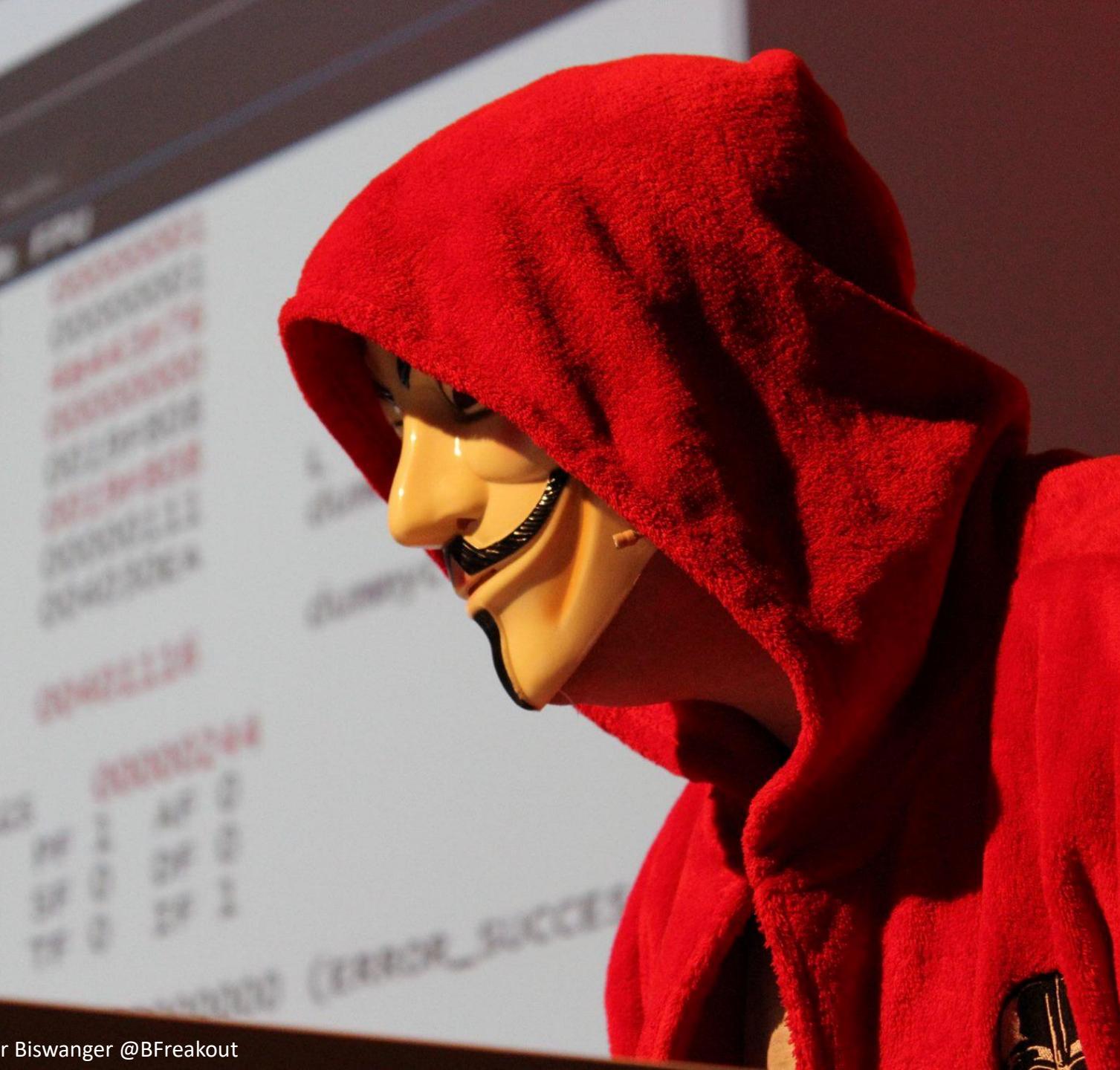


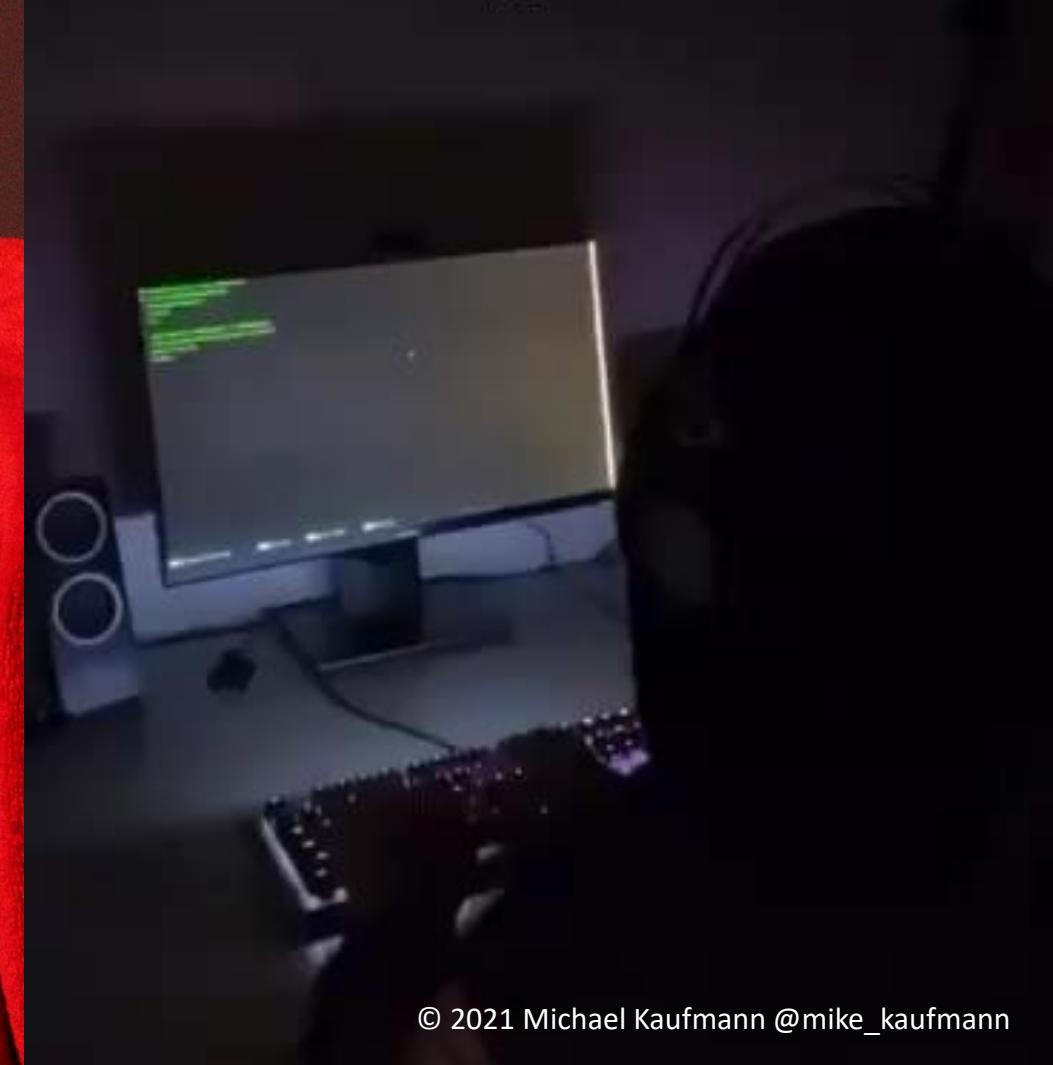
Secure DevOps

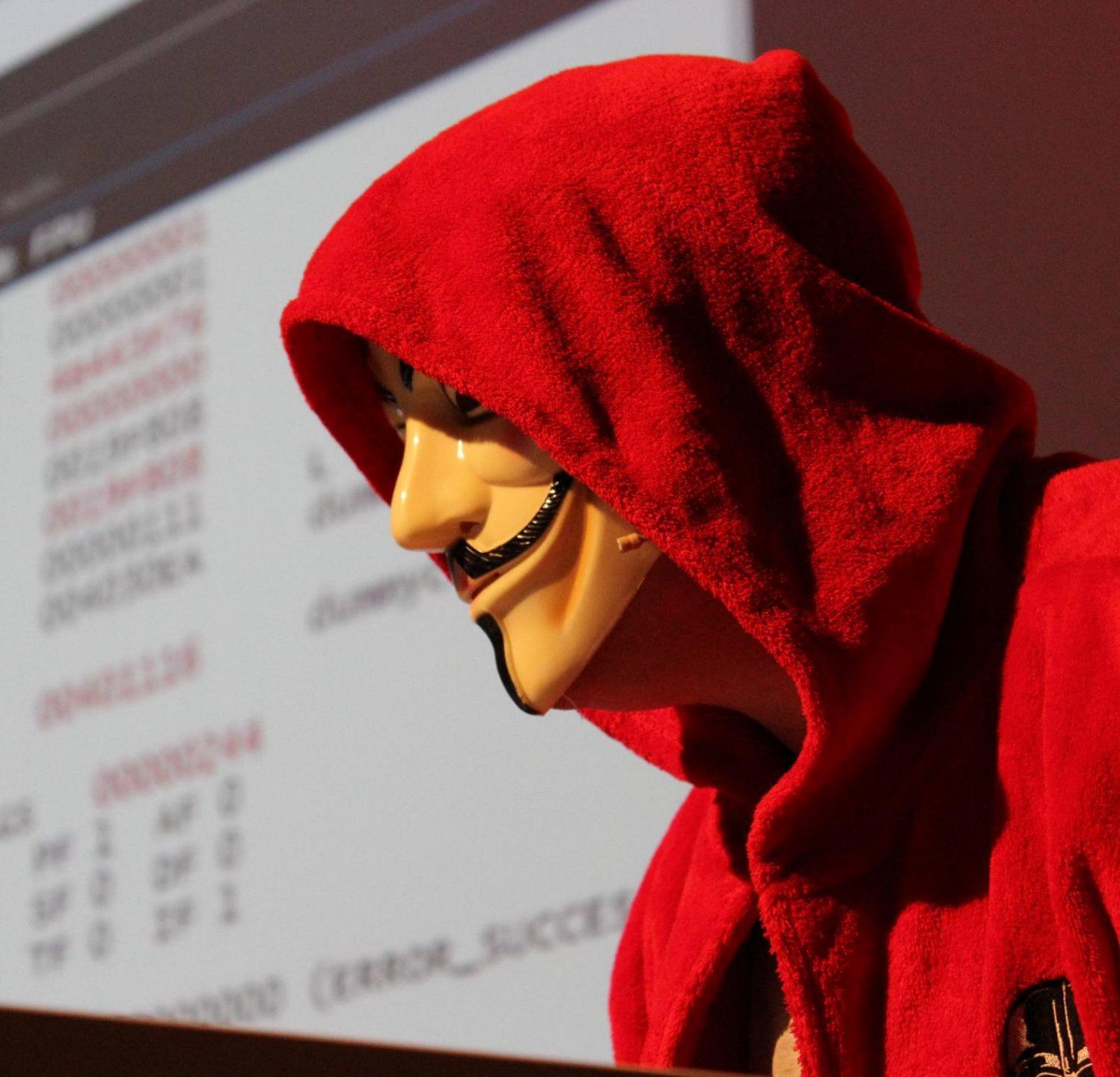
#DevSecOps

Sicherheit in Zeiten von täglichen Deployments



Hackers in movies
vs





How people think they get hacked



How they really get hacked

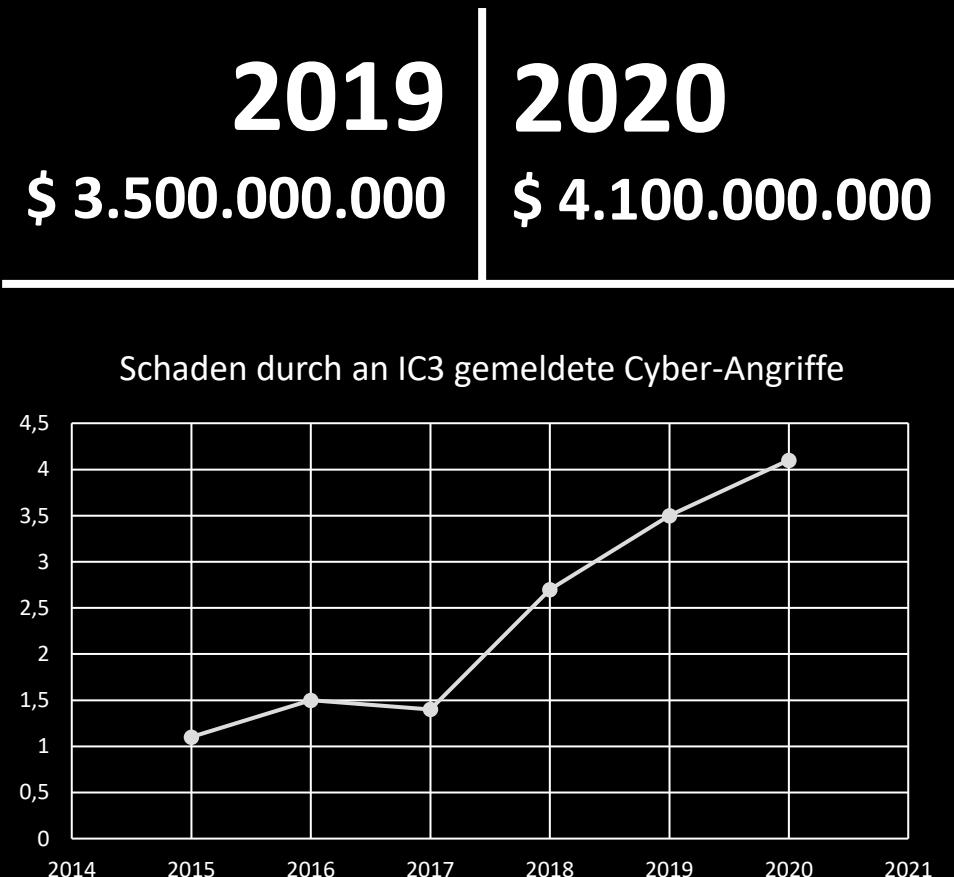
The collage includes:

- A post from 94.1 KXQJ-B asking, "If you had to marry your spouse where you met them, where would your wedding have been?" with 100K comments and 2.3M shares.
- A post from Up 99.3 asking, "Your porn name , is your middle name , and the first car you had." with 26.4K comments and 1.87M shares.
- A post from The Farmer's Wife asking, "The car you passed your drivers test in was a _____" with 14.5K comments and 6.3M shares.
- A post from The Farmer's Wife asking, "Asking where you'd get married again is like asking where you met." with 100K comments and 2.3M shares.
- A post from NAME A SONG asking, "How far away do you live from the place you were born?" with 21.8K comments and 2.8M shares.
