



# TECHORAMA

The word "TECHORAMA" is written in a stylized, colorful font. The letters are primarily red and orange, with blue outlines and a wavy pattern. They are set against a background of horizontal stripes in red, orange, and light blue, separated by thin white lines. Three blue stars are positioned above the letter "E" and three more are positioned below the letter "R".

## Security 101

A Beginner's Guide to Cybersecurity and Zero Trust for Developers

# Security 101

A Beginner's Guide to Cybersecurity and Zero Trust for Developers

## **Agenda:**

In this session, we will delve into fundamental cybersecurity concepts, explore the concept of zero trust, and discuss key themes across various security domains. This session is especially for developers and will focus on how they can improve their company's security posture without having security slowing them down. The session will cover the following topics:

### **Basic Cybersecurity Concepts:**

- Understanding the CIA triad: Confidentiality, Integrity, and Availability.
- Differentiating between risks and threats.
- Exploring the role of security controls in safeguarding systems and data.

### **Zero Trust: A Paradigm Shift**

- Defining zero trust and its principles.
- Why zero trust matters in today's interconnected and dynamic digital landscape.
- How to implement zero trust strategies effectively.

### **Key Concepts Across Security Domains**

- Identity and Access Management (IAM): Managing user identities and permissions.
- Networking Security: Protecting networks from threats.
- Security Operations (SecOps): Incident response, monitoring, and threat hunting.
- Infrastructure Security: Securing servers, cloud resources, and endpoints.
- Data Security: Safeguarding sensitive information.

### **Examples of Security Tools**

- Firewalls: Network security appliances.
- Intrusion Detection Systems (IDS): Detecting suspicious activity.
- Endpoint Protection: Securing devices.
- Encryption Tools: Protecting data in transit and at rest.

### **The Consequences for Developers**

- Securing development environments and pipelines.
- Manage the Software Supply chain and automate the update vulnerable dependencies.
- Automatically securing your code and infrastructure as code using code scanning tools.
- Prevent leaked credentials and manage / rotate credentials in a secure way.

Attendees will gain a solid understanding of cybersecurity fundamentals, learn about zero trust principles, and discover practical tools to secure the entire development lifecycle.

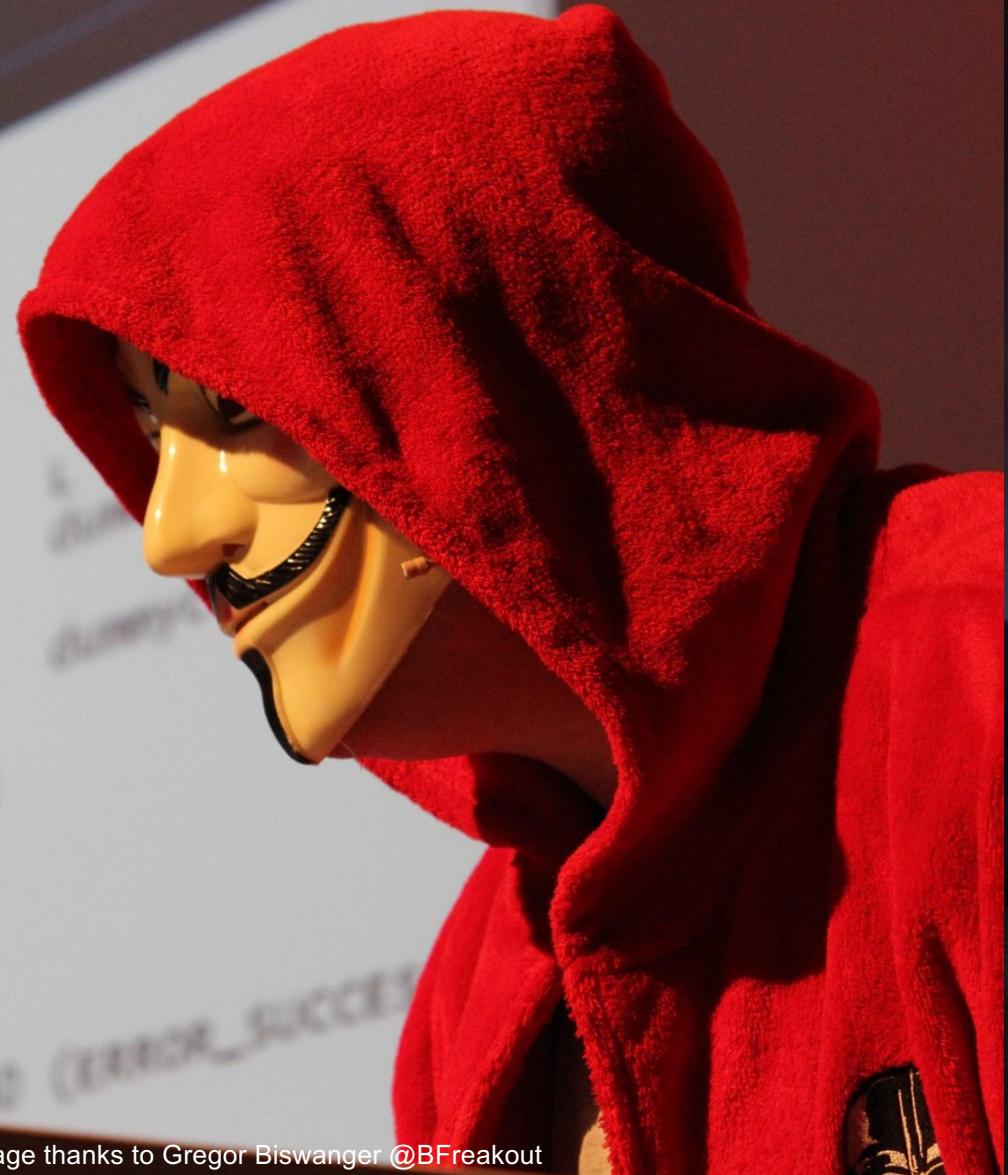
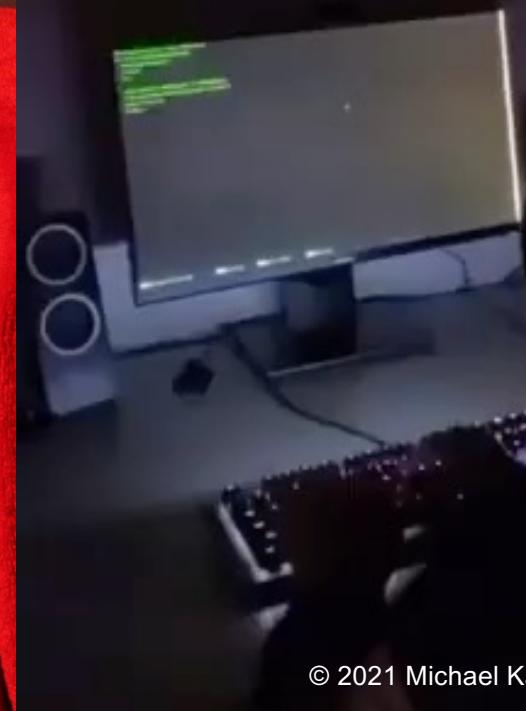


Image thanks to Gregor Biswanger @BFreakout

## Hackers in movies vs



© 2021 Michael Kaufmann @mike\_kaufmann



## How people think they get hacked



## How they really get hacked

<p>If you had to marry your spouse where you met them, where would your wedding have been?</p>	<p>Your porn name , is your middle name , and the first car you had.</p>	<p>How far away do you live from the place you were born?</p>
<p><b>The car you passed your drivers test in was a _____</b></p>	<p>A 'porn name' that exposes your middle name AND a street you grew up on are TWO pieces of info that people need.</p>	<p><b>NAME A SONG</b></p>
<p>STOP. THINK. DO NOT SHARE INFO.</p>	<p>In what city or town was your first job?</p>	<p><b>THAT TAKES YOU BACK TO HIGH SCHOOL</b></p>

# Losses caused by cyber attacks reported to IC3



[https://www.ic3.gov/Media/PDF/AnnualReport/2023\\_IC3Report.pdf](https://www.ic3.gov/Media/PDF/AnnualReport/2023_IC3Report.pdf)

## Top 5 crime types:

- › Phishing
- › Non-Payment / Delivery
- › Data Breach
- › Identity Theft
- › Extortion

## Trends

- › Confidence fraud / Romance scams
- › Cryptocurrency
- › Ransomware
- › Tech support fraud

# Michael Kaufmann

Founder & Managing Director, Xebia Germany



@mike\_kaufmann



@wulfland



<https://writeabout.net>

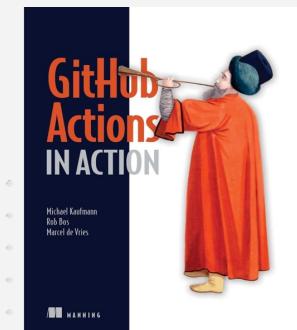
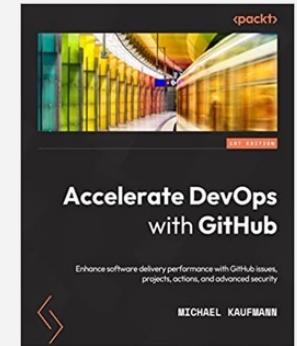


➤ 25 years software developer

➤ 15 years ALM & DevOps

➤ 10 years Git and GitHub

➤ Microsoft RD & MVP



# What is Cybersecurity?

---

*Cybersecurity is the protection of computer software, systems, networks, devices, and data from threats that can lead to unauthorized information disclosure, theft or damage, as well as from the disruption or misdirection of the services they provide.*



# Most Common Types of Cynersecurity Threats

- Phishing
- Malware
- Denial of Service (DoS) and Distributed Denial of Service (DDoS)
- SQL Injection
- Cross-Site Scripting (XSS)
- Social Engineering
- Zero-Day Exploits
- Credential attacks

# Phishing or Spear Phishing



Phishing involves sending deceptive emails or messages that appear to be from legitimate sources in order to trick recipients into revealing sensitive information, such as passwords, credit card numbers, or personal details.

Phishing can also lead victims to malicious websites or to download malware.

The screenshot shows two Microsoft Outlook email windows side-by-side, both from the same recipient, Michael Kaufmann, at oakassociates.com.

**Email 1 (Top):** Subject: Cgi [STAFF] shared "file" with you. It contains a warning about external links and attachments from an external sender. The message body says: "Message from Cgi server." It includes a link to a SharePoint document titled "Ho mes Limited." and a preview of the document.

**Email 2 (Bottom):** Subject: Urgent information about your April 2019 Deposit. It also contains a warning about external links and attachments. The message body says: "Dear Michael," followed by an apology for an internal error in the salary payment in April. It includes a link to a Payroll Accounting page and a note from the payroll team.

# Malware



**Ransomware:** Encrypts files and demands a ransom for decryption.



**Trojans:** Disguised as legitimate software, they give attackers unauthorized access.



**Viruses:** Self-replicating programs that attach to files and spread.



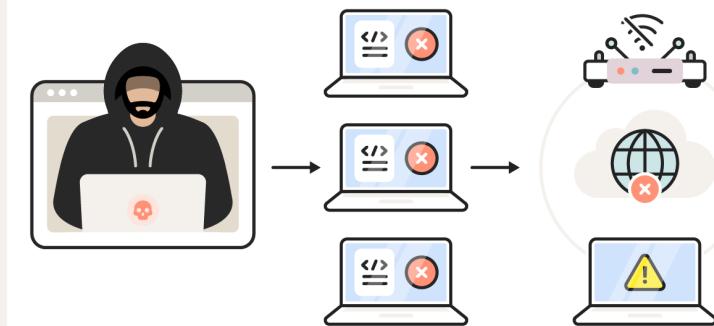
**Worms:** Self-replicating programs that spread through networks.

# Denial of Service (DoS) and Distributed DoS (DDoS)

DoS and DDoS attacks overload a target system, rendering it unavailable to users. DDoS attacks involve using a network of compromised devices to flood a target with traffic, making it difficult for the system to function properly or they may stop the system working entirely.

