



Greenbone

Test Automation & QA Strategy for Short Iterations

Tim Steffens
Software / QA Engineer

these slides are licensed under a [Creative Commons Attribution 4.0 International license](https://creativecommons.org/licenses/by/4.0/)

ABOUT ME

- timsteffens.de
- Father & husband
- ~20 years Software Engineering
- Quest
 - ✓ how to build software that "works well"?
- Fan of
 - ✓ clean code, TDD & functional programming
 - ✓ music (Metal)
 - ✓ outdoors & hiking



ABOUT GREENBONE

What sets us apart

Innovation: Leading in cybersecurity with scalable, cutting-edge solutions that identify and mitigate vulnerabilities.

Value Driven Culture: Work environment based on sincere appreciation, honoring individuality

Employee Empowerment: Recognizing our employees as the primary drivers of our success and committing to nurture growth through continuous learning and comprehensive benefits



ABOUT YOU!

- Software Engineer?
 - Testing Engineer?
 - Architect?
 - Project Lead?
 - None of those?
-
- Worked with test automation?



PLEASE INTERRUPT!



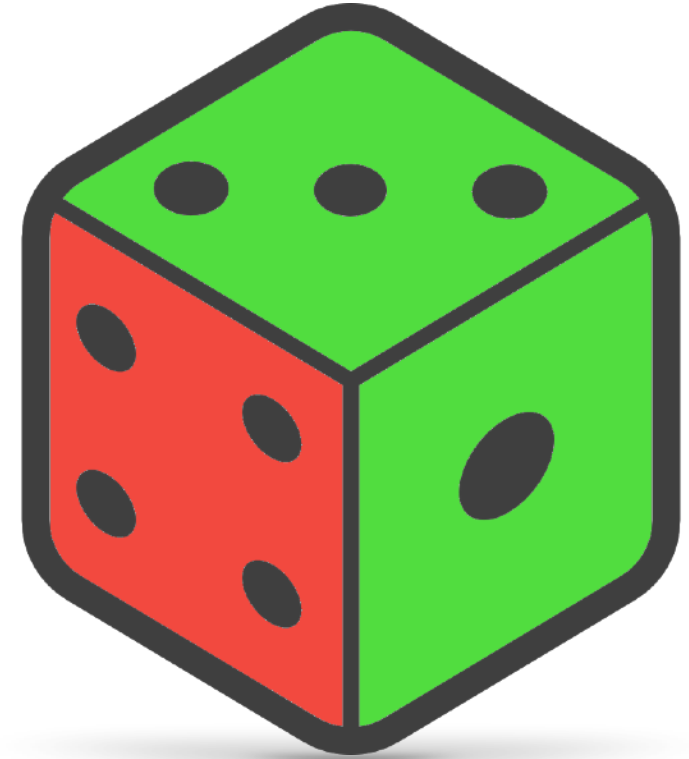
AGENDA

- Setting the Scene
- QA Strategy
- Implementation, Challenges & Learnings
- Outlook



DICE

- Recurring theme in Test Automation: **"the dice"**
 - ✓ AKA **"flaky tests"**
 - ✓ Two runs green, next red - despite no code change ...
- **Who** has already experienced **flaky tests**?

