to mitigate, eliminate or transfer any risks that will not be accepted.

- Bring security into the beginning of system development rather than at deployment time.
- Create a repeatable, improvable process that is used to make decisions, not just a checkbox.
- Document not just the plan but also the risk-reduction results. A threat model process can help organizations understand how effective they are in planning, monitoring, addressing and measuring risks.

As your threat model process matures, teams can start to evaluate risks in systems before they are even developed. Architectural decisions to eliminate a risk rather than only mitigate it will improve security and likely reduce overall operating costs. And as automated DevSecOps platforms are brought into the organization's workflow, a whole host of risks can be managed automatically.

Adapt a good threat model process that works for your organization. Constantly re-evaluate, improve and expand the process until the organization can see measured results from planned risk reductions.

About the Author

Shaun McCullough is a community instructor for the SEC545 Cloud Security Architecture class and gives back to his profession by mentoring and supporting the next generation of cyber professionals. With 25 years of experience as a software engineer, he has been focusing on information security for the past 15 years. Shaun is a consultant with H&A Security Solutions, focusing on secure cloud operations, building DevSecOps pipelines and automating security controls in the cloud. He also served as technical director of red and blue team operations, researched advanced host analytics, and ran threat intelligence on open source platforms in his work with the U.S. Department of Defense.

Sponsor

SANS would like to thank this paper's sponsor:

aws marketplace

How Organizations are Strengthening Their Web Application Security in AWS

The first step in prioritizing your organization's security controls is understanding which AWS native services can most effectively be leveraged to achieve compliance and strengthen your overall security posture.

Secondly, organization's moving their workloads to the cloud will need to prioritize the controls that extend past the available AWS native services. In these cases, third-party solutions exist to address this concern via AWS Marketplace. AWS Marketplace, a digital software catalog where security practitioners can find, try, buy, deploy, and manage software that runs on AWS, offers third-party software solutions that can be integrated with AWS native services and other existing technologies. This can enable organizations to deploy a comprehensive security architecture across the AWS Cloud and within traditional on-premises environments.

Securing web applications starts with having the right tools

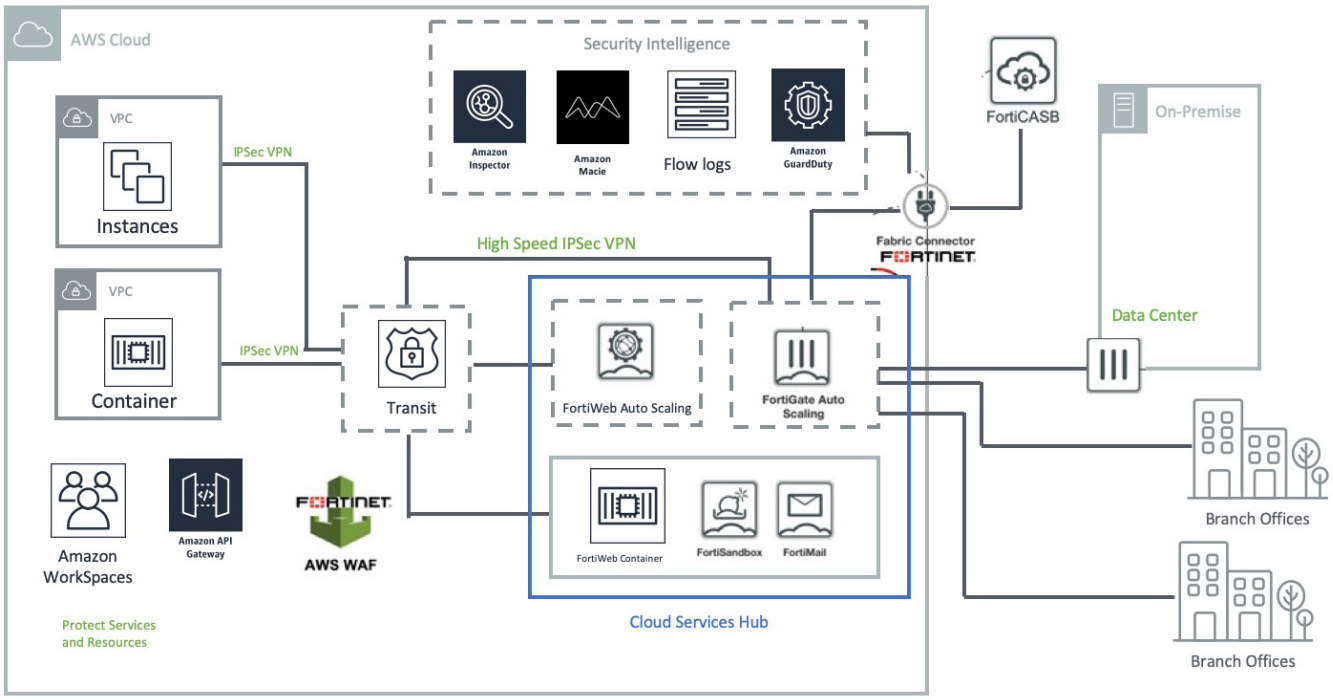
Web application security is a top priority for AWS and many of our customers. AWS handles security with the shared responsibility model. When you are running workloads on AWS, we handle everything from the physical security of our data centers all the way up to the hypervisor. Customers are responsible for building secure applications on AWS, as well as configuring AWS features for services such as AWS Shield and AWS WAF.