## DDoS Attacks Explained

DDoS attacks occur when a hacker uses a zombie network to flood a website/server with traffic or requests until it crashes.



### Attacker

A hacker infects devices to make botnets, forming a **zombie network**.

### Zombie Network

The zombie network **floods a targeted website or server** with traffic.

### Targeted Website/Server

The targeted **website or server crashes**, disconnecting from the internet.

# SQL Injection

```
txtUserId = getRequestString("UserId");
txtSQL = "SELECT * FROM Users WHERE UserId = " + txtUserId;
```

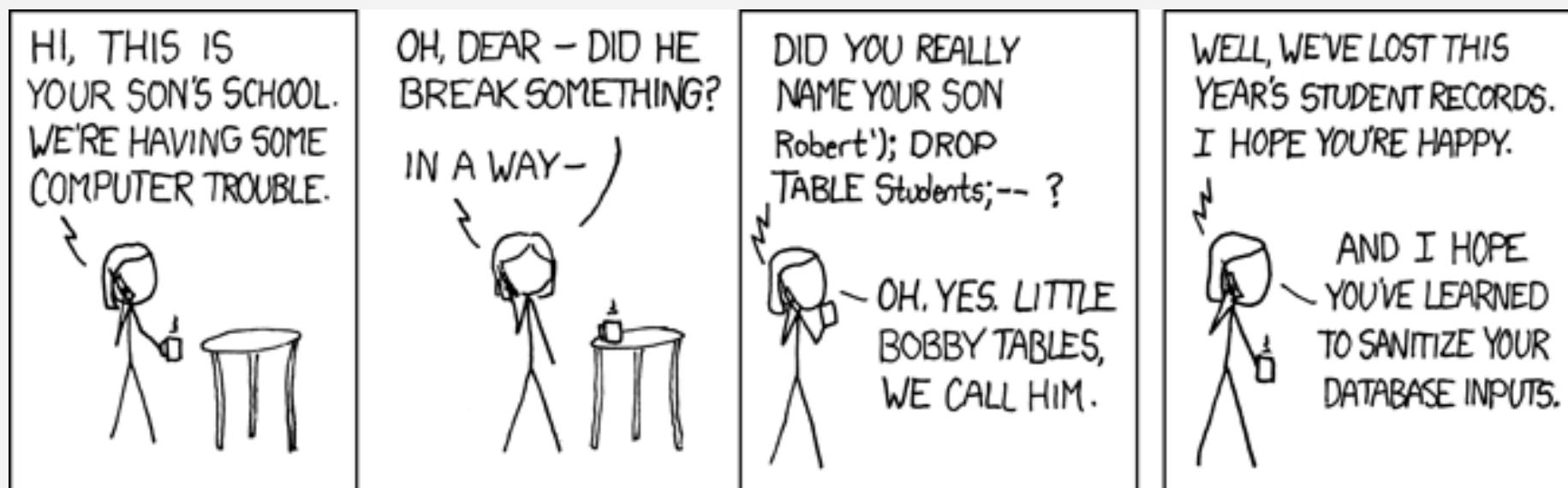
User ID:

```
11 module.exports = function searchProducts () {
12   return (req, res, next) => {
13     let criteria = req.query.q === 'undefined' ? '' : req.query.q || ''
14     criteria = (criteria.length <= 200) ? criteria : criteria.substring(0, 200)
15     models.sequelize.query(`SELECT * FROM Products WHERE ((name LIKE '%${criteria}%' OR description LIKE '%${criteria}%') AND deletedAt IS NULL) ORDER BY name`)
16     .then(([products]) => {
17       const dataString = JSON.stringify(products)
18       if (utils.notSolved(challenges.unionSqlInjectionChallenge)) { // vuln-code-snippet hide-start
19         let solved = true
20         models.User.findAll().then(data => {
21           const users = utils.queryResultToJson(data)
22           if (users.data?.length) {
23             for (let i = 0; i < users.data.length; i++) {
24               solved = solved && utils.containsOrEscaped(dataString, users.data[i].email) && utils.contains(dataString, users.data[i].password)
25               if (!solved) {
26                 break
27               }
28             }
29             if (solved) {
30               utils.solve(challenges.unionSqlInjectionChallenge)
31             }
32           }
33         })
34       }
35     })
36   }
37 }
```

# SQL Injection

```
txtUserId = getRequestId("UserId");
txtSQL = "SELECT * FROM Users WHERE UserId = " + txtUserId;
```

User ID: 105; DROP TABLE Suppliers



(Source: <https://xkcd.com/327/>)

# Cross-Site Scripting (XSS)

Attackers inject malicious scripts into web applications, which are then executed by unsuspecting users' browsers. This can lead to the theft of user data and/or the spreading of malware.

```
var Affix = function (element, options) {
  this.options = $.extend({}, Affix.DEFAULTS, options)

  this.$target = $(this.options.target)
    .on('scroll.bs.affix.data-api', $.proxy(this.checkPosition, this))
    .on('click.bs.affix.data-api', $.proxy(this.checkPositionWithEventLoop, this))

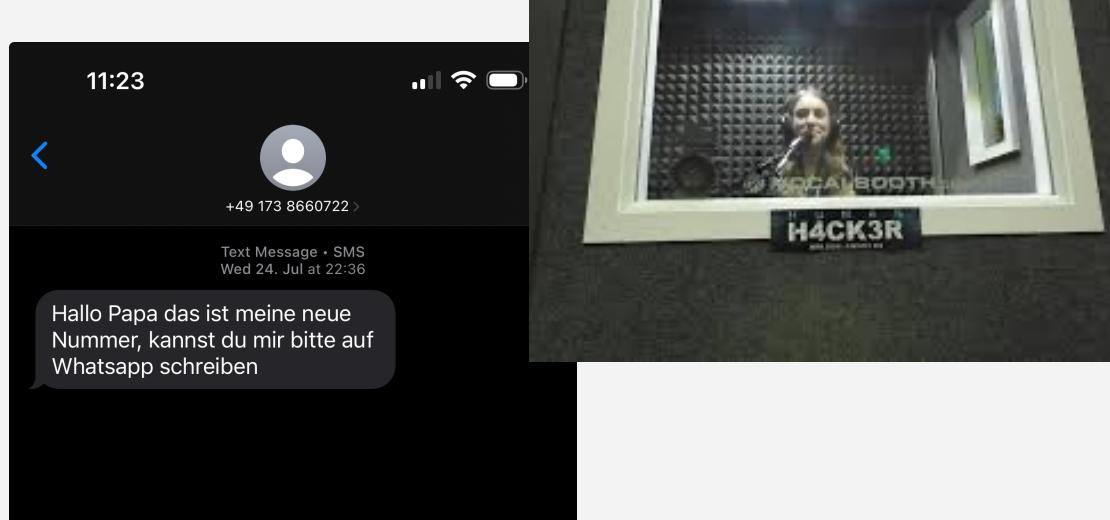
  this.$element      = $(element)
  this.affixed      = null
  this.unpin        = null
  this.pinnedOffset = null

  this.checkPosition()
}
```

```
143 // vuln-code-snippet start localXssChallenge xssBonusChallenge
144 filterTable () {
145   let queryParam: string = this.route.snapshot.queryParams.q
146   if (queryParam) {
147     queryParam = queryParam.trim()
148     this.ngZone.runOutsideAngular(() => { // vuln-code-snippet hide-start
149       this.io.socket().emit('verifyLocalXssChallenge', queryParam)
150     } // vuln-code-snippet hide-end
151     this.dataSource.filter = queryParam.toLowerCase()
152     this.searchValue = this.sanitizer.bypassSecurityTrustHtml(queryParam)
153     this.gridDataSource.subscribe((result: any) => {
154       if (result.length === 0) {
155         this.emptyState = true
156       } else {
157         this.emptyState = false
158       }
159     })
160   } else {
161     this.dataSource.filter = ''
162     this.searchValue = undefined
163     this.emptyState = false
164   }
165 }
```

# Social Engineering

Social engineering exploits human psychology to manipulate individuals into divulging confidential information or performing actions that compromise security. (OSINT)



## Social Engineering Tactics to Watch For

Knowing the red flags can help you avoid becoming a victim.



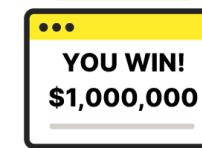
Your 'friend' sends you a strange message.



Your emotions are heightened.



The request is urgent.



The offer feels too good to be true.



You're receiving help you didn't ask for.



The sender can't prove their identity.

# Zero-Day (0day) Exploits

---

- These attacks target vulnerabilities in software or hardware that are not yet known to the vendor or public. Attackers take advantage of these vulnerabilities before patches are developed.