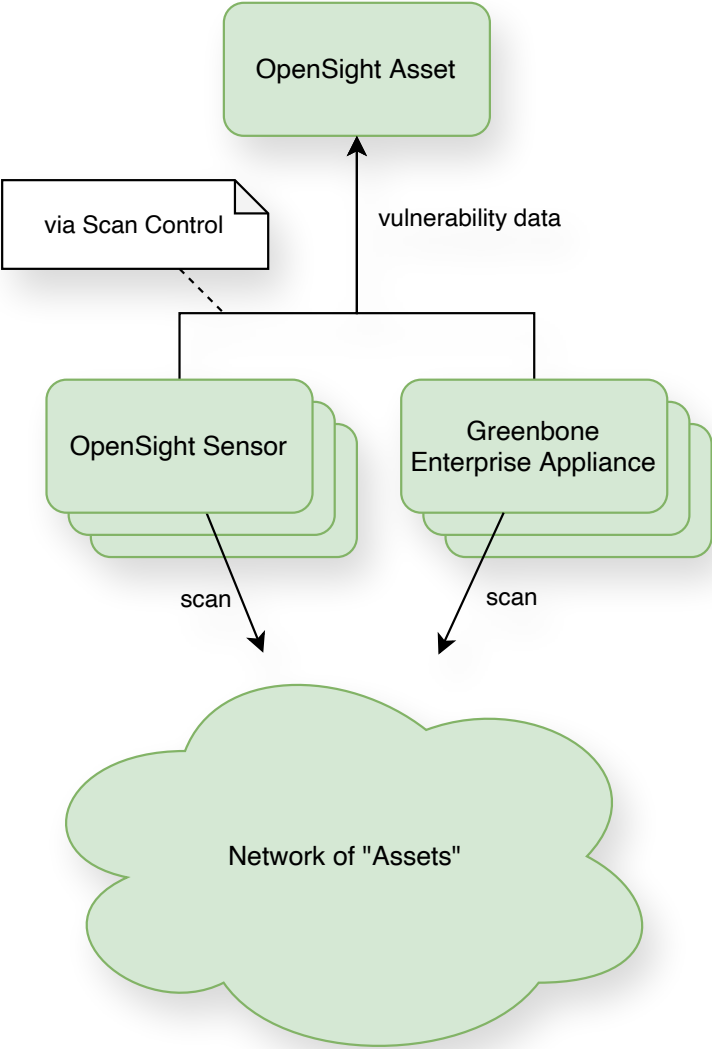


DISCLAIMER

- Out of scope:
 - NFA: Security Testing, Load Testing
 - Code Coverage
 - QA development practices (e.g. Pair Programming, Code Reviews)
 - Frontend Unit Tests



SETTING: PRODUCT



Greenbone

Analysis

Dashboard

Assets

Vulnerabilities

Import

Dashboard

Match All

Number of assets

2,286

Vulnerability trends

2,000

1,800

1,600

1,400

1,200

1,000

800

600

400

200

0

Apr 10Apr 11Apr 12Apr 13Apr 14Apr 15Apr 16Apr 17Apr 18Apr 19Apr 20Apr 21Apr 22Apr 23

Critical

High

Medium

Critical

Top 10 vulnerabilities

Name	Assets
mysql / mariadb default credentials (m...	68
distcc rce vulnerability (cve-2004-2687)	66
postgresql default credentials (postgre...	59
vnc brute force login	52
twiki xss and command execution vuln...	44
operating system (os) end of life (eol)	