AWS Shield is a managed Distributed Denial of Service (DDoS) protection service that safeguards applications running on AWS. AWS Shield provides always-on detection and automatic inline mitigations that minimize application downtime and latency.

AWS WAF is a web application firewall that helps protect your web applications from common web exploits that could affect application availability, compromise security, or consume excessive resources.

There are also many third-party solutions available that can assist you with your web application security goals. A popular vendor in AWS Marketplace is Fortinet, who offers multiple solutions to enhance web application security.

- **Fortinet Managed Rules for AWS WAF:** Instead of developing custom rules, you can save time while protecting your web application or APIs against common threats, such as OWASP Top 10 security risks, threats specific to Content Management Systems, or emerging Common Vulnerabilities and Exposures (CVE) by using managed rule groups.
- **AWS Quick Start for Fortinet FortiGate:** This Quick Start automatically deploys Fortinet FortiGate Auto Scaling Baseline into a new or existing virtual private cloud (VPC) in the AWS Cloud. It is intended to be a baseline for those who plan to implement or extend Fortinet's Security Fabric workloads in the AWS Cloud.
- **Fortinet FortiWeb Cloud WAF-as-a-Service:** FortiWeb Cloud WAF SaaS enables rapid application deployments in the public cloud while addressing compliance standards and protecting mission-critical hosted applications.
- **Fortinet Cloud Security Services Hub with AWS Transit Gateway:** For AWS Cloud users, Fortinet-enabled cloud services deliver a variety of security capabilities from a central location. These are integrated via the Fortinet Security Fabric. These services all leverage a cloud-native ability to automatically scale and replicate services in other regions. The Transit Gateway also helps solve another common challenge: interconnecting VPCs.



There are a variety of other vendors and solutions in AWS Marketplace that also address this challenge, such as Barracuda CloudGen WAF and Imperva SecureSphere WAF.

aws marketplace

Why use AWS Marketplace?

AWS Marketplace simplifies software licensing and procurement by offering thousands of software listings from popular categories like Security, Networking, Storage, Business Intelligence, Machine Learning, Database, and DevOps. Organizations can leverage offerings from independent security software vendors in AWS Marketplace to secure applications, data, storage, networking, and more on AWS, and enable operational intelligence across their entire environment.

Customers can use 1-Click deployment to quickly launch pre-configured software and choose software solutions in both Amazon Machine Image (AMI) formats and SaaS subscriptions, with software entitlement options such as hourly, monthly, annual, and multi-year.

AWS Marketplace is supported by a global team of security practitioners, solution architects, product specialists, and other experts to help security teams connect with the software and resources needed to prioritize security operations in AWS.

How to get started with security solutions in AWS Marketplace

Security teams are using AWS native services and solutions from independent software vendors in AWS Marketplace to help build automated, innovative, and secure solutions to address relevant use cases and further harden their cloud security posture. The following steps can help you get started:

Browse free trials of the solutions mentioned above by clicking on the logos below