# Credential Attacks



**Brute force attacks:** attackers repeatedly guess passwords



**Credential stuffing attacks:** stolen credentials used on other systems



**Responder:** <https://github.com/Igandx/Responder>



**Pineapple:** wireless auditing platform



ATT&CK Matrix for Enterprise

layout: side ▾ show sub-techniques hide sub-techniques

Reconnaissance 10 techniques	Resource Development 8 techniques	Initial Access 10 techniques	Execution 14 techniques	Persistence 20 techniques	Privilege Escalation 14 techniques	Defense Evasion 43 techniques	Credential Access 17 techniques	Discovery 32 techniques	Lateral Movement 9 techniques	Collection 17 techniques	Command and Control 18 techniques	Efiltration 9 techniques	Impact 14 techniques
Active Scanning (2)	Acquire Access	Content Injection	Cloud Administration Command	Account Manipulation (8)	Abuse Elevation Control Mechanism (8)	Adversary-in-the-Middle (2)	Account Discovery (6)	Exploitation of Remote Services	Adversary-in-the-Middle (2)	Application Layer Protocol (4)	Automated Exfiltration (2)	Account Access Removal	
Gather Victim Host Information (2)	Acquire Infrastructure (2)	Drive-by Compromise	Command and Scripting Interpreter (12)	BITS Jobs	Access Token Manipulation (5)	Brute Force (4)	Application Window Discovery	Internal Spearphishing	Communication Through Removable Media	Communication Through Network (2)	Data Transfer Size Limits		
Gather Victim Identity Information (2)	Compromise Accounts (2)	Exploit Public-Facing Application	Container Administration Command	Account Manipulation (6)	BITS Jobs	Credentials from Password Stores (6)	Browser Information Discovery	External Tool Transfer	Exfiltration Over Alternative Protocol (2)	Exfiltration Over IP (2)	Data Encrypted for Impact		
Gather Victim Network Information (2)	Compromise Infrastructure (2)	External Remote Services	Deploy Container	Build Image on Host	Debugger Evasion	Exploitation for Credential Access	Cloud Infrastructure Discovery	File and Directory Discovery	File and Registry Discovery	File and Registry Discovery	Data Manipulation (2)		
Gather Victim Org Information (4)	Develop Capabilities (4)	Hardware Additions	Exploitation for Client Execution	Debugger Evasion	Forge Web Credentials (2)	Forced Authentication	Cloud Service Dashboard	File and Directory Discovery	File and Registry Discovery	File and Registry Discovery	Defacement (2)		
Phishing for Information (4)	Phishing (4)	Inter-Process Communication (2)	Compromise Host Software Binary	Create or Modify System Process (2)	Input Capture (4)	Forge Web Credentials (2)	Cloud Service Discovery	File and Registry Discovery	File and Registry Discovery	File and Registry Discovery	Data Encryption (2)		
Search Closed Sources (2)	Replication Through Removable Media	Native API	Create Account (2)	Domain or Tenant Policy Modification (2)	Input Capture (4)	Get Application Access Token	Cloud Storage Object Discovery	File and Registry Discovery	File and Registry Discovery	File and Registry Discovery	Exfiltration Over C2 Channel (2)		
Search Open Technical Databases (2)	Obtain Capabilities (2)	Obtain Capabilities (2)	Create or Modify System Process (2)	Event Triggered Execution (2)	Input Capture (4)	Modify Authentication Process (2)	Container and Resource Discovery	Clipboard Data	File and Registry Discovery	File and Registry Discovery	Exfiltration Over Other Network (2)		
Search Open Websites/Domains (2)	Supply Chain Compromise (2)	Scheduled Task/Job (2)	Domain or Tenant Policy Modification (2)	External Remote Services	Input Capture (4)	Multi-Factor Authentication Interception	Debugger Evasion	Cloud Service Discovery	Cloud Service Discovery	Cloud Service Discovery	Financial Theft (2)		
Search Victim-Owned Websites	Stage Capabilities (2)	Serverless Execution	Event Triggered Execution (2)	Event Triggered Execution (14)	Execution Guardrails (1)	Multi-Factor Authentication Request Generation	Device Driver Discovery	Cloud Service Discovery	Cloud Service Discovery	Cloud Service Discovery	Endpoint Denial of Service (4)		
	Trusted Relationship	Valid Accounts (4)	External Remote Services	Exploitation for Privilege Escalation	Exploitation for Defense Evasion	Network Sniffing	Domain Trust Discovery	Cloud Service Discovery	Cloud Service Discovery	Cloud Service Discovery	Firmware Corruption		
	Valid Accounts (4)	Shared Modules	Hijack Execution Flow (12)	Hijack Execution Flow (12)	File and Directory Permissions Modification (2)	OS Credential Dumping (2)	File and Directory Discovery	Container and Resource Discovery	Container and Resource Discovery	Container and Resource Discovery	Inhibit System Recovery		
	Software Deployment Tools	System Services (2)	Implant Internal Image	Process Injection (12)	Hide Artifacts (2)	Steal Application Access Token	File and Directory Discovery	Debugger Evasion	Clipboard Data	Clipboard Data	Network Denial of Service (2)		
	User Execution (2)	User Execution (2)	Modify Authentication Process (2)	Scheduled Task/Job (2)	Hijack Execution Flow (12)	Steal or Forge Authentication Certificates	File and Registry Discovery	Device Driver Discovery	Cloud Service Discovery	Cloud Service Discovery	Resource Hijacking		
	Windows Management Instrumentation	Office Application Startup (2)	Office Application Startup (2)	Valid Accounts (4)	Impair Defenses (11)	Steal or Forge Kerberos Tickets (4)	File and Registry Discovery	Domain Trust Discovery	Cloud Service Discovery	Cloud Service Discovery	Service Stop		
		Power Settings	Pre-OS Boot (5)	Indicator Removal (2)	Impersonation	Steal Web Session Cookie	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery	System Shutdown/Reboot		
		Scheduled Task/Job (5)	Scheduled Task/Job (5)	Indirect Command Execution	Masquerading (2)	Unsecured Credentials (6)	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
		Server Software Component (2)	Traffic Signaling (2)	Modify Authentication Process (2)	Modify Cloud Compute Infrastructure (2)	Peripheral Device Discovery	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Modify Registry	Modify Registry	Permission Groups Discovery (2)	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Modify System Image (2)	Modify System Image (2)	Process Discovery	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Network Boundary Bridging (1)	Network Boundary Bridging (1)	Query Registry	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Obfuscated Files or Information (13)	Obfuscated Files or Information (13)	Remote System Discovery	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Plist File Modification	Plist File Modification	System Information Discovery	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Pre-OS Boot (2)	Pre-OS Boot (2)	System Location Discovery (1)	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Process Injection (12)	Reflective Code Loading	System Network Configuration Discovery (2)	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Rogue Domain Controller	Rogue Domain Controller	System Network Connections Discovery	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Rootkit	Rootkit	System Owner/User Discovery	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Subvert Control Systems (2)	Subvert Control Systems (2)	System Service Discovery	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				System Binary Proxy Execution (14)	System Binary Proxy Execution (14)	System Time Discovery	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				System Script Proxy Execution (2)	System Script Proxy Execution (2)	Virtualization/Sandbox Evasion (2)	File and Registry Discovery	File and Registry Discovery	Cloud Service Discovery	Cloud Service Discovery			
				Template Injection	Traffic Signaling (2)								
				Unused/Unsupported Cloud Regions	Unused/Unsupported Cloud Regions								
				Use Alternate Authentication Material (4)	Use Alternate Authentication Material (4)								
				Valid Accounts (4)	Valid Accounts (4)								
				Virtualization/Sandbox Evasion (2)	Virtualization/Sandbox Evasion (2)								

# The MITRE ATT&CK framework

ATT&CK: Adversarial Tactics, Techniques, and Common Knowledge: <https://attack.mitre.org/>

- Understand Adversarial Behavior
- Plan and Implement Defense Strategies
- Incident Response and Threat Hunting

# Keep up to date:

---

Open Web Application Security Project (OWASP) top 10 vulnerabilities: <https://owasp.org/Top10/>

---

Common Vulnerabilities and Exposures (CVEs): <https://cve.mitre.org/>

---

Microsoft Security Response Center blogs: <https://msrc.microsoft.com/blog/>

---

National Institute of Standards and Technology (NIST): <https://www.dhs.gov/topics/cybersecurity>

---

Cybersecurity and Infrastructure Security Agency (CISA): <https://www.cisa.gov/resources-tools/resources/free-cybersecurity-services-and-tools>

---

National Cybersecurity Center of Excellence (NCCoE): <https://www.dhs.gov/topics/cybersecurity>

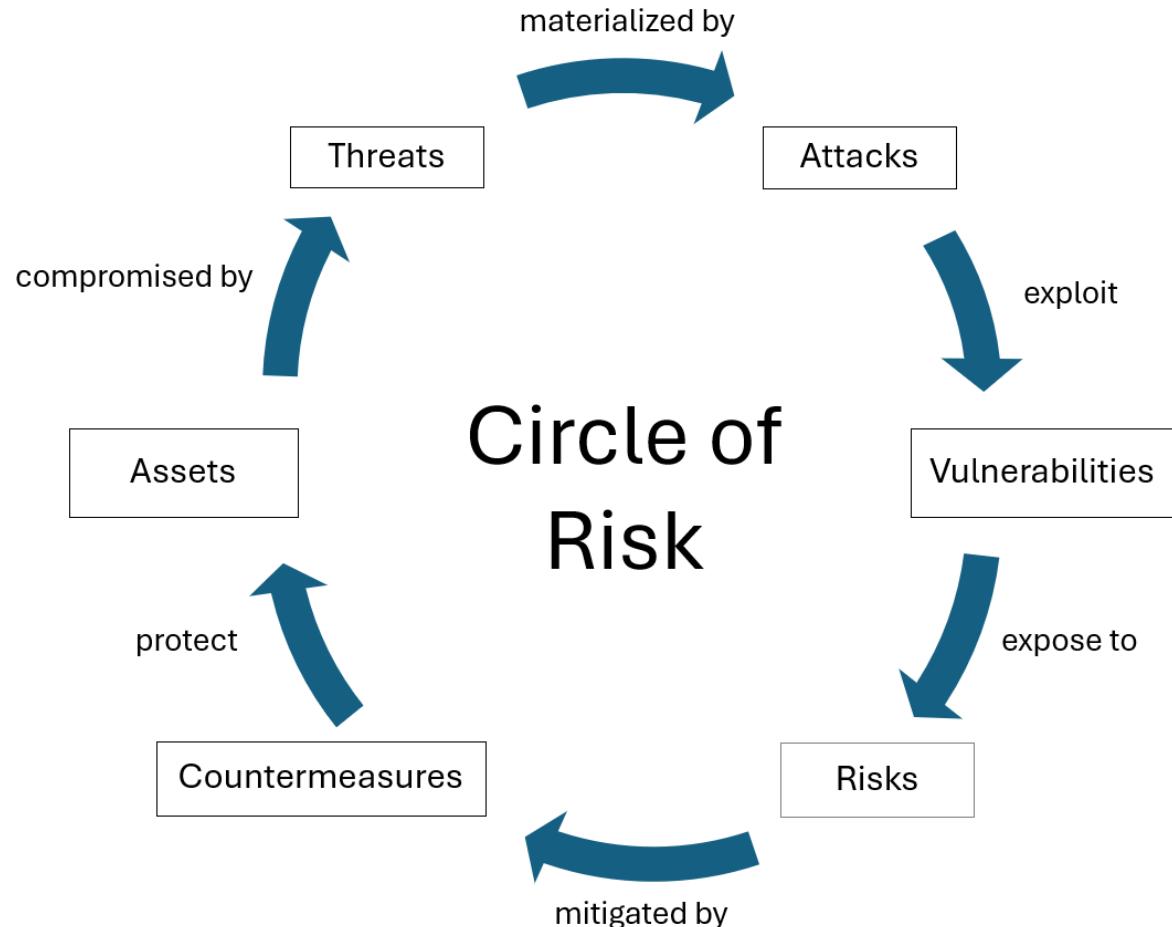
---

United States Computer Emergency Readiness Team (US-CERT): <https://www.cisa.gov/resources-tools/resources/free-cybersecurity-services-and-tools>

# Risk Management

## Risk Assessment:

1. Identify Assets and Threats
2. Assess Vulnerabilities
3. Likelihood Assessment
4. Impact Assessment
5. Risk Calculation
6. Prioritization and Decision-Making
7. Risk Treatment
8. Continuous Monitoring and Review



# Security Practices



# Assume breach

---

*„Fundamentally, if somebody wants to get in, they're getting in. Alright, good. Accept that.“*

- Michael V. Hayden  
former General of the US Air Force and  
former Director of NSA and CIA

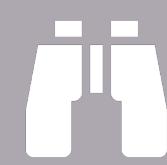


# From Prevent Breach to Assume Breach



## Prevent Breach

- Security layers (Firewall, VPN, DMZ)
- Risk analysis
- Threat modeling
- Architecture reviews
- Code security reviews
- Security testing



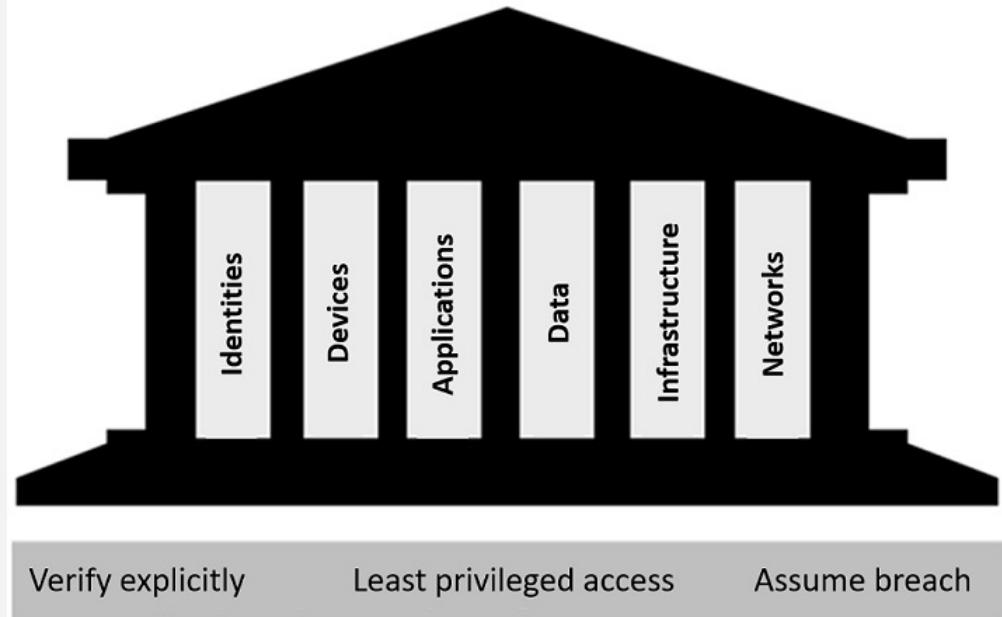
## Assume Breach

- Red-Team | Blue-Team exercises (war games)
- Central security monitoring (SIEM)
- Penetration testing in production
- Detection of anomalies