- A post from The Farmer's Wife asking, "STOP. THINK. DO NOT SHARE INFO." with 14.5K comments and 6.3M shares.
- A post from The Farmer's Wife asking, "THAT TAKES YOU BACK TO HIGH SCHOOL" with 14.5K comments and 6.3M shares.

Wenn wir über Sicherheit sprechen...

- „Wir hatten bisher noch nie einen Sicherheitsvorfall“
- „Unsere Entwickler sind Experten“
- „Unsere Firma ist zu unwichtig, um angegriffen zu werden“
- „Wir haben uns schon gut abgesichert“
- „Unsere Firewalls sind sicher“

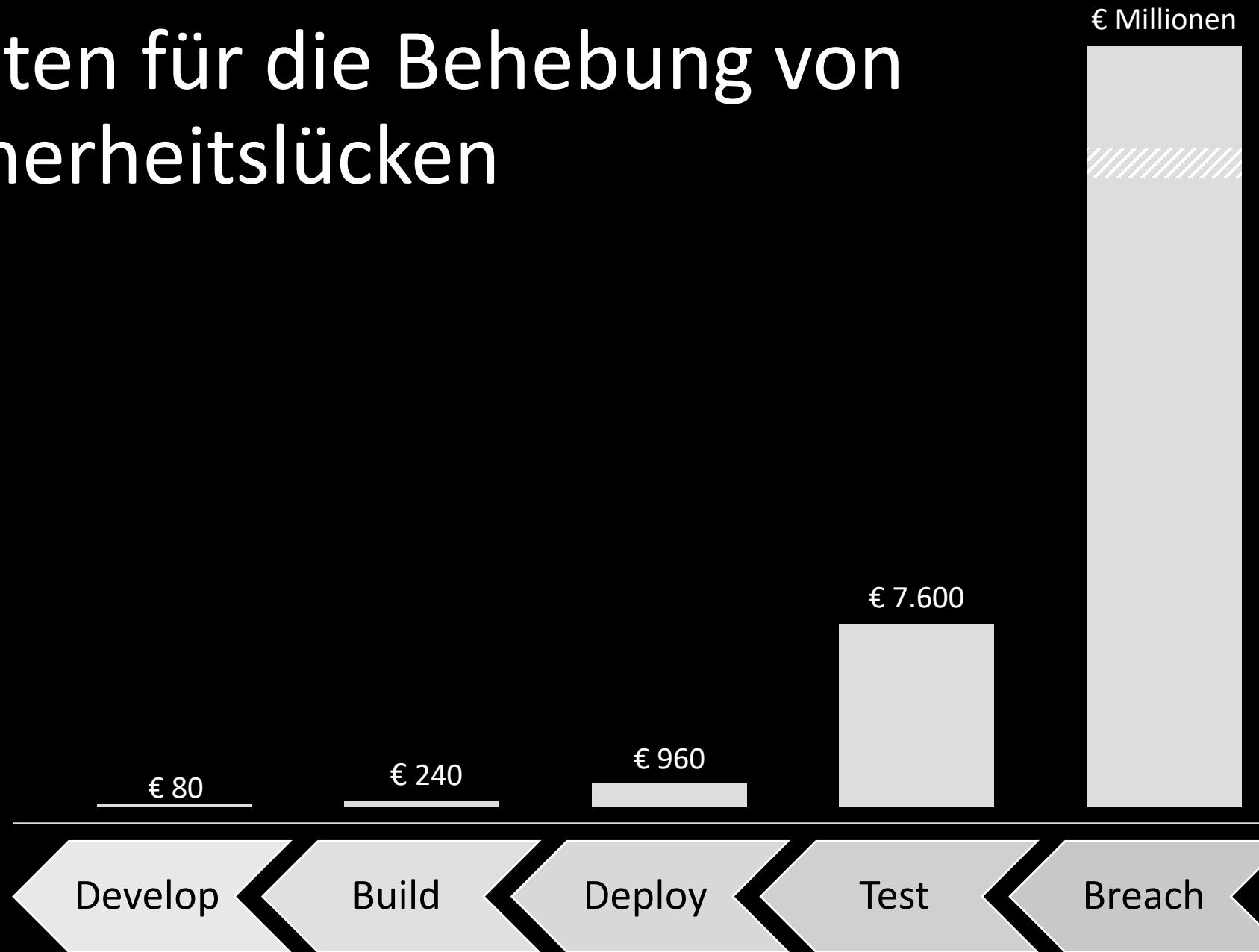
Schaden durch an den IC3 gemeldete Cyber-Angriffe:



SolarWinds

March to June 2020: access to government and other systems through a compromised update to SolarWinds' **Orion** software

Kosten für die Behebung von Sicherheitslücken



Assume-Breach-Paradigma

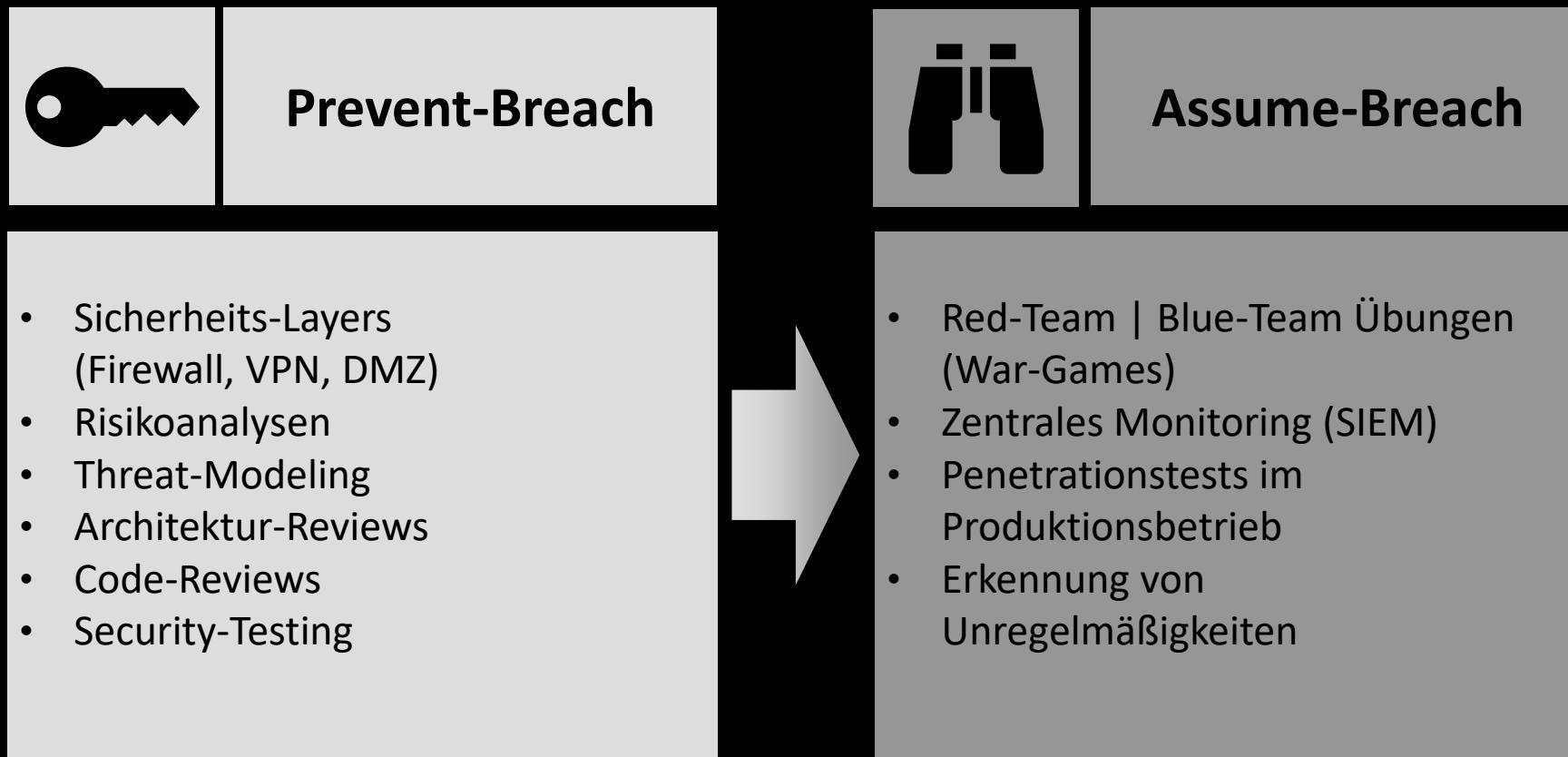
„Im Grunde ist es so, wenn jemand hereinkommen will, dann kommt er auch rein. In Ordnung. Akzeptieren Sie das.“