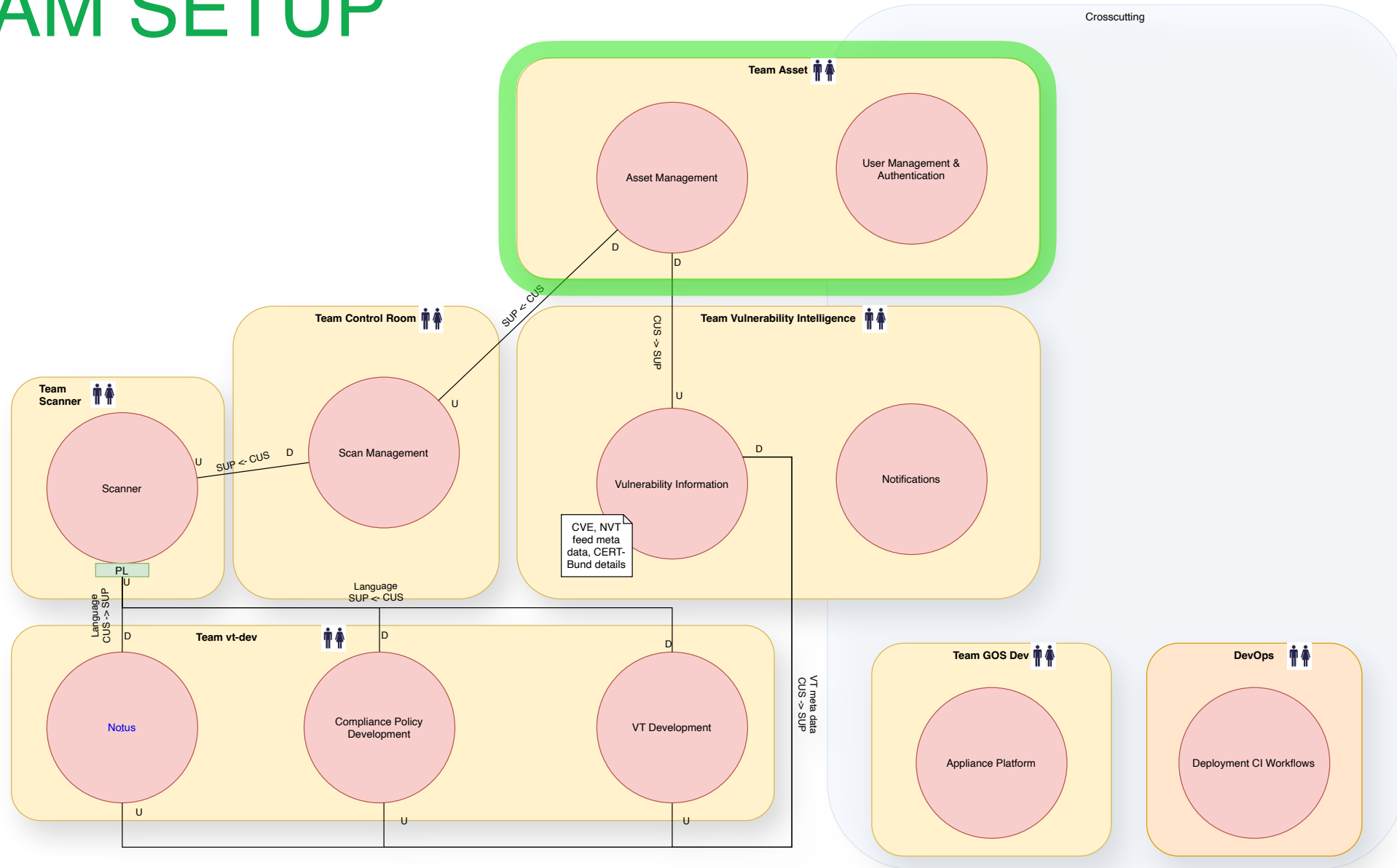
High

Top 10 vulnerab

Name
vsftpd compromised so
distcc detection
the rlogin service is runn
phpinfo() output reporti
ssh brute force logins w
sed setups vul