# Zero Trust Principles

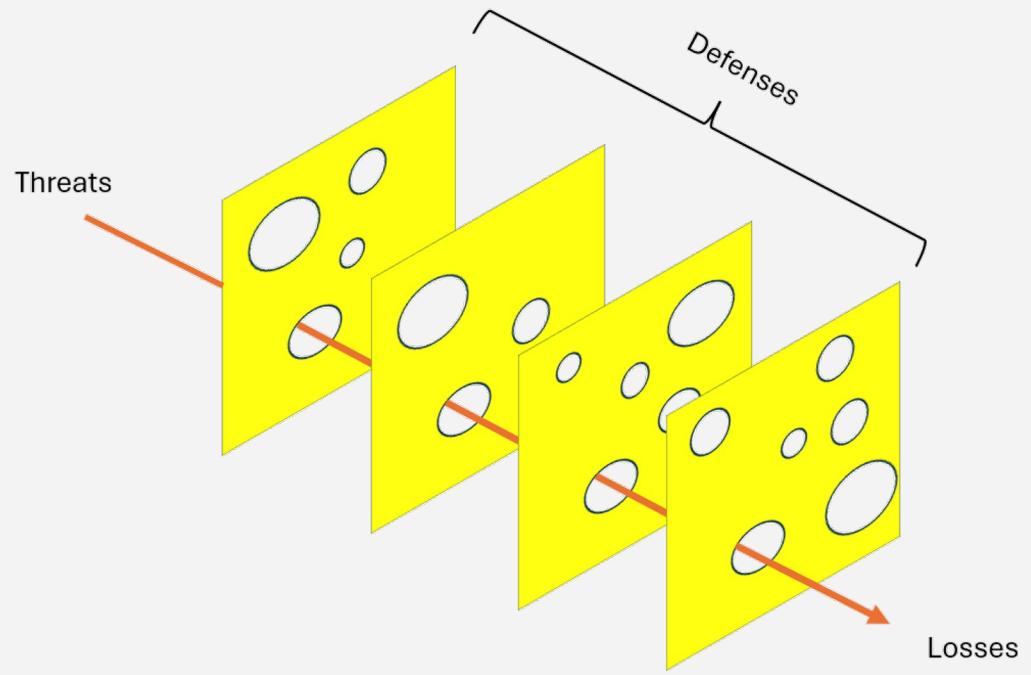
- › **Verify Identity:**  
Authentication and authorization are rigorously applied to all users and devices, regardless of their location.
- › **Least Privilege:**  
Users and devices are granted the minimum level of access necessary to perform their tasks.
- › **Micro-Segmentation:**  
Network resources are divided into smaller segments to limit lateral movement within the network in case of a breach.
- › **Continuous Monitoring:**  
Ongoing monitoring and analysis of user and device behavior. Detection of anomalies and potential threats.
- › **Data Encryption:**  
Data is encrypted both in transit and at rest to prevent unauthorized access.
- › **Strict Access Control:**  
Access controls are enforced based on context, such as user roles, device health, and network location.

**Zero Trust Methodology**  
“Trust no one, verify everything”



# Differences from Traditional Security Architectures

- › **Perimeter vs. Identity-Centric:**  
Traditional models focus on securing the perimeter assuming that internal users and devices can be trusted. Zero Trust assumes that threats can originate from both inside and outside the network.
- › **Implicit vs. Explicit Trust:**  
Traditional models implicitly trust devices and users within the network until proven otherwise. Zero Trust explicitly verifies identities and continuously monitors for anomalies.
- › **Flat vs. Segmented Network:**  
Traditional architectures often involve flat networks where internal users have broad access. Zero Trust advocates segmenting the network into smaller, isolated zones to contain potential breaches.
- › **Reactive vs. Proactive:**  
Traditional security often relies on reactive measures such as perimeter firewalls and intrusion detection. Zero Trust takes a proactive approach by assuming breaches are likely and minimizing their impact.





# Identity and Access Management (IAM)

A set of processes, technologies, and policies that are implemented to ensure that the right individuals have the appropriate access to resources within an organization's digital environment.

Key concepts:

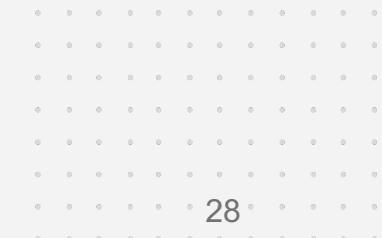
- › **Least privilege**
- › **Segregation of duties**



# Identity and Access Management (IAM)

Authentication and Multi-Factor Authentication (MFA or 2FA):

- › **Something you know:** This includes passwords, PINs, or another secret knowledge that only the authorized user should possess.
- › **Something you have:** This involves physical tokens or devices like smart cards, security tokens, or mobile phones used to confirm the user's identity.
- › **Something you are:** This refers to biometric factors like fingerprints, facial recognition, or retinal scans that are unique to an individual.





# Identity and Access Management (IAM)

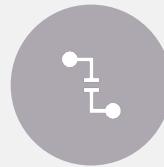
## **Authorization:**

- Authorization is the process of granting or denying specific permissions and privileges to authenticated users or. It determines what actions or operations a user is allowed to perform within a system or on specific resources.
- Authorization can be thought of as answering the question, "What can an authenticated user do?" It involves defining and enforcing access control policies to protect sensitive data and resources from unauthorized access or modification.

# Other IAM Capabilities



**Single Sign-On (SSO):** Accessing multiple applications and systems with a single set of credentials



**Role-Based Access Control (RBAC):** Assigning permissions based on predefined roles



**Adaptive Authentication:** Assessing risk factors such as location, device, time of access, and user behavior to dynamically adjust the level of authentication required



**Biometric Authentication:** Using unique biological characteristics such as fingerprints, facial features, voice patterns



**Privileged Access Management (PAM):** Securing access to critical systems and data by enforcing strict controls on privileged accounts (just-in-time access)



**Behavioral Analytics:** Monitoring user behavior and establishing baseline patterns. Deviations can trigger alerts.

# Network Security

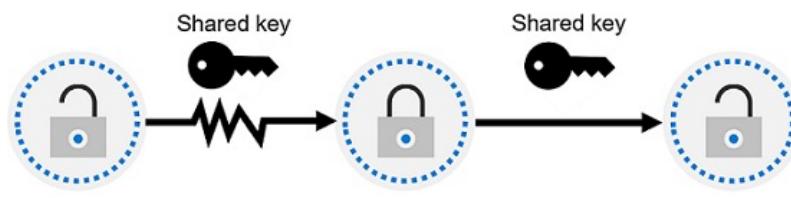
---

- Traditional firewalls
- Web application firewalls
- Cloud security groups
- CDN
- Load balancers
- Bastion hosts
- VPNs
- DDoS protection

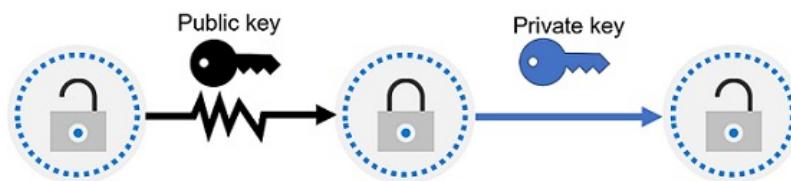
	Layer		Function
Host layers	7	Application	High-level protocols such as for resource sharing or remote file access, e.g. HTTP.
	6	Presentation	Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption
	5	Session	Managing communication sessions, i.e., continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
	4	Transport	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing
Media layers	3	Network	Structuring and managing a multi-node network, including addressing, routing and traffic control
	2	Data link	Transmission of data frames between two nodes connected by a physical layer
	1	Physical	Transmission and reception of raw bit streams over a physical medium

# Encryption and hashing

Symmetric Encryption



Asymmetric Encryption



Original  
Text

Hashing  
Algorithm

101010  
010101  
101010

# Security Operations

## **Security Operations Center (SOC):**

A centralized team responsible for 24/7 monitoring, analysis, and response to security events.

## **Incident Response Team:**

A specialized team focused on responding to security incidents and breaches. They conduct investigations, coordinate response efforts, and facilitate recovery.

## **Threat Hunting Team:**

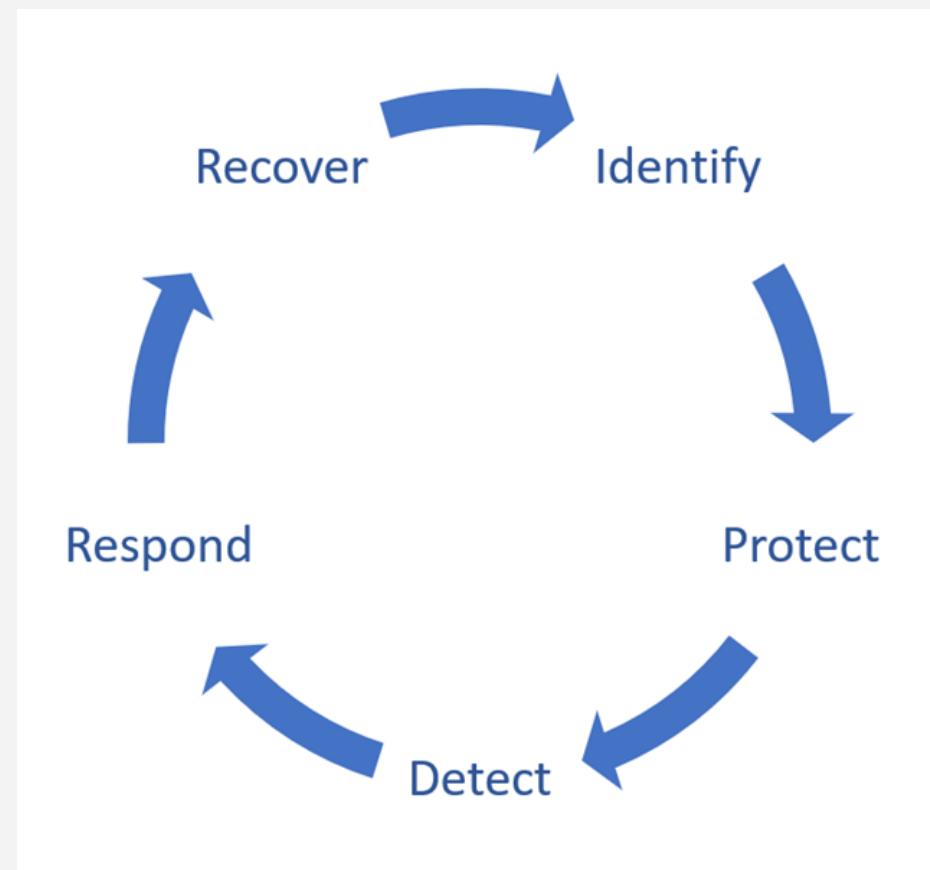
A team that proactively searches for signs of advanced threats and hidden vulnerabilities that may not be detected by traditional security tools.

## **Red Team/Blue Team:**

The red team simulates attacks to identify vulnerabilities, while the blue team defends against those attacks. Both teams work together to improve security measures.

## **Managed Security Services Provider (MSSP):**

Some organizations outsource their security operations to third-party providers specializing in security monitoring and incident response.





# Security Information and Event Management (SIEM)

---

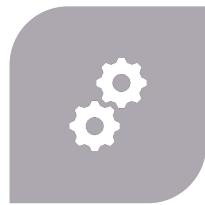
- **Log Collection:** Collect logs and security event data.
- **Data Normalization:** Normalize log data into a common format.
- **Event Correlation:** Correlate events to identify patterns and anomalies.
- **Alerting and Notification:** Generate alerts and notifications in real time.
- **Incident Detection:** Detection of security incidents, including unauthorized access, data breaches, malware infections, and insider threats.
- **User and Entity Behavior Analytics (UEBA):** Identify abnormal user and entity behaviors that may indicate compromised accounts or insider threats.
- **Threat Intelligence Integration:** Integrate with threat intelligence feeds to enhance threat detection by comparing known indicators of compromise (IOCs) with network activity.
- **Automation and Orchestration:** Automate responses to common security incidents, reducing response times and manual effort.
- **Dashboard and Visualization:** Allow monitoring security data and creating custom reports.
- **Integration with Other Security Tools:** Integrate with other security tools and technologies, such as endpoint detection and response (EDR) solutions.