- Michael V. Hayden

ehemaligen Generals der US Air Force und ehemaliger Direktor der NSA und CIA)



Von Prevent-Breach zu Assume-Breach



Zero-Trust-Policy

- Alle Systeme – auch die internen - sind so gesichert, als wären Sie über das Internet erreichbar: MFA aktiviert, HTTPS verschlüsselt, System immer gepatcht etc.
- Jeder Mitarbeiter hat nur die Zugriffe, die er absolut benötigt solange er sie benötigt. Lokale Administratorrechte werden möglichst vermieden.
- Keiner hat mit seinen normalen Zugangsdaten Zugriff auf Produktionssysteme. Auch Test-Accounts, die auf Entwicklungs- und Testumgebungen verwendet werden, haben keinen Zugriff auf die Produktion.



Red-Team | Blue-Team

- Ziel des Spiels
- Teamaufstellung
- Spieldauer

Red-Team | Blue-Team

Verhaltenskodex:

- Keinen echten Schaden anrichten
- Nicht mehr als nötig tun, um Ziel zu erreichen
- Physikalische Angriffe in normalem Rahmen
(Niemanden bedrohen, keine Zugangskarten stehlen oder drucken etc. – „gesunder Menschenverstand“)
- Die Namen der Personen, die kompromittiert wurden, nicht preisgegeben



Red-Team | Blue-Team

Spielregeln:

- Die Verfügbarkeit des Produktionssystems darf nicht beeinträchtigt werden → rote Team hat gewonnen
- Echte Kundendaten dürfen nicht kompromittiert werden (Daten von internen Kunden schon)
- Kompromittierte Daten (Zugangsdaten, personenbezogene Daten etc.) müssen sicher abgelegt werden
- Die Sicherheit des Produktivsystems darf nicht übermäßig geschwächt werden
- Es darf kein unnötiger Schaden entstehen
- Proof!



A dark, atmospheric photograph of a hooded figure sitting at a desk in what appears to be a server room or a high-tech control center. The figure is seen from behind, wearing a dark hoodie. In front of them are several computer monitors displaying complex, glowing blue and white code and data visualizations. The background is filled with the glowing lights of server racks and other equipment, creating a sense of a high-stakes, technical environment.

Wie funktionieren Angriffe?

- Angriffe auf Entwickler
- Angriffe auf das Engineering-System
- Angriffe auf Produktionssysteme

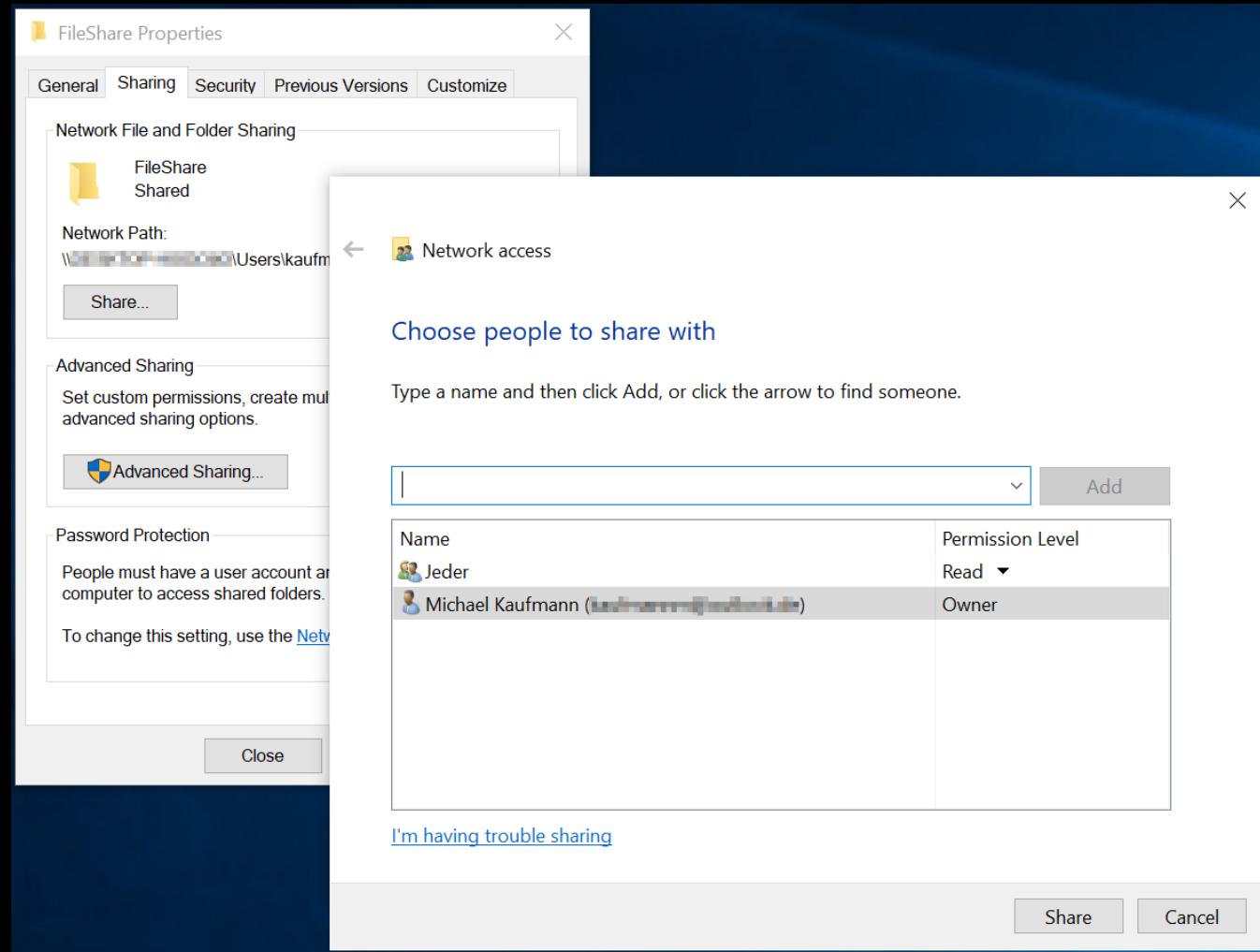
Angriffe auf Entwickler

- Nicht geschützte Dateifreigaben
- Zugangsdaten in Textdateien oder im Code
- Mimikatz
- Phishing

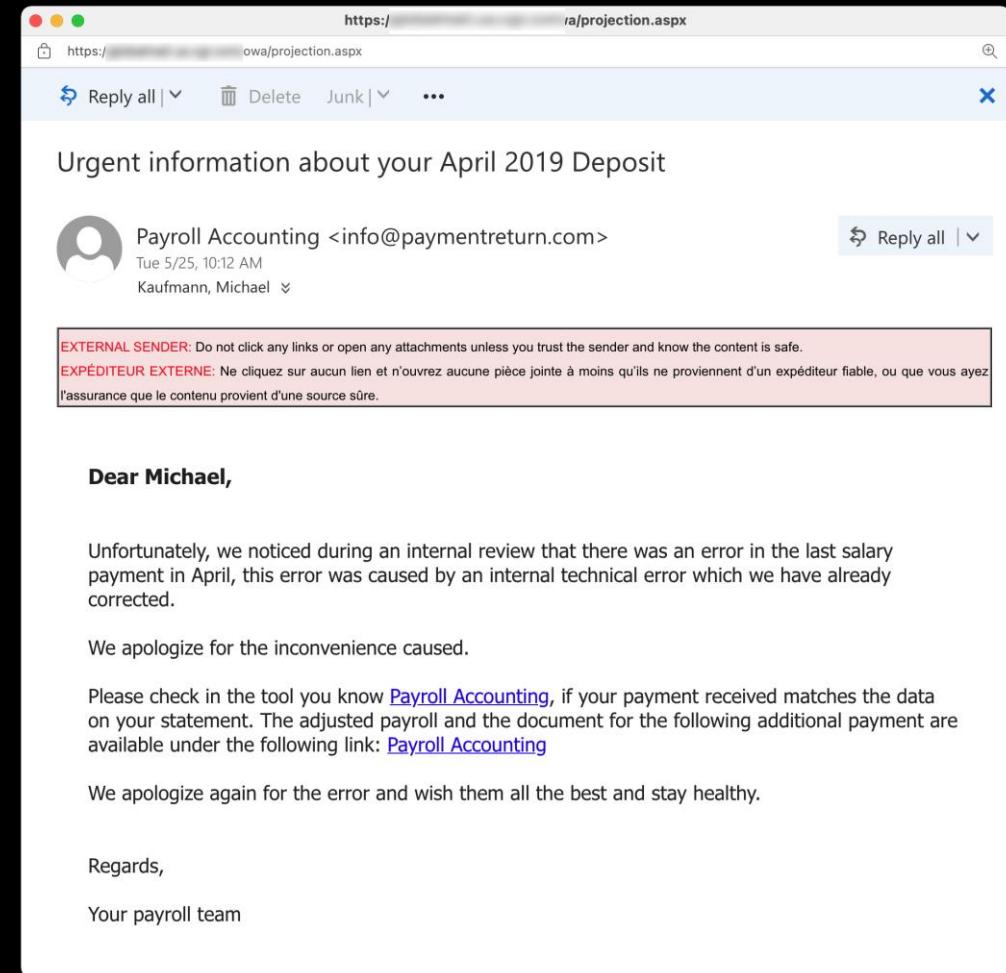
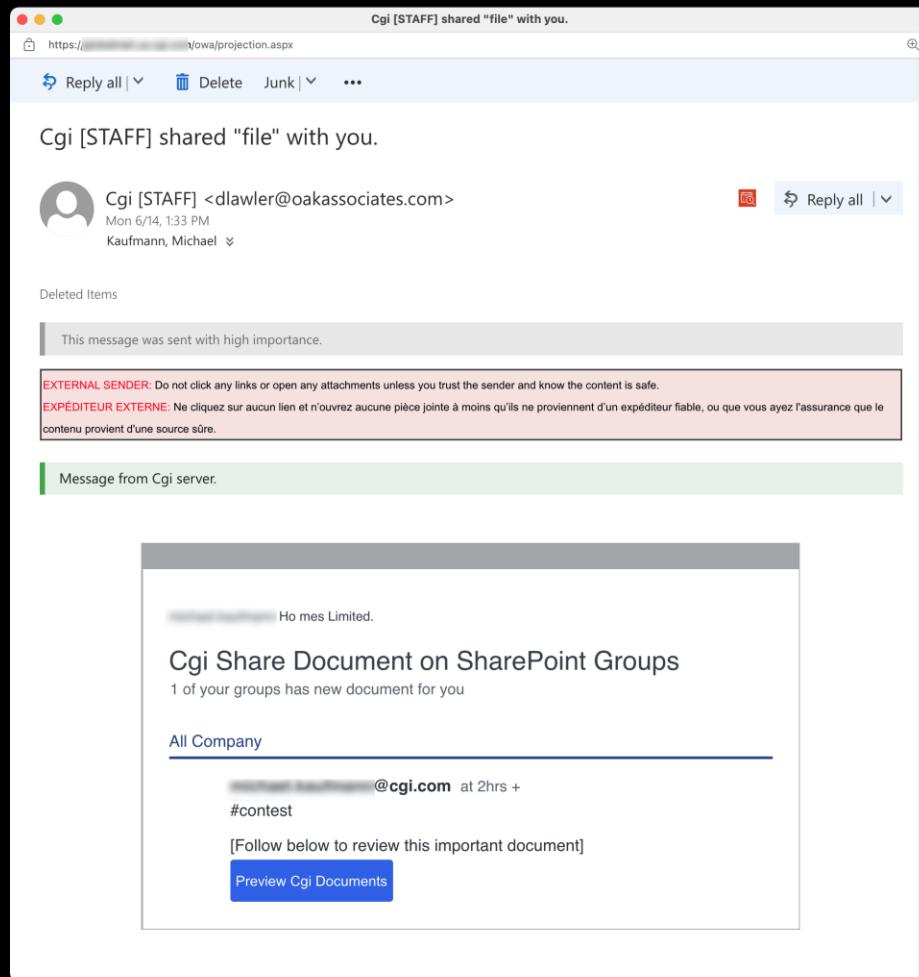
Angriffe auf das Engineering-System