# Application Security (AppSec)

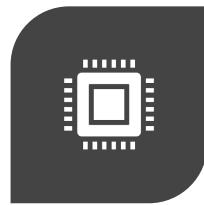
Application security refers to the practice of protecting software applications from security threats, vulnerabilities, and attacks. It encompasses the processes, techniques, and tools used to identify, mitigate, and prevent security risks throughout the development, deployment, and maintenance lifecycle of an application.



**SECURE BY DESIGN:** SECURITY SHOULD BE INTEGRATED INTO THE APPLICATION'S DESIGN AND ARCHITECTURE.



**INPUT VALIDATION:** ALL USER INPUTS SHOULD BE VALIDATED.



**OUTPUT ENCODING:** DATA SENT TO THE CLIENT SHOULD BE PROPERLY ENCODED TO PREVENT VULNERABILITIES LIKE CROSS-SITE SCRIPTING (XSS).



**AUTHENTICATION AND AUTHORIZATION:** AUTHENTICATE USERS AND AUTHORIZE THEIR ACCESS TO RESOURCES BASED ON THEIR ROLES AND PERMISSIONS.



**DATA PROTECTION:** SENSITIVE DATA SHOULD BE ENCRYPTED WHEN STORED, TRANSMITTED, AND PROCESSED TO PREVENT UNAUTHORIZED ACCESS.

# Application Security (AppSec)

Application security refers to the practice of protecting software applications from security threats, vulnerabilities, and attacks. It encompasses the processes, techniques, and tools used to identify, mitigate, and prevent security risks throughout the development, deployment, and maintenance lifecycle of an application.



**SESSION MANAGEMENT:**  
SECURE SESSION MANAGEMENT  
ENSURES USER SESSIONS ARE  
PROTECTED FROM HIJACKING  
AND UNAUTHORIZED ACCESS.



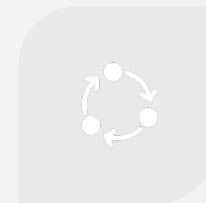
**SECURE DEPENDENCIES:** KEEP  
ALL SOFTWARE DEPENDENCIES  
UP TO DATE WITH SECURITY  
PATCHES TO PREVENT  
VULNERABILITIES.



**ERROR HANDLING AND LOGGING:**  
IMPLEMENT SECURE ERROR  
HANDLING TO AVOID REVEALING  
SENSITIVE INFORMATION AND  
ENSURE SECURE LOGGING  
PRACTICES.



**SECURITY TESTING:** REGULARLY  
TEST APPLICATIONS FOR  
VULNERABILITIES USING  
METHODS LIKE PENETRATION  
TESTING, CODE REVIEWS, AND  
AUTOMATED SCANNING TOOLS.



**SECURE SOFTWARE  
DEVELOPMENT LIFECYCLE  
(SDLC):** INTEGRATE SECURITY  
PRACTICES INTO EVERY PHASE  
OF THE SOFTWARE  
DEVELOPMENT LIFECYCLE.

# AppSec Key Capabilities

# Software Composition Analysis (SCA) a.k.a. Dependency Scanning



GitHub (Dependency-Graph/Dependabot)



anchore (<https://anchore.com/>)



Dependency-Track  
(<https://dependencytrack.org/>)

The screenshot shows a list of Dependabot alerts. There are 4 open alerts and 0 closed ones. The alerts are for the following packages:

- marsdb (critical severity)
- express-jwt (high severity)
- sanitize-html (moderate severity)
- jsonwebtoken (critical severity)

Each alert includes a timestamp (3 minutes ago by GitHub), a package.json link, and a severity indicator.

The screenshot shows a dependency graph interface. It displays a list of dependencies defined in package.json, with 138 entries. Several vulnerabilities are highlighted with yellow boxes and icons:

- auth0 / express-jwt Known security vulnerability in 0.1.3
- auth0 / node-jsonwebtoken Known security vulnerability in 0.4.0
- c58 / marsdb Known security vulnerability in ^ 0.6.11
- apostrophecms / sanitize-html Known security vulnerability in 1.4.2
- istanbuljs / istanbuljs @istanbuljs/nyc-config-typescript ^ 1.0.1
- Seally / types-chai @types/chai ^ 4.2.14
- DefinitelyTyped / DefinitelyTyped @types/chai-as-promised ^ 7.1.3

The interface also includes tabs for Dependencies, Dependents, and Dependabot, and a message about potential security vulnerabilities.

# Static Application Security Testing (SAST)

Analyze source code, bytecode, or binary code to identify security vulnerabilities in the application's codebase.



## Whitebox-Testing

- › GitHub Code Analysis
- › SonarQube, Fortify, Checkmarx, and Veracode
- › Semgrep (<https://semgrep.dev/>)
- › Mobile-Security-Framework (MobSF) (<https://github.com/MobSF/Mobile-Security-Framework-MobSF>)

Client-side cross-site scripting

Writing user input directly to the DOM allows for a cross-site scripting vulnerability.

Open Error CWE-79 CWE-116 security

Branch: main ▾ Dismiss ▾

frontend/src/app/search-result/search-result.component.ts

```
149     this.io.socket().emit('verifyLocalXssChallenge', queryParam)
150 } // vuln-code-snippet hide-end
this.dataSource.filter = queryParam.toLowerCase()
152 this.searchValue = this.sanitizer.bypassSecurityTrustHtml(queryParam) // vuln-code-snippet vuln-line localXssChallenge
```

Cross-Site scripting vulnerability due to `user-provided` value.

CodeQL Show paths

```
153 this.gridDataSource.subscribe((result: any) => {
154   if (result.length === 0) {
155     this.emptyState = true
156   }
157 })
```

Tool Rule ID Query  
CodeQL js/xss View source

Directly writing user input (for example, a URL query parameter) to a webpage without properly sanitizing the input first, allows for a cross-site scripting vulnerability.

Show more ▾

First detected in commit 9474e68 3 days ago

Create codeql-analysis.yml Verified × 9474e68

frontend/src/app/search-result/search-result.component.ts#L152 on branch main



Tool Rule ID Query  
CodeQL js/xss View source

Directly writing user input (for example, a URL query parameter) to a webpage without properly sanitizing the input first, allows for a cross-site scripting vulnerability.

This kind of vulnerability is also called *DOM-based cross-site scripting*, to distinguish it from other types of cross-site scripting.

**Recommendation**

To guard against cross-site scripting, consider using contextual output encoding/escaping before writing user input to the page, or one of the other solutions that are mentioned in the references.

**Example**

The following example shows part of the page URL being written directly to the document, leaving the website vulnerable to cross-site scripting.

```
function setLanguageOptions() {
  var href = document.location.href,
    deflt = href.substring(href.indexOf("default") + 8);
  document.write("<OPTION value=" + deflt + ">" + deflt + "</OPTION>");
  document.write("<OPTION value=2>English</OPTION>");
}
```

**References**

- OWASP: DOM based XSS Prevention Cheat Sheet.
- OWASP: XSS (Cross Site Scripting) Prevention Cheat Sheet.
- OWASP DOM Based XSS.
- OWASP Types of Cross-Site Scripting.
- Wikipedia: Cross-site scripting.
- Common Weakness Enumeration: [CWE-79](#).
- Common Weakness Enumeration: [CWE-116](#).

Show less ▾

# Dynamic Application Security Testing (DAST)

Scans a running application to identify vulnerabilities by sending input requests and analyzing responses.



## Blackbox-Testing

- › OWASP ZAP ( Zed Attack Proxy, <https://owasp.org/www-project-zap> )
- › Burp Suite von PortSwigger ( <https://portswigger.net/burp> )

The screenshot shows the OWASP ZAP interface in Standard Mode. The top navigation bar includes 'Standard Mode', 'Sites', 'Contexts', 'HUB Contexts', and 'Sites'. Below this is a tree view of 'Sites' containing various URLs. The main pane displays a 'Text' tab with a JSON alert payload. At the bottom, there's a 'History' tab showing a list of network requests and responses, and a 'WebSockets' tab.

The screenshot illustrates the OWASP ZAP process. A browser window shows a 'Not Secure' warning for 'https://xyz-demo-shop.azurewebsites.net'. The ZAP interface is overlaid, with the title 'Tailwind Traders'. The 'Sites Tree' and 'Sites' tabs are visible. A tooltip indicates 'OWASP ZAP intercepts the traffic'. The main content area shows a shopping cart page with shelves of products. A 'Spider' tool panel on the right has several targets listed. A tooltip says 'Head-Up-Display (HUB) to analyze and attack the site using the spider'. Another tooltip says 'Items can be customized'. A 'WebSockets' tab is also present at the bottom left.

# Dynamic Application Security Testing (DAST)



## Blackbox-Testing

- › OWASP ZAP ( Zed Attack Proxy, <https://owasp.org/www-project-zap> )
- › Burp Suite von PortSwigger ( <https://portswigger.net/burp> )

The screenshot shows a GitHub Actions workflow named "OWASP ZAP". It includes three actions: "OWASP ZAP Full Scan", "OWASP ZAP API Scan", and "OWASP ZAP Baseline Scan". The "OWASP ZAP Full Scan" action uses a GitHub token to create issues in the repository. The "OWASP ZAP API Scan" and "OWASP ZAP Baseline Scan" actions also use GitHub tokens.

```
jobs:
  owasp:
    name: OWASP Full Scan
    runs-on: ubuntu-latest
    steps:
      - name: OWASP ZAP Full Scan
        uses: zaproxy/action-full-scan@v0.2.0
        with:
          # GitHub Token to create issues in the repository
          token: ${{ github.token }}
          target: https://target
```

The screenshot shows a ZAP Scanning Report for sites xyz-demo-shop.azurewebsites.net. It displays a summary of alerts with 3 Medium, 18 High, and 20 Low risk instances. A GitHub issue #173 is created with a link to the workflow run, containing a list of alerts including Content Security Policy (CSP) Header Not Set, Missing Anti-clickjacking Header, and Proxy Disclosure.

ZAP Scanning Report

Sites: http://xyz-demo-shop.azurewebsites.net https://xyz-demo-shop.azurewebsites.net

Generated on Sun, 9 Jan 2022 19:16:45

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	3
Low	8
Informational	3
False Positives:	0

Alerts

Name	Risk Level	Number of Instances
Content Security Policy (CSP) Header Not Set	Medium	3
Missing Anti-clickjacking Header	Medium	3
Proxy Disclosure	Medium	18
Cookie with SameSite Attribute None	Low	2
Cookie without SameSite Attribute	Low	2
HTTP/2 Content Available via HTTP	Low	11
Incomplete or No Cache-control Header Set	Low	5
Private IP Disclosure	Low	1
Strict-Transport-Security Header Not Set	Low	11
Timestamp Disclosure - Unix	Low	20
X-Content-Type-Options Header Missing	Low	11
Cookie Stack Detector	Informational	18
Informational - Suspicious Comments	Informational	2
Modern Web Application	Informational	4

A report is added as a build artifact in HTML, JSON, and Markdown

ZAP Full Scan Report #173

github-actions bot opened this issue 21 hours ago · 0 comments

github-actions bot commented 21 hours ago

Contributor

A GitHub Issue is created with a link to the workflow run

41

# Runtime Application Self-Protection (RASP)

Monitors and protects applications in real-time, detecting and responding to security threats as they occur.



## Blackbox-Testing

- › Veracode Runtime Protection
- › F5 Advanced WAF with RASP
- › Datadog Application Security Management

The dashboard displays a chart of 'Site Threats' over a 30-day period, showing a significant spike in attacks around May 9th. A pie chart indicates attack types: SQL Injection (blue), XSS - Cross-site Scripting (green), and others (grey). Below is an 'Event Log' table showing 356 events, mostly protected by Veracode, with details like date, event type, server, request path, and origin IP.