- Entwicklungsumgebungen
- Testumgebungen / Test-Accounts
- Angriffe auf die Software-Lieferketten
- Code-Ändern (eine Zeile reicht)
- Pipeline-Zugriff (Code/Scripte ausführen)

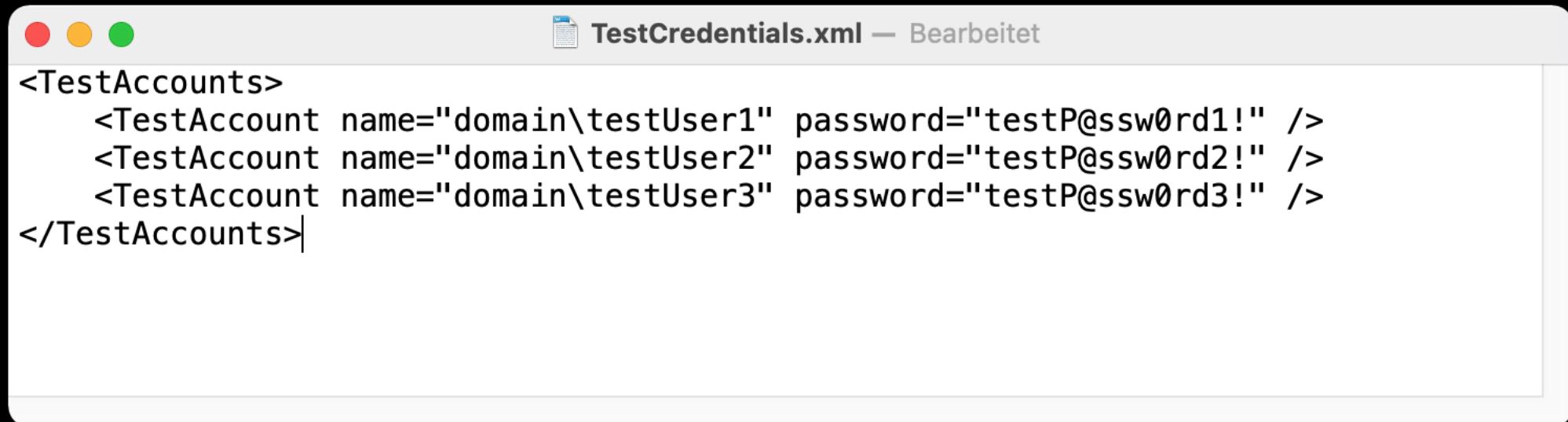
Nicht geschützte Dateifreigaben



Phishing



Zugangsdaten in Textdateien



The screenshot shows a window titled "TestCredentials.xml — Bearbeitet" (Edit) with three test accounts defined in XML:

```
<TestAccounts>
    <TestAccount name="domain\testUser1" password="testP@ssw0rd1!" />
    <TestAccount name="domain\testUser2" password="testP@ssw0rd2!" />
    <TestAccount name="domain\testUser3" password="testP@ssw0rd3!" />
</TestAccounts>
```

kaufm@DESKTOP-HI0G080 ~\OneDrive\Downloads\mimikatz_trunk\x64
.\mimikatz.exe

```
.#####. mimikatz 2.2.0 (x64) #19041 May 31 2021 00:08:47
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ## > https://blog.gentilkiwi.com/mimikatz
'## v ##' Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####' > https://pingcastle.com / https://mysmartlogon.com ***/
```

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonpasswords

```
Authentication Id : 0 ; 12361794 (00000000:00bca042)
Session          : Service from 0
User Name        : 77046479-3B72-43E3-8700-E0E189B60C86
Domain           : NT VIRTUAL MACHINE
Logon Server     : (null)
Logon Time       : 22.06.2021 08:26:38
SID              : S-1-5-83-1-3810489593-1138965362-3789618423-3260611593
```

```
msv :
tspkg :
wdigest :
* Username : DESKTOP-HI0G080$  

* Domain   : WORKGROUP
* Password : (null)
kerberos :
ssp :
```

```
* Username : kaufmann@outlook.de
* Domain   : MicrosoftAccount
* Password : (null)
```

```
ssp :
```

```
credman :
```

```
[00000000]
```

```
* Username : mkgwriteteabout.net
* Domain   : outlook.office365.com
```

```
* Password : 0000-d005-7000-4120-07B09800
```

```
[00000001]
```

```
* Username : (null)
```

```
* Domain   : MicrosoftOffice16 Data:SSPI:mkgwriteteabout.net
```

```
* Password : 0000-d005-7000-4120-07B09800
```

```
[00000002]
```

```
* Username : github.auth
```

```
* Domain   : vscodevscode.github-authentication/github.auth
```

```
* Password : 0000-d005-7000-4120-07B09800
```

```
00 4b 60 6c 40 46 37 4c 4f 72 4e 37 67 66 6a 44 55 36 4b 6a 51 36 65 6b 64 45 38 30 2f 0d 4a 35 61 78 57 42 4e 58 44 71
```

```
42 43 4a 52 44 74 42 39 47 63 34 45 4a 59 5a 69 43 36 4b 4a 2f 53 49 68 39 6e 55 67 6f 4e 6e 53 57 67 71 4a 47 6d 33 58
```

```
4e 30 34 34 75 4c 6e 4c 52 42 2f 38 32 43 51 8d 6a 45 41 43 61 2f 34 5a 4e 66 2b 47 54 6b 53 56 4d 69 75 52 79 49 74 4e
```

```
56 76 5a 5a 6f 35 42 68 4e 75 75 58 61 38 2f 4f 6a 58 5a 38 62 7a 7a 53 71 50 74 61 62 43 78 74 31 6b 61 36 35 59 74 6d
```

```
55 6d 6a 6a 54 75 6a 4e 66 47 6a 65 42 45 41 45 61 2f 34 5a 4e 66 2b 47 54 6b 53 56 4d 65 75 52 79 49 74 4e
```

```
64 4b 60 6c 40 46 37 4c 4f 72 4e 37 67 66 6a 44 55 36 4b 6a 51 36 65 6b 64 45 38 30 2f 0d 4a 35 61 78 57 42 4e 58 44 71
```

```
42 43 4a 52 44 74 42 39 47 63 34 45 4a 59 5a 69 43 36 4b 4a 2f 53 49 68 39 6e 55 67 6f 4e 6e 53 57 67 71 4a 47 6d 33 58
```

```
4e 30 34 34 75 4c 6e 4c 52 42 2f 38 32 43 51 8d 6a 45 41 43 61 2f 34 5a 4e 66 2b 47 54 6b 53 56 4d 65 75 52 79 49 74 4e
```

```
56 76 5a 5a 6f 35 42 68 4e 75 75 58 61 38 2f 4f 6a 58 5a 38 62 7a 7a 53 71 50 74 61 62 43 78 74 31 6b 61 36 35 59 74 6d
```

```
72 8d 6a 53 38 2b 72 53 6e 69 47 46 34 42 55 33 48 6a 70 4f 5a 73 43 47 6a 6e 46 63 37 6d 73 46 6f 4e 78 64 78 6a 37 59
```

```
64 43 2b 47 48 5a 4a 37 68 79 2b 78 48 6f 46 48 7a 32 51 76 6d 2f 78 5a 6a 36 75 8d 8a 38 55 49 6c 47 38 52 2b 71 37 47
```

```
4d 55 34 44 6f 72 43 7a 69 4a 78 4a 47 31 59 67 49 51 34 51 45 79 61 7a 55 4e 49 67 4f 78 63 68 34 47 4f 53 44 41 6e 35
```

```
5a 53 6d 2b 41 4c 51 37 38 6b 37 53 6a 8d 8a 84 5a 40 34 74 6c 45 6e 73 31 77 68 63 35 59 60 2f 68 33 56 32 78 39 5a
```

69 57 37 74 2f 54 6e 2f 64 6d 74 55 73 2f 42 6e 41 3d 3d 0d 4a 00 00

[00000003]

* Username : wulfland

* Domain : GitHub - https://api.github.com/wulfland

* Password : 096d805700410070000

[00000004]

* Username : wulfland@hotmail.com

* Domain : https://gitlab.com

* Password : 096d805700410070000

cloudap :