ID	Date	Event Type	Server	Request Path	Origin IP	Rows	User
76497784-640b-4c2	5/19/16 3:16 PM	Multiple Events	testapp-1-prot-verac...	/www/active-Reflected.XSS(R)	70.102.160.101	0	
7a7550a0-4007-420	5/19/16 3:03 PM	Multiple Events	testapp-1-prot-verac...	/www/active-Reflected.XSS(R)	70.102.160.101	0	
c9880050-7e10-430	5/19/16 2:52 PM	Multiple Events	testapp-1-prot-verac...	/www/active-Reflected.XSS(R)	70.102.160.101	0	
798d77aaf-7116-41f	5/19/16 2:51 PM	Multiple Events	testapp-1-prot-verac...	/testapp-master.SNAPSHOT/test	70.102.160.101	0	

The dashboard shows a critical SSRF vulnerability triggered on April 14, 2022, at 8:10:42 pm. It details the compromised URL (`http://localhost/`) and the reason (a user parameter controlled the domain part of the URL and directed it to a sensitive resource). A sample attack flow diagram illustrates the path from a compromised URL through various services like net/http, web-store-mongo, product-recommendation, auth-dotnet-httpClient, and auth-dotnet-postgres to the final target, auth-dotnet.

**WHAT HAPPENED**  
1 IP has successfully triggered a SSRF vulnerability

**VULNERABILITY TRIGGERED**  
Compromised URL: `http://localhost/`  
Reason: A user parameter controlled the domain part of the URL and directed it to a sensitive resource.

**SAMPLE ATTACK FLOW**

```
graph LR; A[9de9a03d-97efc53062] --> B[net/http]; B --> C[web-store-mongo]; C --> D[product-recommen...]; D --> E[auth-dotnet-httpClient]; E --> F[auth-dotnet-postgres]
```

**SUGGESTED NEXT STEPS**

- Block the attacker IP at the edge (WAF/CDN)
- Declare an incident if you consider that this signal needs to be escalated

**INSIGHTS**

# Container Scanning

## Container Vulnerability Analysis (CVA) / Container Security Analysis (CSA)

### Open source:

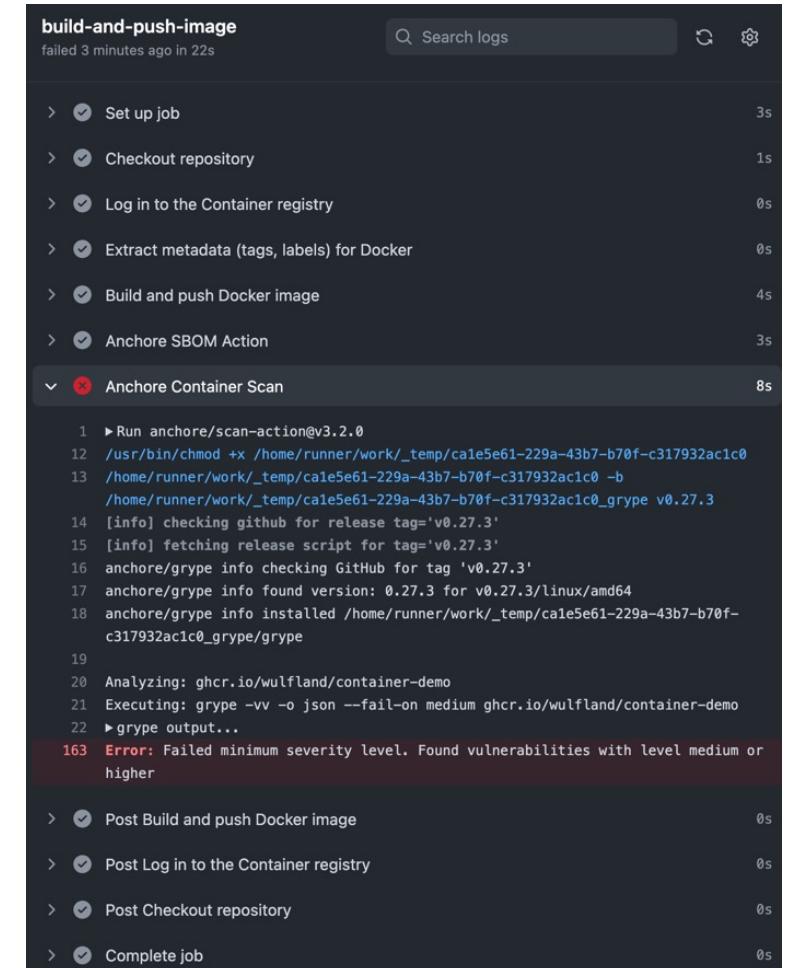
- › Anchore gryp  
<https://github.com/anchore/grype/>
- › Clair  
<https://quay.github.ioclair/>

### Commercial:

- › WhiteSource  
<https://www.whitesourcesoftware.com/solution-for-containers/>
- › Aqua  
<https://www.aquasec.com/products/container-security/>

```
- name: Anchore Container Scan
  uses: anchore/scan-action@v3.2.0
  with:
    image: ${{ env.REGISTRY }}/{{ env.IMAGE_NAME }}
    debug: true
```

<https://github.com/wulfland/container-demo/actions/runs/2179243137>



```
build-and-push-image
failed 3 minutes ago in 22s
Search logs
3s
1s
0s
0s
4s
3s
8s
3s
12 /usr/bin/chmod +x /home/runner/work/_temp/ca1e5e61-229a-43b7-b70f-c317932ac1c0
13 /home/runner/work/_temp/ca1e5e61-229a-43b7-b70f-c317932ac1c0 -b
14 [info] checking github for release tag='v0.27.3'
15 [info] fetching release script for tag='v0.27.3'
16 anchore/grype info checking GitHub for tag 'v0.27.3'
17 anchore/grype info found version: 0.27.3 for v0.27.3/linux/amd64
18 anchore/grype info installed /home/runner/work/_temp/ca1e5e61-229a-43b7-b70f-c317932ac1c0_grype/grype
19
20 Analyzing: ghcr.io/wulfland/container-demo
21 Executing: grype -vv -o json --fail-on medium ghcr.io/wulfland/container-demo
22 ► grype output...
163 Error: Failed minimum severity level. Found vulnerabilities with level medium or
higher
0s
0s
0s
0s
0s
0s
0s
```

# Infrastructure Scanning

## Infrastructure policies

### Open source:

- › Checkov  
<https://www.aquasec.com/products/container-security/>
- › OpenVAS

### Commercial:

- › Defender for Cloud  
<https://azure.microsoft.com/en-us/services/defender-for-cloud>
- › Azure Policy  
<https://docs.microsoft.com/de-de/azure/governance/policy/>

```
- name: Checkov GitHub Action
  uses: bridgecrewio/checkov-action@master
  with:
    directory: ch15_sec/
    output_format: sarif

- name: Upload SARIF file
  uses: github/codeql-action/upload-sarif@v1
  with:
    sarif_file: results.sarif
    if: always()
```

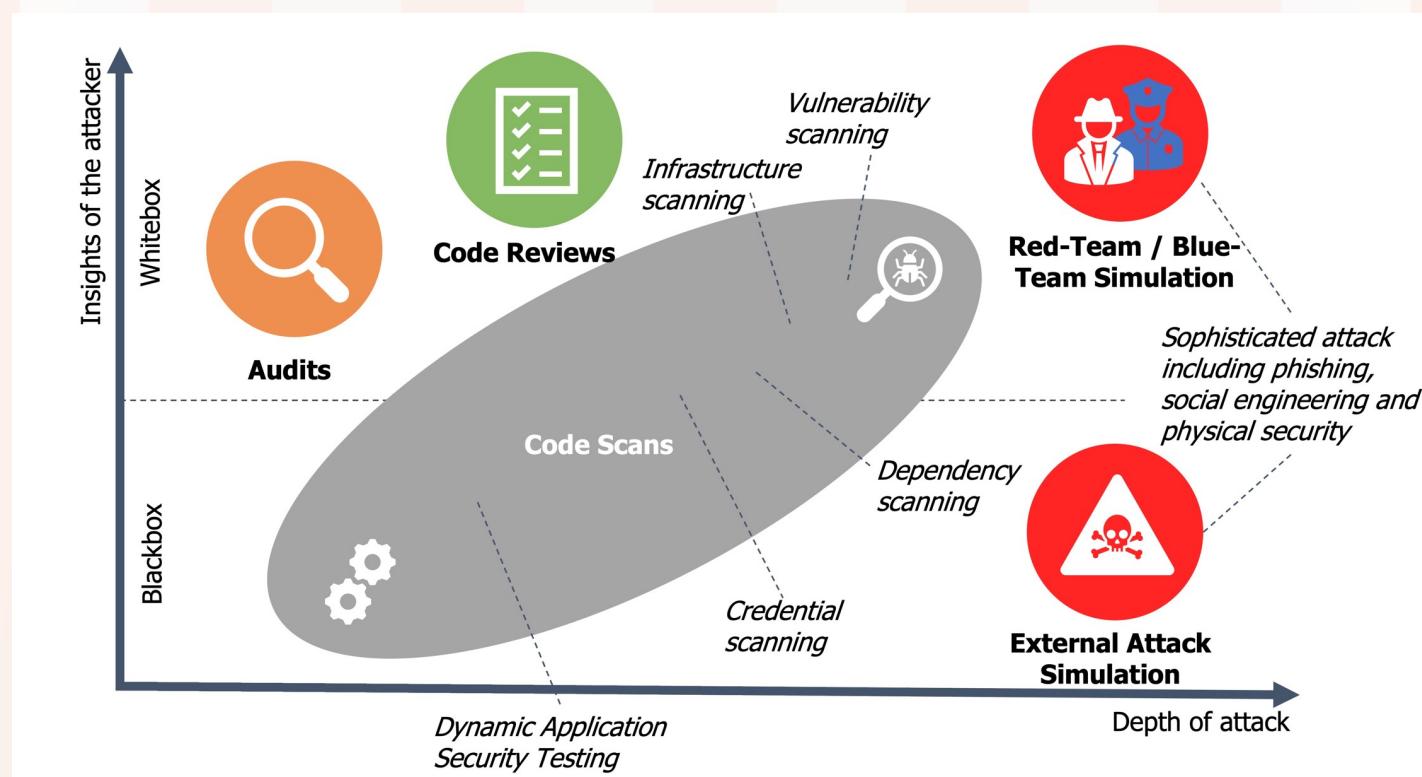
## Code scanning

Latest scan	Branch	Workflow	Lines scanned	Duration	Result
10 minutes ago	main	CodeQL	1.45k / 1.39k ⓘ	5m 26s	21 alerts

Filters  tool:checkov is:open branch:main  
 Clear current search, filters and sorts

✓ 2 Open	✗ 0 Closed	Tool	Rule	Branch	Severity	Sort
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Ensure that S3 bucket has a Public Access block	Error	aws.tf:1	Detected 15 minutes ago by checkov	main
<input type="checkbox"/>	<input type="checkbox"/>	Ensure that S3 bucket has cross-region replication enabled	Error	aws.tf:1	Detected 15 minutes ago by checkov	main

# Security Testing



# The event-stream incident



Social engineering attack



Supply chain attack:  
event-stream@3.3.6 -> flatmap-stream@0.1.1



Code execution in build process  
targeting copay



Harvest the user's bitcoin and  
private keys

**Malicious Package in flatmap-stream**  
Critical severity | GitHub Reviewed | Published on 1 Sep 2020 · Updated on 1 Oct 2021

Vulnerability details | Dependabot alerts 0

Package flatmap-stream ( npm ) Affected versions = 0.1.1

Description

Version 0.1.1 of flatmap-stream is considered malicious.

This module runs an encrypted payload targeting a very specific application, copay and because they shared the same description it would have likely worked for copay-dash.

The injected code:

- Read in AES encrypted data from a file disguised as a test fixture
- Grabbed the npm package description of the module that imported it, using an automatically set environment variable
- Used the package description as a key to decrypt a chunk of data pulled in from the disguised file

The decrypted data was part of a module, which was then compiled in memory and executed.