Cachedir : d470e18c8f14149c4504f151e1041892c5fe9c1a50cc2b0ea1fdaf444010495d

Key GUID : {00000000-5f1c-30a9-0000-000000000000}

PRT : []

DPAPI Key: 540039007500670039006a006d0057004100700034000b00300034005100070009005000310003006100790051006000

500001000c00003007B0000005000050005000500078eff4292dd33423cd146db9f101ec09a1b69c4faaf (sha1: 1457570c0ea9fc74668051f3e1185ec
c8411575)

Authentication Id : 0 ; 195058 (00000000:0002f9f2)

Session : Interactive from 1

User Name : DWM-1

Domain : Window Manager

Logon Server : (null)

Logon Time : 21.06.2021 22:24:46

SID : S-1-5-20-0-1

msv :

tspkg :

wdigest :

* Username : DESKTOP-HJ8G09S

* Domain : WORKGROUP

* Password : (null)

Beispiel



Zugangsdaten Entwickler

- E-Mail → Phishing von Kollegen
- Zugriff auf Maschinen → Logon, Mimikatz
- Quellcode → Code einfügen/modifizieren
- Zugriff Pipeline → Skripte ausführen
- Zugriff Testumgebung → Testet Prod?
- Zugriff Prod → Game Over



```
[  
  "/api/*": {  
    "changeOrigin": true,  
    "target": "https://api.project-demo.de",  
    "auth": "REDACTED"  
  }  
]
```

Schema: <https://json.schemastore.org/appsettings.json>

```
1  {
2    "AppSettings": {  
3      "ScheduleExecutionIntervalInMinutes": 60  
4    },  
5    "MailSettings": {  
6      "Url": "Outlook.office365.com",  
7      "User": "REDACTED",  
8      "Password": "REDACTED",  
9      "ProcessedFolderName": "REDACTED",  
10     "ErrorFolderName": "REDACTED",  
11     "OutOfToleranceFolderName": "REDACTED"  
12   }  
13 }
```

```
const id_token_string = 'REDACTED';  
  
app.use(  
  '/api/test',  
  createProxyMiddleware({  
    target: 'https://api.test.REDACTED.com/',  
    changeOrigin: true,  
    pathRewrite: {  
      '^/api/test': '/test',  
    },  
    headers: {  
      'Content-Type': 'application/json',  
      'Credentials': true,  
      'Cookie': `id_token=${id_token_string}`  
    },  
  });
```

27 lines (27 sloc) | 1.45 KB

```
1  <?xml version="1.0" encoding="utf-8" ?>
2  <configuration>
3      <configSections>
4          <sectionGroup name="applicationSettings" type="System.Configuration.ApplicationSettingsG</pre>
```

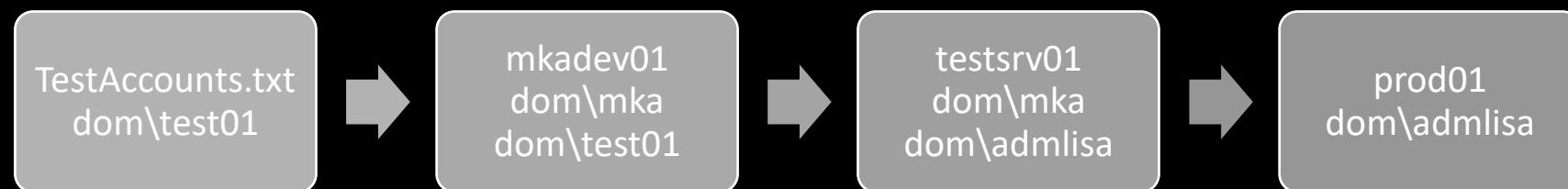
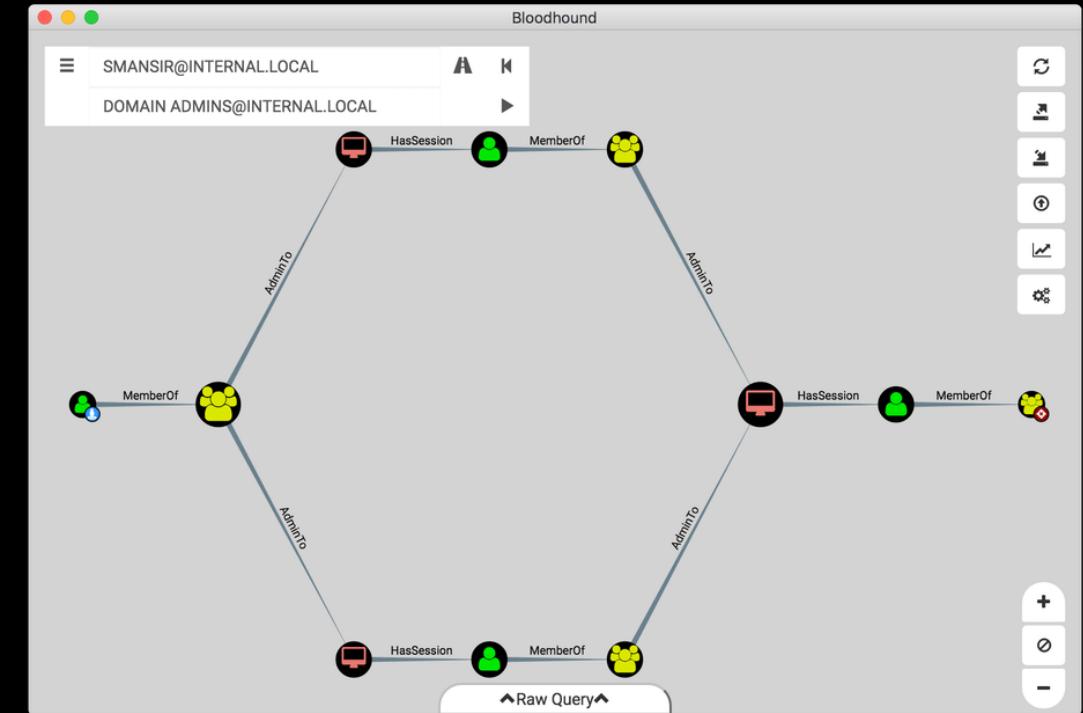
Von Dev zu Prod

Bloodhound: <https://github.com/adaptivethreat/Bloodhound>

grpmgr test01 Administrators /enum

Andere Möglichkeiten an Zugangsdaten zu kommen:

- Phishing
- Responder (<https://github.com/Igandx/Responder>)
- Pineapple
- ...
- Schwache Passwörter



Angriffe auf das Produktionssystem

- SQL-Injection
- XSS (Cross-Site-Scripting)



SQL Injection

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

105; DROP TABLE Suppliers

```
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     }
38 }
```

XSS (Cross-Site-Scripting)

```
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 }
```

```
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()
}
```

Was kann man tun (Blue-Team)?

- MFA für alle Systeme
- Lokale Adminrolle entziehen
- Strikte Trennung Test/Prod
- Key-Vault für alle Secrets
 - Trennung Laufzeit/deployment
 - Test/Prod
 - Automatische Rotation
- Separate Subscriptions
- Codespaces / VDI für Development
- SIEM (Azure Security Center, Splunk)
- Credential Scanning (Code/Fileshares)



How to shift left security?



Secret Scanning

Adafruit IO	Dropbox	Plivo
Adafruit IO Key	Dropbox Access Token	Plivo Auth Token
Adobe	Dropbox Short Lived Access Token	Postman
Adobe Device Token	Dynatrace	Postman API Key
Adobe JSON Web Token	Dynatrace Access Token	Proctorio
Adobe Service Token	Dynatrace Internal Token	Proctorio Consumer Key
Adobe Short-Lived Access Token	Finicity	Proctorio Linkage Key
Alibaba Cloud	Finicity App Key	Proctorio Registration Key
Alibaba Cloud Access Key ID and Access Key Secret pair	Frame.io	Proctorio Secret Key
Amazon Web Services (AWS)	Frame.io Developer Token	Pulumi
Amazon AWS Access Key ID and Secret Access Key pair	Frame.io JSON Web Token	Pulumi Access Token
Atlassian	GitHub	PyPI
Atlassian API Token	GitHub App Installation Access Token	PyPI API Token
Atlassian JSON Web Token	GitHub OAuth Access Token	RubyGems
Azure	GitHub Personal Access Token	RubyGems API Key
Azure Active Directory Application Secret	GitHub Refresh Token	Samsara
Azure DevOps Personal Access Token	GitHub SSH Private Key	Samsara API Token
Azure SAS Token	GoCardless	Samsara OAuth Access Token
Azure Service Management Certificate	GoCardless Live Access Token	SendGrid
Azure SQL Connection String	GoCardless Sandbox Access Token	SendGrid API Key
Azure Storage Account Key	Google Cloud	Shopify
Clojars	Google API Key	Shopify Access Token
Clojars Deploy Token	Google Cloud Private Key ID	Shopify App Shared Secret
CloudBees CodeShip	Hashicorp Terraform	Shopify Custom App Access Token
CloudBees CodeShip Credential	Terraform Cloud / Enterprise API Token	Shopify Private App Password
Databricks	Hubspot	Slack
Databricks Access Token	Hubspot API Key	Slack API Token
Datadog	Mailchimp	Slack Incoming Webhook URL
Datadog API Key	Mailchimp API Key	Slack Workflow Webhook URL
Discord	Mandrill API Key	SSLMate
Discord Bot Token	Mailgun	SSLMate API Key
Doppler	Mailgun API Key	SSLMate Cluster Secret
Doppler CLI Token	MessageBird	Stripe
Doppler Personal Token	MessageBird API Key	Stripe Live API Restricted Key
Doppler SCIM Token	npm	Stripe Live API Secret Key
Doppler Service Token	npm Access Token	Stripe Test API Restricted Key
	NuGet	Stripe Test API Secret Key
	NuGet API Key	Tencent Cloud
	OpenAI	Tencent Cloud Secret ID
	OpenAI API Key	Twilio
	Palantir	Twilio Account String Identifier
	Palantir JSON Web Token	Twilio API Key
		Valour
		Valour Access Token

Code

- GitHub (private!)
- gitLeaks.
- SpectralOps
- Git-Secrets
- Whispers
- Gittyleaks
- Git-all-secrets
- ...

Fileshare

- Bash/PowerShell
- Dumpster

Software Composition Analysis (SCA)

GitHub (Dependency-Graph/Dependabot)
anchore (<https://anchore.com/>) oder
Dependency-Track (<https://dependencytrack.org/>)

Dependabot alerts

Dismiss all ▾

⚠ 4 Open ✓ 0 Closed

Manifest ▾ Sort ▾

Severity	Alerts
critical severity	marsdb 3 minutes ago by GitHub package.json
high severity	express-jwt 3 minutes ago by GitHub package.json
moderate severity	sanitize-html 3 minutes ago by GitHub package.json
critical severity	jsonwebtoken 3 minutes ago by GitHub package.json

Dependency graph

Dependencies Dependents Dependabot

⚠ We found potential security vulnerabilities in your dependencies.
Dependencies defined in these manifest files have known security vulnerabilities and should be updated:
[package.json](#) 7 vulnerabilities found

[View Dependabot alerts](#)

Only the owner of this repository can see this message.

These dependencies are defined in `workshop-2021-learning-journey`'s manifest files, such as [package.json](#) and [frontend/package.json](#).

Dependency	Vulnerability	Version
auth0 / express-jwt	Known security vulnerability in 0.1.3	0.1.3
auth0 / node-jsonwebtoken	Known security vulnerability in 0.4.0	0.4.0
c58 / marsdb	Known security vulnerability in ^ 0.6.11	^ 0.6.11
apostrophecms / sanitize-html	Known security vulnerability in 1.4.2	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

Software Composition Analysis (SCA)

GitHub (Dependency-Graph/**Dependabot**)
anchore (<https://anchore.com/>) oder
Dependency-Track (<https://dependencytrack.org/>)

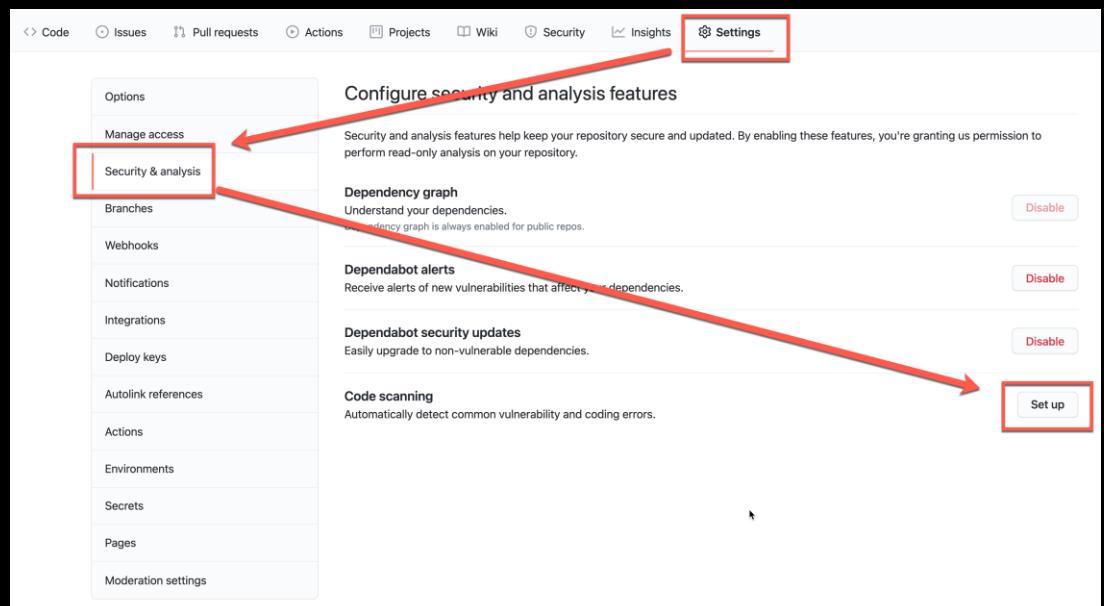
The screenshot shows a GitHub security alert digest for the repository 'wulfland/DemoShop'. It highlights a known security vulnerability for the 'bootstrap' dependency. The current version is 3.0.0, and it suggests upgrading to 3.4.1. The vulnerability is identified as CVE-2019-8331 with a moderate severity. A blue button at the bottom right encourages the user to 'Review all vulnerable dependencies'.

The screenshot shows a GitHub pull request titled 'Bump bootstrap from 3.0.0 to 3.4.1 in /src/CupCakeHeaven/CupCakeHeaven #1'. The pull request is from the 'dependabot' bot and merges into the 'master' branch. It fixes a security vulnerability. The PR includes a message from 'dependabot' bot stating: 'Bumps bootstrap from 3.0.0 to 3.4.1. Dependabot will resolve any conflicts with this PR as long as you don't alter it yourself. You can also trigger a rebase manually by commenting @dependabot rebase.' The pull request has a 'Verified' status and a commit hash 'da99c84'. The right sidebar shows the pull request details: no reviews, still in progress, no assignees, and a 'dependencies' label. The bottom section of the screenshot shows the GitHub pull request interface with options to 'Merge pull request' or 'Close pull request'.

Static Application Security Testing (SAST)

Whitebox-Testing

- GitHub CodeQL
- SonarQube
- Semgrep (<https://semgrep.dev/>)
- Mobile-Security-Framework (MobSF)
(<https://github.com/MobSF/Mobile-Security-Framework-MobSF>)



A screenshot of the GitHub Marketplace page for code scanning. The top section is titled 'Get started with code scanning' with the sub-instruction 'Automatically detect common vulnerabilities and coding errors'. It features a card for 'CodeQL Analysis by GitHub' which performs security analysis from GitHub for C, C++, C#, Java, JavaScript, TypeScript, Python, and Go developers. Below this, there are cards for various security analysis tools from the Marketplace:

- 42Crunch API Security Audit by 42crunch: Use the 42Crunch API Security Audit REST API to perform static application security testing (SAST) on OpenAPI/Swagger files.
- Codacy Security Scan by Codacy: Free, out-of-the-box, security analysis provided by multiple open source static analysis tools.
- CxSAST by Checkmarx: Scan your code with Checkmarx CxSAST and see your results in the GitHub security tab.
- DefenseCode ThunderScan by DefenseCode: Scan your code with ThunderScan® SAST to detect security vulnerabilities in more than 30 programming languages.
- DevSkim by Microsoft CST-E: DevSkim is a security linter that highlights common security issues in source code.
- Fortify on Demand Scan by Micro Focus: Integrate Fortify's comprehensive static code analysis (SAST) for 27+ languages into your DevSecOps workflows to build secure software faster.
- Kubescan by Controlplane: Security risk analysis for Kubernetes resources. Submit pod-types (such as deployment, cronjob) to receive an itemised security risk score.
- Mayhem for API by ForAllSecure: Automatically test your REST APIs with your OpenAPI specs and Postman collections.

Each card includes a 'Set up this workflow' button.

Code Scanning (CodeQL)

Fork: <https://github.com/advanced-security/workshop-2021-learning-journey>

```
name: "CodeQL"

on:
  push:
    branches: [ master ]
  pull_request:
    # The branches below must be a subset of the branches above
    branches: [ master ]

  jobs:
    analyze:
      name: Analyze
      runs-on: ubuntu-latest
      permissions:
        actions: read
        contents: read
        security-events: write

    strategy:
      fail-fast: false
      matrix:
        language: [ 'javascript', 'java' ]
        # CodeQL supports [ 'cpp', 'csharp', 'go', 'java', 'javascript', 'python' ]

    steps:
      - name: Checkout repository
        uses: actions/checkout@v2

      # Initializes the CodeQL tools for scanning.
      - name: Initialize CodeQL
        uses: github/codeql-action/init@v1
        with:
          languages: ${matrix.language}
          # If you wish to specify custom queries, you can do so here or in a config file.
          # By default, queries listed here will override any specified in a config file.
          # Prefix the list here with "+" to use these queries and those in the config file.
          # queries: ./path/to/local/query, your-org/your-repo/queries@main

      # Autobuild attempts to build any compiled languages (C/C++, C#, or Java).
      # If this step fails, then you should remove it and run the build manually (see below)
      - name: Autobuild
        uses: github/codeql-action/autobuild@v1
```

The screenshot shows the GitHub Security tab with the 'Code scanning' section selected. It displays a summary of the latest scan (3 days ago, main branch, CodeQL tool, 50.5k / 49.7k lines scanned, 3m 33s duration, 50 alerts) and a list of open alerts. The alerts are categorized by severity and include:

- Client-side cross-site scripting (main)
- Missing CSRF middleware (main)
- Uncontrolled data used in path expression (main)
- Arbitrary file write during zip extraction ("Zip Slip") (main)

Code Scanning (CodeQL)

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
151 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     })
158   }
159 }
```

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

-o Create codeql-analysis.yml

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

Verified 9474e68

Client-side cross-site scripting

7 steps in search-result.component.ts ▾ X

Step 1 this.ro ... yParams

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

```
142
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
```

Step 2 this.ro ... arams.q

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

```
142
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
```

Step 3 queryParam

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

```
142
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
```

Step 4 queryParam

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

```
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
```

Step 5 queryParam.trim()

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

Code Scanning (CodeQL)

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
151 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     })
158   }
159 }
```

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

-o Create codeql-analysis.yml

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

Verified ✅ 9474e68

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=1>" + 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 ^

Code Scanning (CodeQL)

The image shows two screenshots of a code scanning tool interface. On the left, a pull request for a file named `search-result.component.ts` is being reviewed. A specific line of code at `L152` is highlighted in yellow, indicating a security vulnerability:

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

A tooltip below the line explains the issue: "Cross-site scripting vulnerability due to user-provided value." The right screenshot shows the generated CodeQL query for this rule. A red arrow points from the "View source" link in the pull request to the corresponding line in the query code:

```
1  /**
2  * @name Client-side cross-site scripting
3  * @description Writing user input directly to the DOM allows for
4  *             a cross-site scripting vulnerability.
5  * @kind path-problem
6  * @problem.severity error
7  * @security-severity 6.1
8  * @precision high
9  * @id js/xss
10 * @tags security
11 *      external/cwe/cwe-079
12 *      external/cwe/cwe-116
13 */
14
15 import javascript
16 import semmle.javascript.security.dataflow.DomBasedXss::DomBasedXss
17 import DataFlow::PathGraph
18
19 from DataFlow::Configuration cfg, DataFlow::PathNode source, DataFlow::PathNode sink
20 where
21   (
22     cfg instanceof HtmlInjectionConfiguration or
23     cfg instanceof JQueryHtmlOrSelectorInjectionConfiguration
24   ) and
25     cfg.hasFlowPath(source, sink)
26 select sink.getNode(), source, sink,
27     sink.getNode().(Sink).getVulnerabilityKind() + " vulnerability due to $@.", source.getNode(),
28     "user-provided value"
```

Code Scanning (CodeQL)

Database query built from user-controlled sources

Building a database query from user-controlled sources is vulnerable to insertion of malicious code by the user.

[Open](#) [Error](#) [CWE-89](#) [security](#)

Branch: main ▾

Dismiss ▾

routes/search.ts

```
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 ...')
16   .then(([products]) => {
17     const dataString = JSON.stringify(products)
18     if (utils.notSolved(challenges.unionSqlInjectionChallenge)) { // vuln-code-snippet hide-start
```

This query depends on a user-provided value.

CodeQL [Show paths](#)

Tool	Rule ID	Query
CodeQL	js/sql-injection	View source

If a database query (such as a SQL or NoSQL query) is built from user-provided data without sufficient sanitization, a malicious user may be able to run malicious database queries.