This module performed the following actions:

- Decrypted another chunk of data from the disguised file
- Concatenated a small, commented prefix from the first decrypted chunk to the end of the second decrypted chunk
- Performed minor decoding tasks to transform the concatenated block of code from invalid JS to valid JS (we believe this was done to evade detection by dynamic analysis tools)
- Wrote this processed block of JS out to a file stored in a dependency that would be packaged by the build scripts:

The chunk of code that was written out was the actual malicious code, intended to be run on devices owned by the end users of Copay.

This code would do the following:

- Detect the current environment: Mobile/Cordova/Electron
- Check the Bitcoin and Bitcoin Cash balances on the victim's copay account
- If the current balance was greater than 100 Bitcoin, or 1000 Bitcoin Cash:
  - Harvest the victim's account data in full
  - Harvest the victim's copay private keys
  - Send the victim's account data/private keys off to a collection

Follow

right9ctrl 東京都 Committed to this repository

GHSA ID: GHSA-9x64-5r7x-2q53

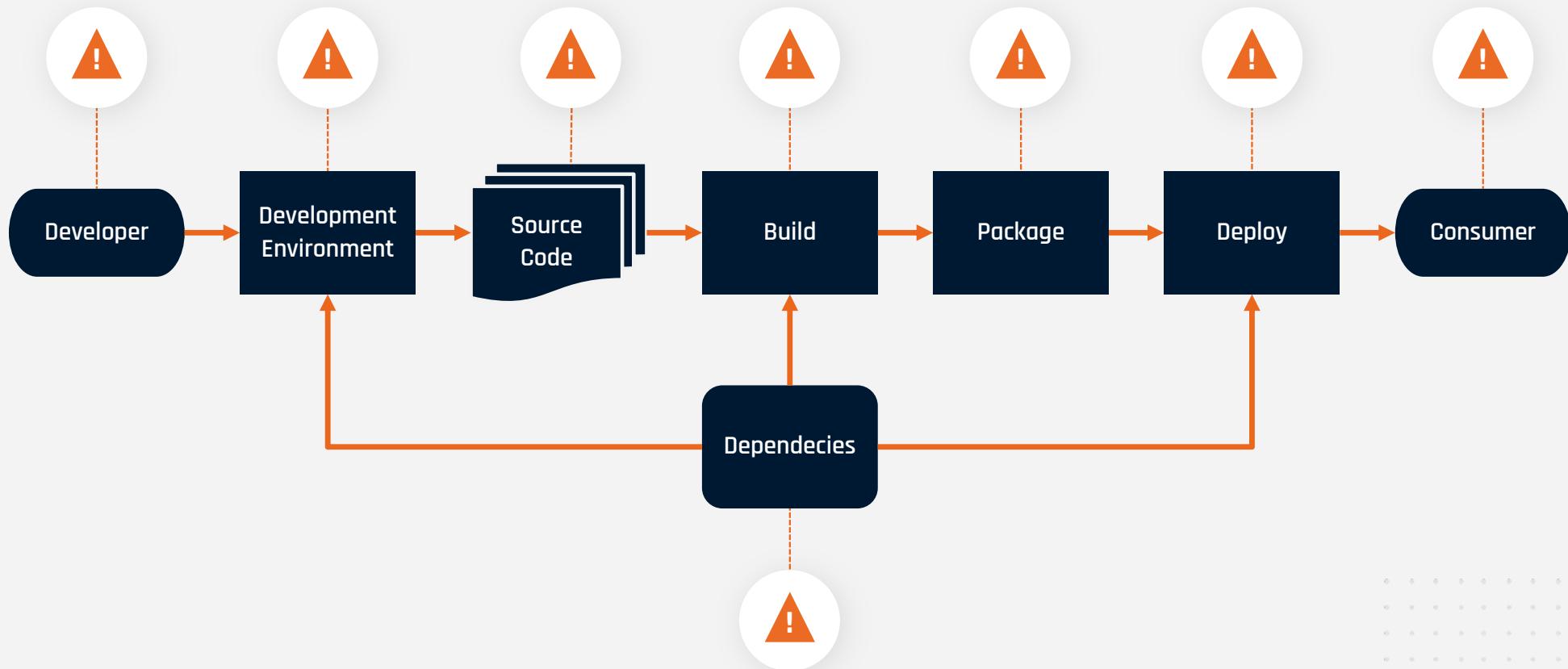
CWEs: CWE-506

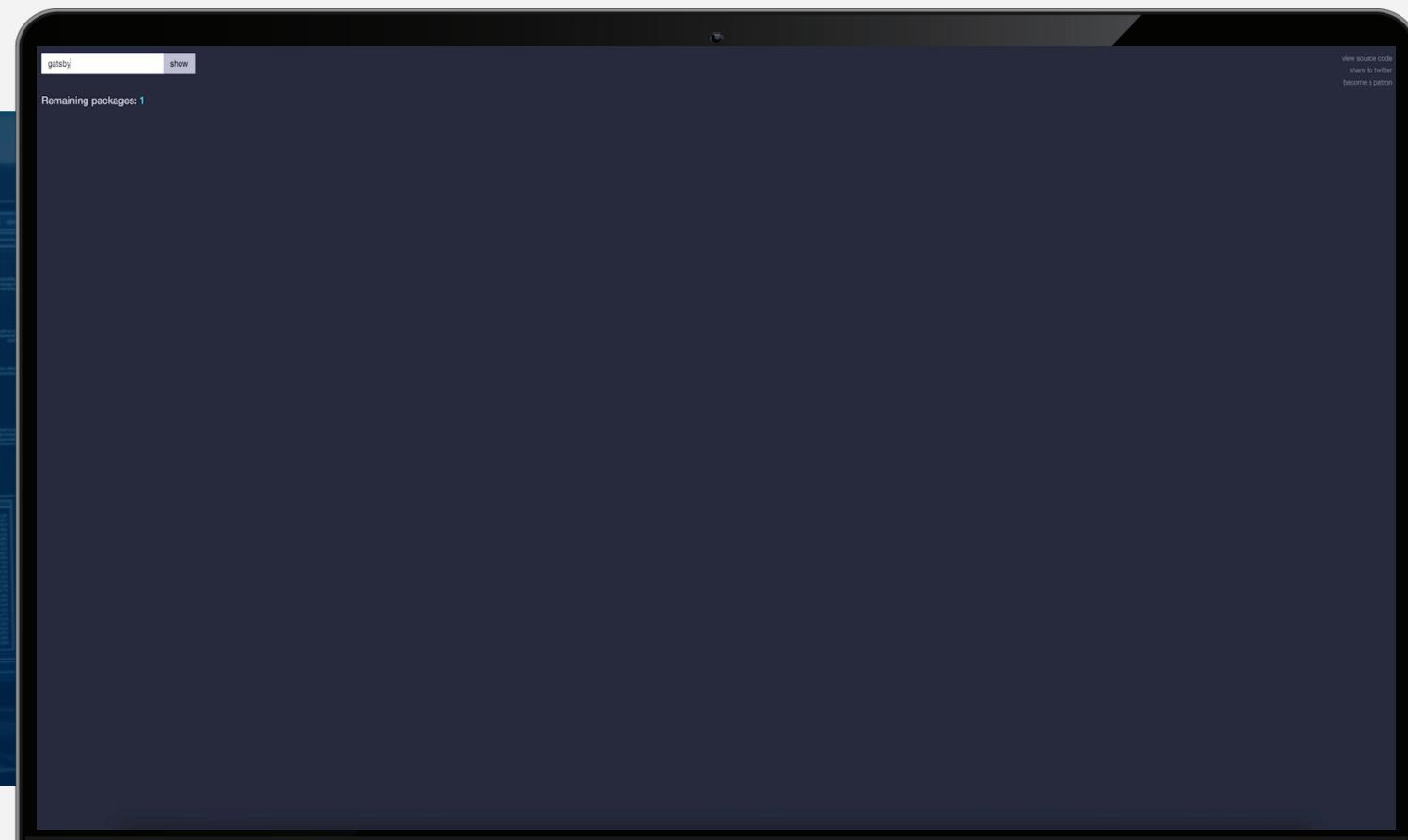
CVSS Score: 9.8 Critical CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

This advisory has been edited. See History.

See something to contribute? Suggest improvements for this vulnerability

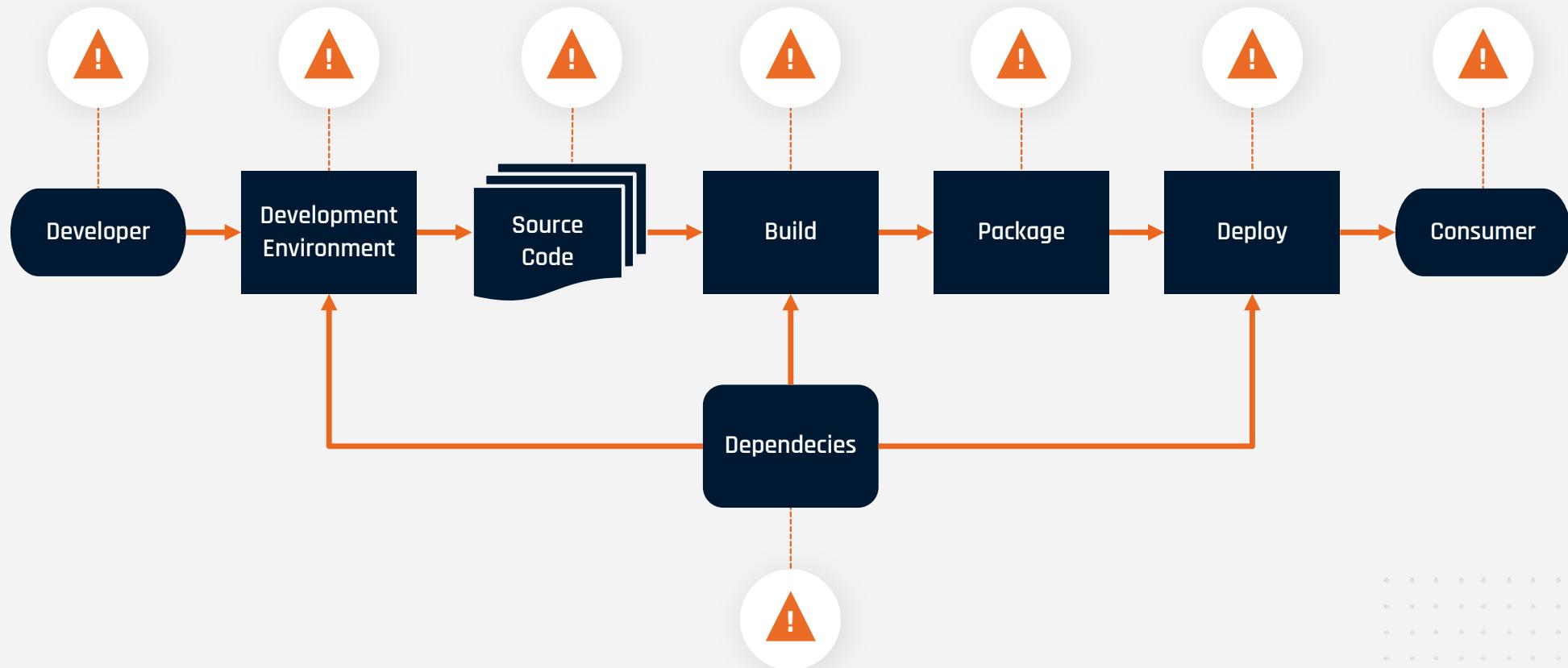
# Attack vectors



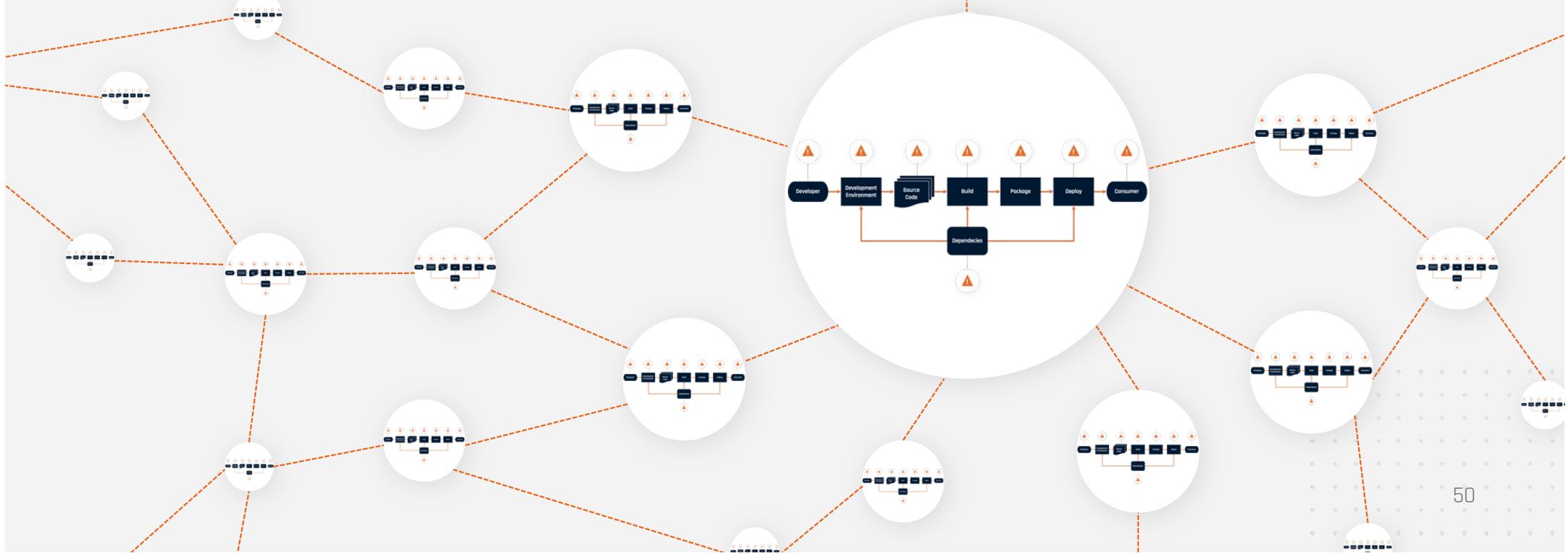
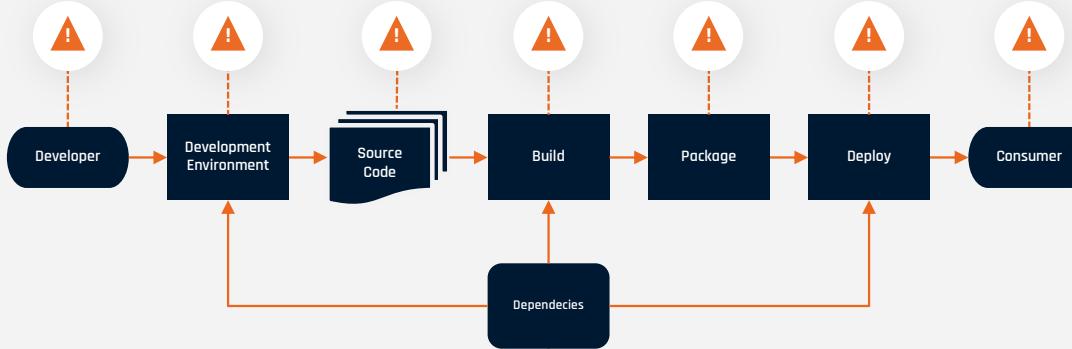


NoteBook

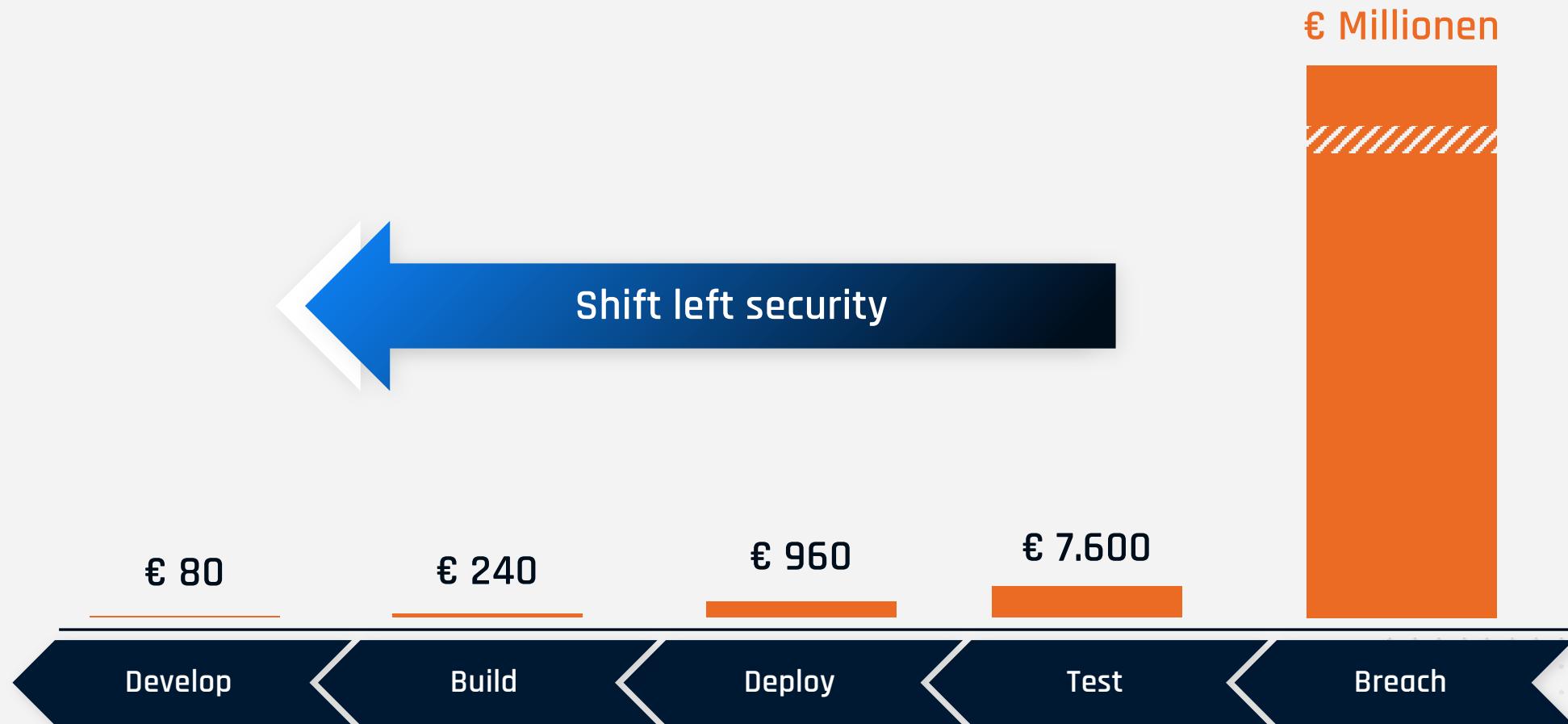
# Attack vectors



# Attack vectors



# Costs for fixing a security vulnerability



Source: Ponemon Institute Cost of a Data Breach 2020

# The Consequences for Developers

- Secure your development environments and pipelines!
- Manage your Software Supply chain and automate the update of (vulnerable) dependencies!
- Automatically securing your code and infrastructure as code using code scanning tools.
- Prevent leaked credentials and manage / rotate credentials in a secure way.
- Use best practices and common standards!
- Learn the basics – stay up to date with security!

Check out: <https://github.com/microsoft/Security-101>

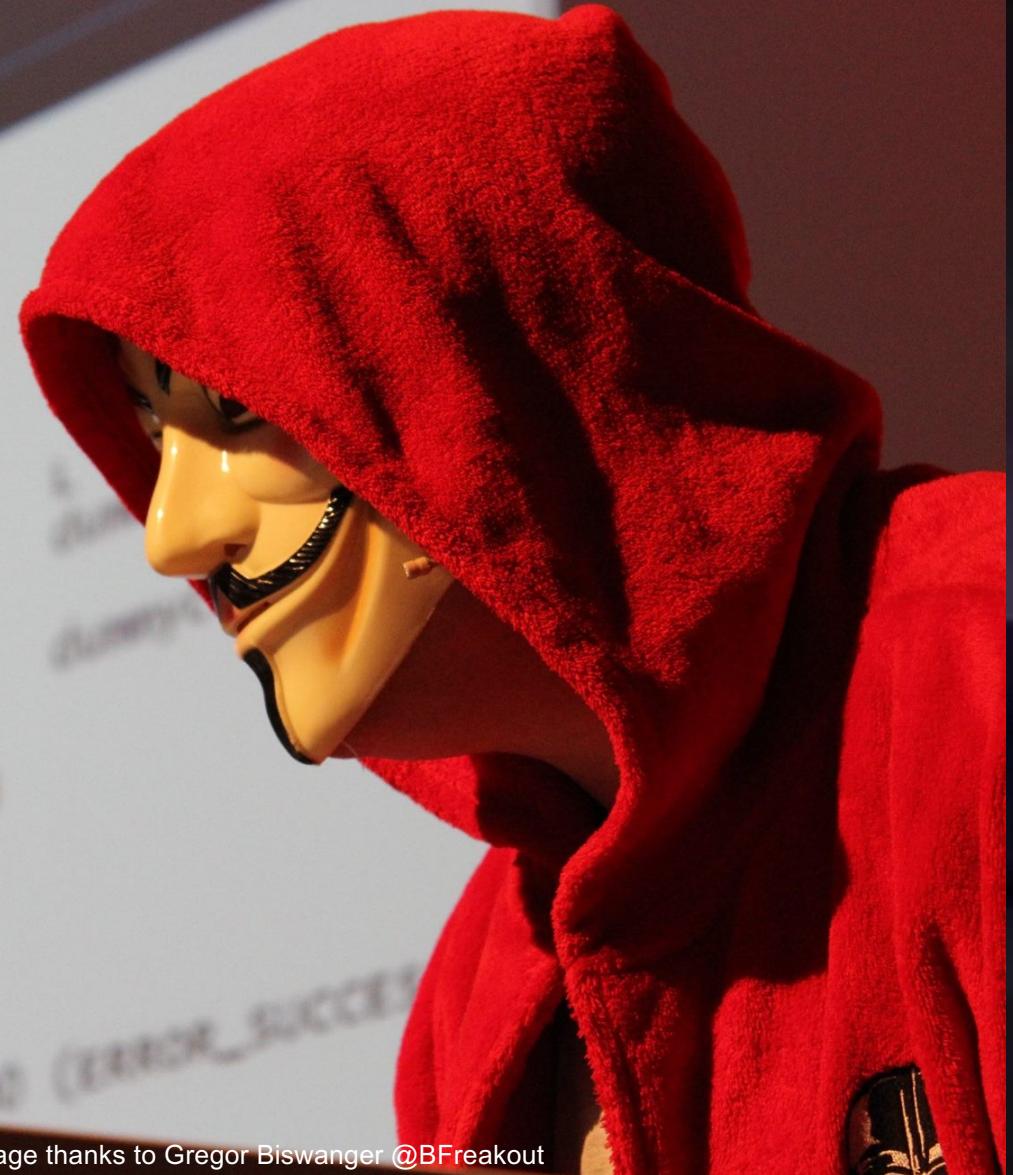


Image thanks to Gregor Biswanger @BFreakout

## Hackers in movies vs



© 2021 Michael Kaufmann @mike\_kaufmann

# Thank you



Blog : <https://writeabout.net>



Twitter : @mike\_kaufmann



GitHub : @wulfland



LinkedIn : <https://www.linkedin.com/in/mikaufmann/>