Show more ▾

First detected in commit 9474e68 3 days ago

27 lines (25 sloc) | 883 Bytes

```
1 /**
2  * @name Database query built from user-controlled sources
3  * @description Building a database query from user-controlled sources is vulnerable to insertion of
4  *             malicious code by the user.
5  * @kind path-problem
6  * @problem.severity error
7  * @security-severity 8.8
8  * @precision high
9  * @id js/sql-injection
10 * @tags security
11 *         external/cwe/cwe-089
12 */
13
14 import javascript
15 import semmle.javascript.security.dataflow.SqlInjection
16 import semmle.javascript.security.dataflow.NosqlInjection
17 import DataFlow::PathGraph
18
19 from DataFlow::Configuration cfg, DataFlow::PathNode source, DataFlow::PathNode sink
20 where
21   (
22     cfg instanceof SqlInjection::Configuration or
23     cfg instanceof NosqlInjection::Configuration
24   ) and
25     cfg.hasFlowPath(source, sink)
26   select sink.getNode(), source, sink, "This query depends on @{$.}, source.getNode(),
27   "a user-provided value"
```

CodeQL: Pull-Request Integration

Update server.ts #1

Open wulfand wants to merge 1 commit into main from wulfand-patch-search

Conversation 0 Commits 1 Checks 3 Files changed 1

+10 -0

Changes from all commits ▾ File filter ▾ Conversations ▾ Jump to ▾ Dismiss ▾

server.ts

```
@@ -371,6 +371,16 @@
371   371     app.get('/api/Deliveries/:id', delivery.getDeliveryMethod())
372   372     // vuln-code-snippet end changeProductChallenge
373   373
374 + 374     + app.get('/api/search', function (req, res, context) {
375 + 375       var search_name = req.query.search;
376 + 376       if (search_name !== null) {
377 + 377         models.sequelize.query(`SELECT * FROM Products WHERE name LIKE '%${search_name}%' ORDER BY name`)
378 + 378       .then(([ products ]) => {
379 + 379         res.json(utils.queryResultToJson(products))
380 + 380       })
381 + 381     })
382 + 382   }
383 + 383 });

374 384   /* Verify the 2FA Token */
375 385   app.post('/rest/2fa/verify',
376 386     new RateLimit({ windowMs: 5 * 60 * 1000, max: 100 }),
```

Check failure on line 374 in server.ts

Code scanning

Missing rate limiting

This route handler performs a database access, but is not rate-limited.

Show more details

Dismiss ▾

375 + var search_name = req.query.search;
376 +
377 + if (search_name !== null) {
378 + models.sequelize.query(`SELECT * FROM Products WHERE name LIKE '%\${search_name}%' ORDER BY name`)

Check failure on line 378 in server.ts

Code scanning

Database query built from user-controlled sources

This query depends on a user-provided value.

Show more details

Show paths

Dismiss ▾

379 + .then(([products]) => {
380 + res.json(utils.queryResultToJson(products))
381 + })
382 + }
383 + });

374 384 /* Verify the 2FA Token */
375 385 app.post('/rest/2fa/verify',
376 386 new RateLimit({ windowMs: 5 * 60 * 1000, max: 100 }),

Database query built from user-controlled sources

Building a database query from user-controlled sources is vulnerable to insertion of malicious code by the user.

Open Error CWE-89 security

Branch: refs/pull/1/merge

Dismiss ▾

server.ts

```
375 var search_name = req.query.search;
376
377 if (search_name !== null) {
378   models.sequelize.query(`SELECT * FROM Products WHERE name LIKE '%${search_name}%' ORDER BY name`)
```

This query depends on a user-provided value.

CodeQL Show paths

379 .then(([products]) => {
380 res.json(utils.queryResultToJson(products))
381 })

Tool Rule ID Query

CodeQL js/sql-injection View source

If a database query (such as a SQL or NoSQL query) is built from user-provided data without sufficient sanitization, a malicious user may be able to run malicious database queries.

Show more ▾

First detected in commit 95fe43a 44 seconds ago

Merge 906306f into d45d3f0

server.ts#L378 on branch refs/pull/1/merge

Verified 95fe43a

CodeQL: Config

```
# Initializes the CodeQL tools for scanning.
- name: Initialize CodeQL
  uses: github/codeql-action/init@v1
  with:
    languages: ${{ matrix.language }}
    # If you wish to specify custom queries, you can do so here or in a config file.
    # By default, queries listed here will override any specified in a config file.
    # Prefix the list here with "+" to use these queries and those in the config file.
    # queries: ./path/to/local/query, your-org/your-repo/queries@main
    queries: security-extended
```

CodeQL: Config

```
# Initializes the CodeQL tools for scanning.  
- name: Initialize CodeQL  
  uses: github/codeql-action/init@v1  
  with:  
    languages: ${{ matrix.language }}  
  # If you wish to specify custom queries, you can do so here or in a config file.  
  # By default, queries listed here will override any specified in a config file.  
  # Prefix the list here with "+" to use these queries and those in the config file.  
  # queries: ./path/to/local/query, your-org/your-repo/queries@main  
  # queries: security-extended  
  config-file: ./github/codeql-config.yml
```

[Raw](#)

```
codeql-config.yml
```

```
1  name: "Custom CodeQL Configuration"  
2  
3  # Disabling the default queries is critical part of this.  
4  # Nothing will be disabled if the default queries still run.  
5  # - https://docs.github.com/en/code-security/secure-coding/configuring-code-scanning#disabling-the-default-queries  
6  disable-default-queries: true  
7  
8  queries:  
9    # Point to the custom query suite that needs to be used  
10   - uses: ./github/codeql/custom-javascript.qls  
11  
12 # Ignore some of the paths as these are some open source components.  
13 # This only works for Interpreted Languages (JavaScript / TypeScript / Python)  
14 # - https://docs.github.com/en/code-security/secure-coding/automatically-scanning-your-code-for-vulnerabilities-and-errors/conf  
15 paths-ignore:  
16   - '**/node_modules'  
17   # Ignoring our test folders  
18   - '**/test'
```

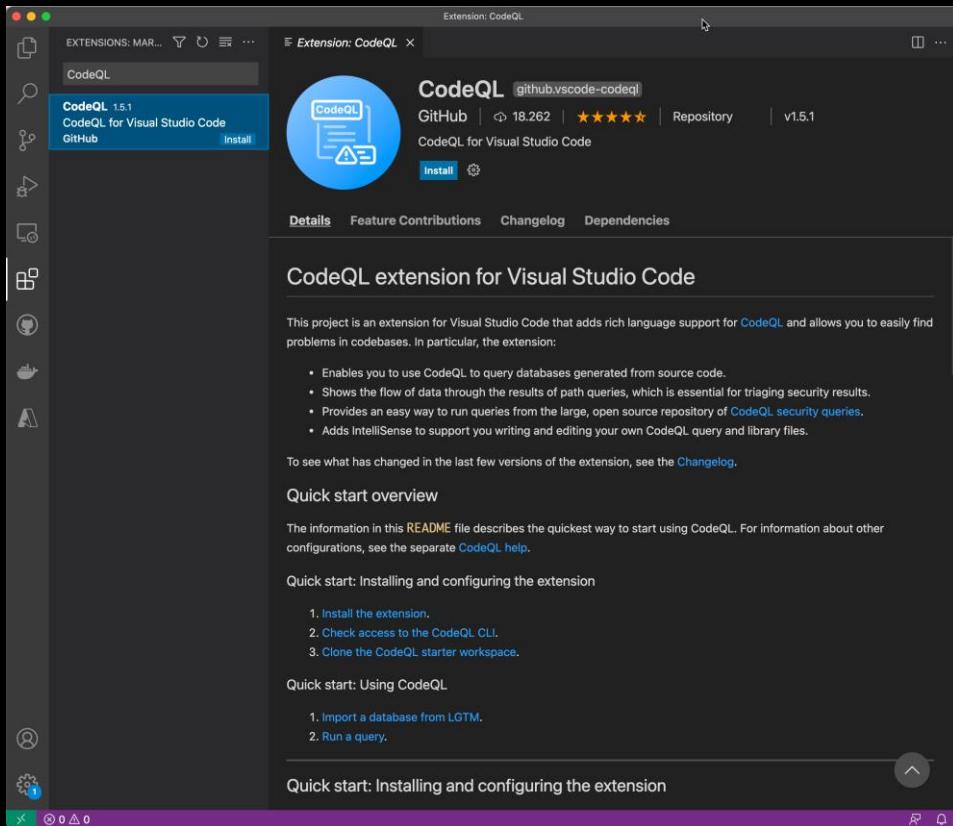
CodeQL: Config

```
custom-javascript.qls
Raw

1 # Documentation
2 # - https://codeql.github.com/docs/codeql-cli/creating-codeql-query-suites/
3 # - https://github.com/github/codeql/actions/workflows/query-list.yml
4 # - https://github.com/github/codeql/blob/main/misc/suite-helpers/security-extended-selectors.yml
5 # - https://codeql.github.com/docs/codeql-cli/creating-codeql-query-suites/#examples
6
7 - description: "Custom JavaScript Suite"
8
9 # Import/Extend from an existing CodeQL suite
10 - import: codeql-suites/javascript-security-extended.qls
11   from: codeql-javascript
12
13
14 # Exclude queries that are not relevant for my organisation or have a
15 # higher False Positive rate than we are confident with.
16 - exclude:
17   id:
18     # https://github.com/github/codeql/blob/main/javascript/ql/src/Security/CWE-770/MissingRateLimiting.ql
19     - js/missing-rate-limiting
```

Eigene CodeQL Queries

<https://github.com/wulfland/CodeQLDemo>



The screenshot shows the VS Code interface with the "CodeQL" extension installed. The "EXPLORER" sidebar shows a file tree with several CodeQL query files (Query1.ql, Query2.ql, etc.) and a "README.md" file. The "QUERIES" tab is selected. The "CodeQL Query Results" panel on the right shows the results of a query named "Query2.ql". The results table lists 737 results, with columns for the query number, the dollarCall function, and the dollarArg variable. The "From a URL (as a zip file)" option in the "Add a CodeQL database" dialog is highlighted with a red box. The URL for the download is also shown: https://github.com/githubsatellit/eworkshops/codeql/releases/download/v1.0/esbena_bootstrap-pre-27047_javascript.zip.

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>)

```
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 the GitHub Action page for 'OWASP ZAP Full Scan'. At the top, there's a blue circular icon with a white lightning bolt, the text 'GitHub Action', the name 'OWASP ZAP Full Scan', the version 'v0.2.0' (with a 'Latest version' button), and a green 'Use latest version' button. To the right, there's a 'Verified creator' badge, a 'Stars' section (97 stars), a 'Contributors' section with four icons, and a 'Categories' section with 'Security' and 'Testing'. Below these are 'Links' for 'Open issues' (2), 'Pull requests' (1), and 'Report abuse'. The main content area is titled 'ZAP Action Full Scan' and describes the action as running the OWASP ZAP Full Scan to perform DAST. It includes a warning about performing attacks on targets and a note about alert submission via GitHub issues. There are three input fields: 'target' (required URL), 'docker_name' (optional docker file name), and 'rules_file_name' (optional relative path to rules file). A code block at the bottom shows sample rules file content:

```
10011 IGNORE (Cookie Without Secure Flag)  
10015 IGNORE (Incomplete or No Cache-control and Pragma HTTP Header Set)
```

Infrastructure Scanning

Container Vulnerability Analysis (CVA)

Container Security Analysis (CSA)

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

Andere Infrastruktur:

- Azure Policy
<https://docs.microsoft.com/de-de/azure/governance/policy/>
- Checkov
<https://www.aquasec.com/products/container-security/>
- OpenVAS
<https://www.openvas.org/>

```
name: build
on:
  push:
    branches:
      - master
  pull_request:
jobs:
  build:
    name: Build
    runs-on: ubuntu-18.04
    steps:
      - name: Checkout code
        uses: actions/checkout@v2

      - name: Run Trivy vulnerability scanner in repo mode
        uses: aquasecurity/trivy-action@master
        with:
          scan-type: 'fs'
          ignore-unfixed: true
          format: 'template'
          template: '@/contrib/sarif.tpl'
          output: 'trivy-results.sarif'
          severity: 'CRITICAL'

      - name: Upload Trivy scan results to GitHub Security tab
        uses: github/codeql-action/upload-sarif@v1
        with:
          sarif_file: 'trivy-results.sarif'
```

DefectDojo

<https://github.com/DefectDojo/django-DefectDojo>

Demo: demo.defectdojo.org

Log in: admin / defectdojo@demo#appsec

The screenshot shows the DefectDojo web interface for the project 'Bodgeit'. The top navigation bar includes links for Overview, Metrics, Engagements (38), Findings (204), Endpoints (27), Benchmarks, and Settings. A search bar and a notification bell icon (227) are also present.

Description: Bodgeit
Bodgeit
• Easy to install - just requires java and a servlet engine, e.g. Tomcat
• Self contained (no additional dependencies other than 2 in the above line)
• Easy to change on the fly - all the functionality is implemented in JSPs, so no IDE required* Cross platform
• Open source* No separate db to install and configure - it uses an 'in memory' db that is automatically (re)initialized on start up
• More testing

Metrics:
CRITICAL: 0
HIGH: 33
MEDIUM: 80
LOW: 85
INFORMATIONAL: 6
TOTAL: 204

Technologies: Apache 2.0 v.1

Regulations (1): GDPR EU & EU Data Extra-Territorial Applicability Privacy

Metadata:

Business Criticality	High
Product Type	Research and Development
Platform	Web
Lifecycle	Construction
Origin	Third Party Library
User Records	1,000
Revenue	50,000.00

Languages:

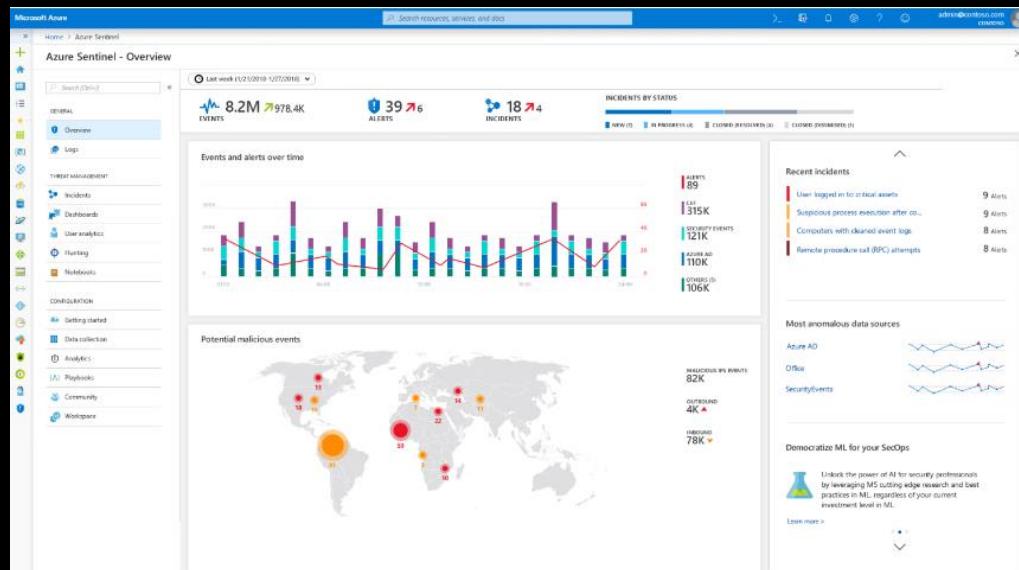
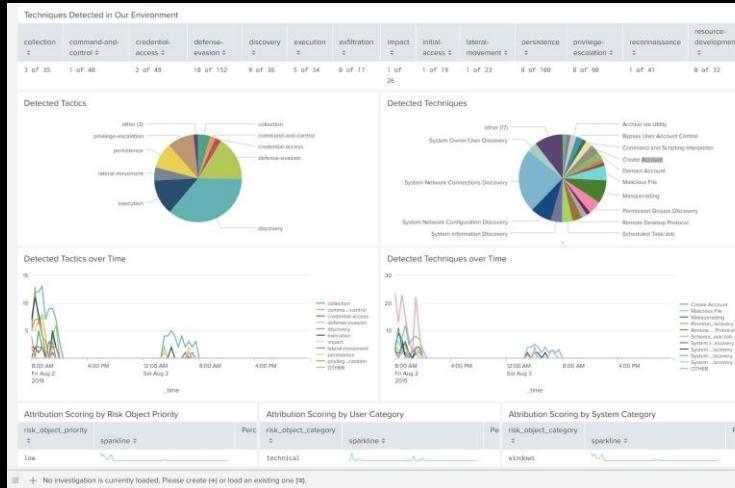
XML	~10%
JavaScript	~30%
Java	~60%

Total Files: 55 Lines of Code: 10,400

Security Information & Event Management (SIEM)

Beispiel:

- Azure Sentinel
- Splunk
- Zentrales Logging
- Multicloud/Hybrid
- Anomalieerkennung (ML)
- Echtezeit Warnungen



6 tips to integrate security into your DevOps practices

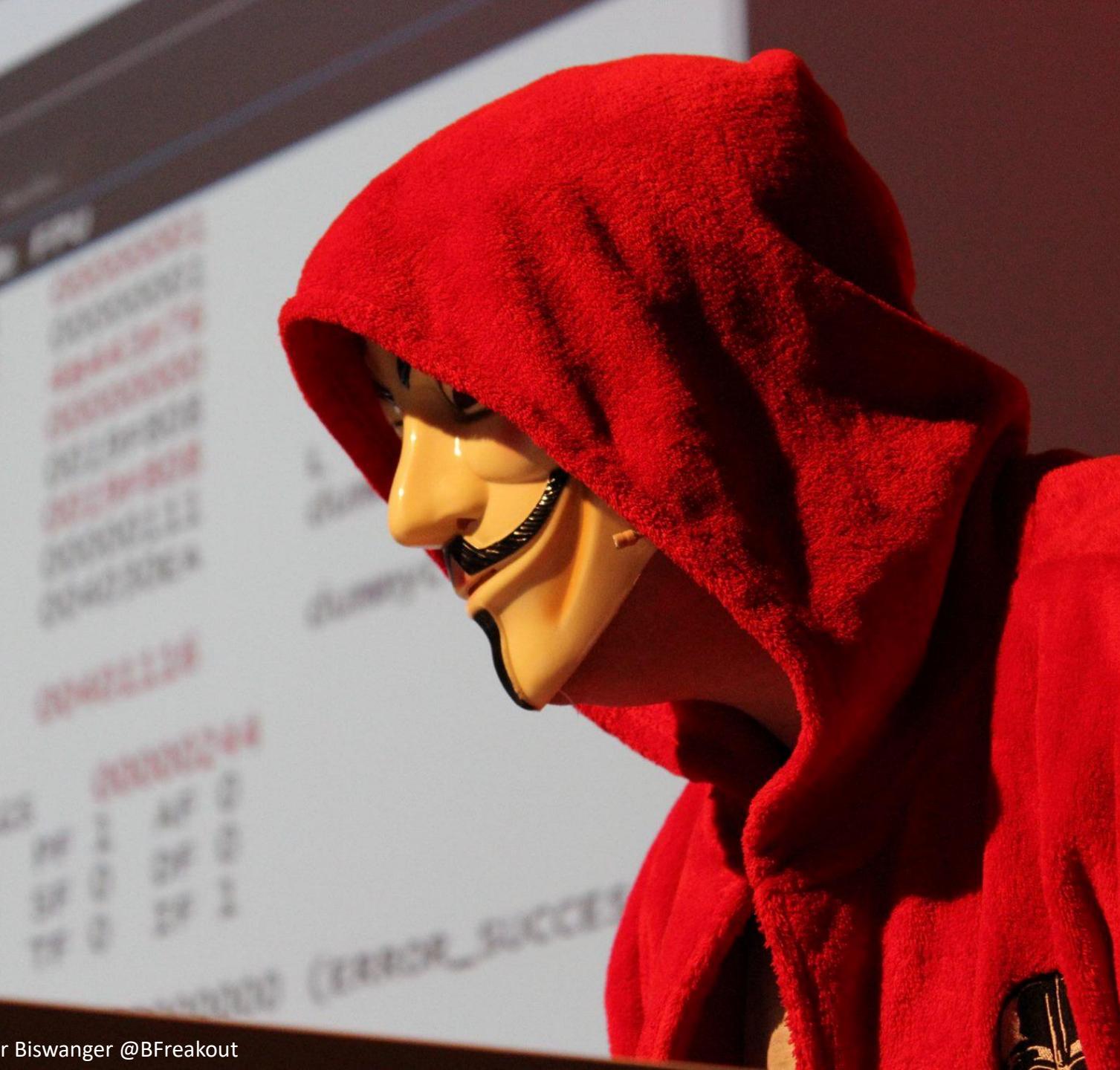
1. Build a **security-first culture across the business**
2. Integrate security in the early stages of the development lifecycle
3. Monitor and observe continuously with purpose
4. Embrace everything-as-code
5. Realize compliancy with policy automation
6. Secure and visualize your software supply chain



<https://azure.microsoft.com/en-us/resources/6-tips-to-integrate-security-into-your-devops-practices/>

Wenn wir über Sicherheit sprechen...

- „~~Wir hatten bisher noch nie einen Sicherheitsvorfall~~“
- „~~Unsere Entwickler sind Experten~~“
- „~~Unsere Firma ist zu unwichtig, um angegriffen zu werden~~“
- „~~Wir haben uns schon gut abgesichert~~“
- „~~Unsere Firewalls sind sicher~~“
- „Wir müssen was tun!“
- „Sicherheit hat oberste Priorität!“
- „Wenn wir einen Sicherheitsvorfall hatten, dann ist es zu spät!“



Hackers in movies
vs





Vielen Dank

Blog: <https://writeabout.net>

Twitter: @mike_kaufmann

GitHub: @wulfland

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