Greenbone

TEAM SETUP



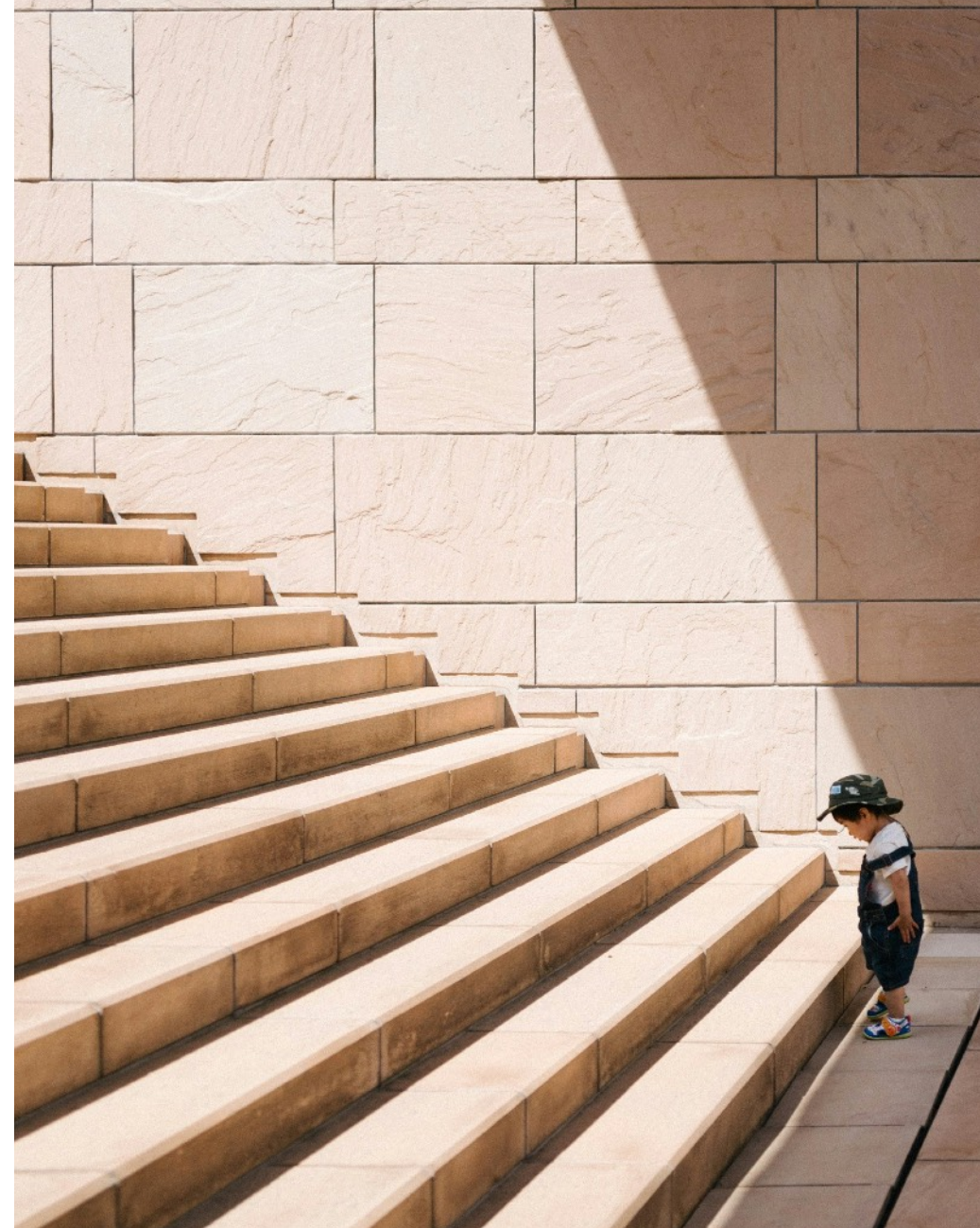
SETTING: AGILE

- Agile Approach
 - ✓ **Scrum**
 - ✓ short iterations: **2 week sprints**
 - ✓ **multiple increments** released per sprint
 - ✓ optimized for **fast feedback**



CHALLENGES

- Too few resources to **test** everything **manually**
- Especially **regression**
 - ✓ needs to be repeated **several times a week**
 - ✓ should as **reliable** as possible
 - ✓ reasonable **effort**



THE ANSWER: TEST AUTOMATION

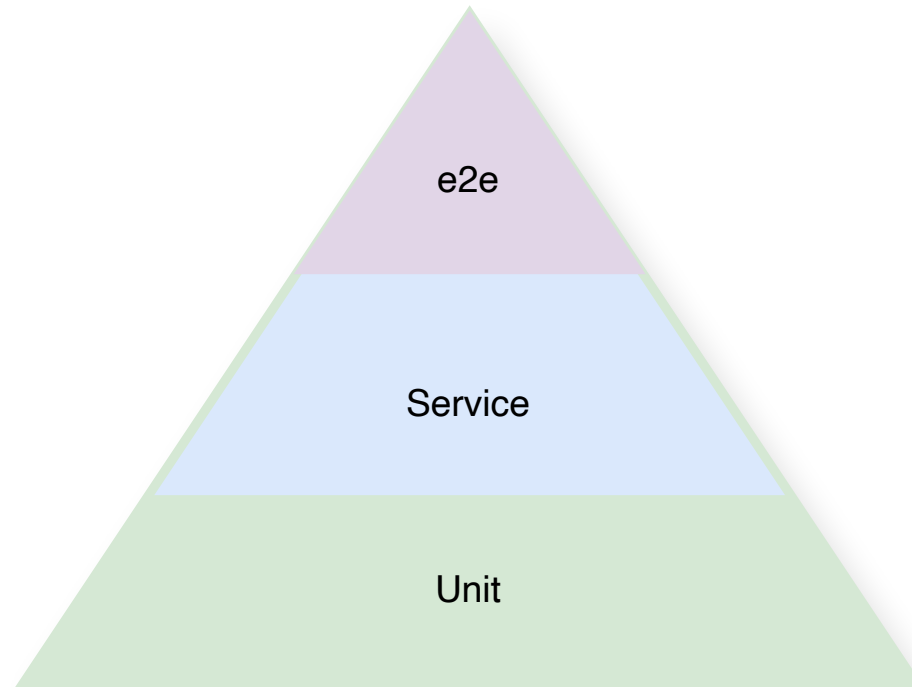


QA STRATEGY

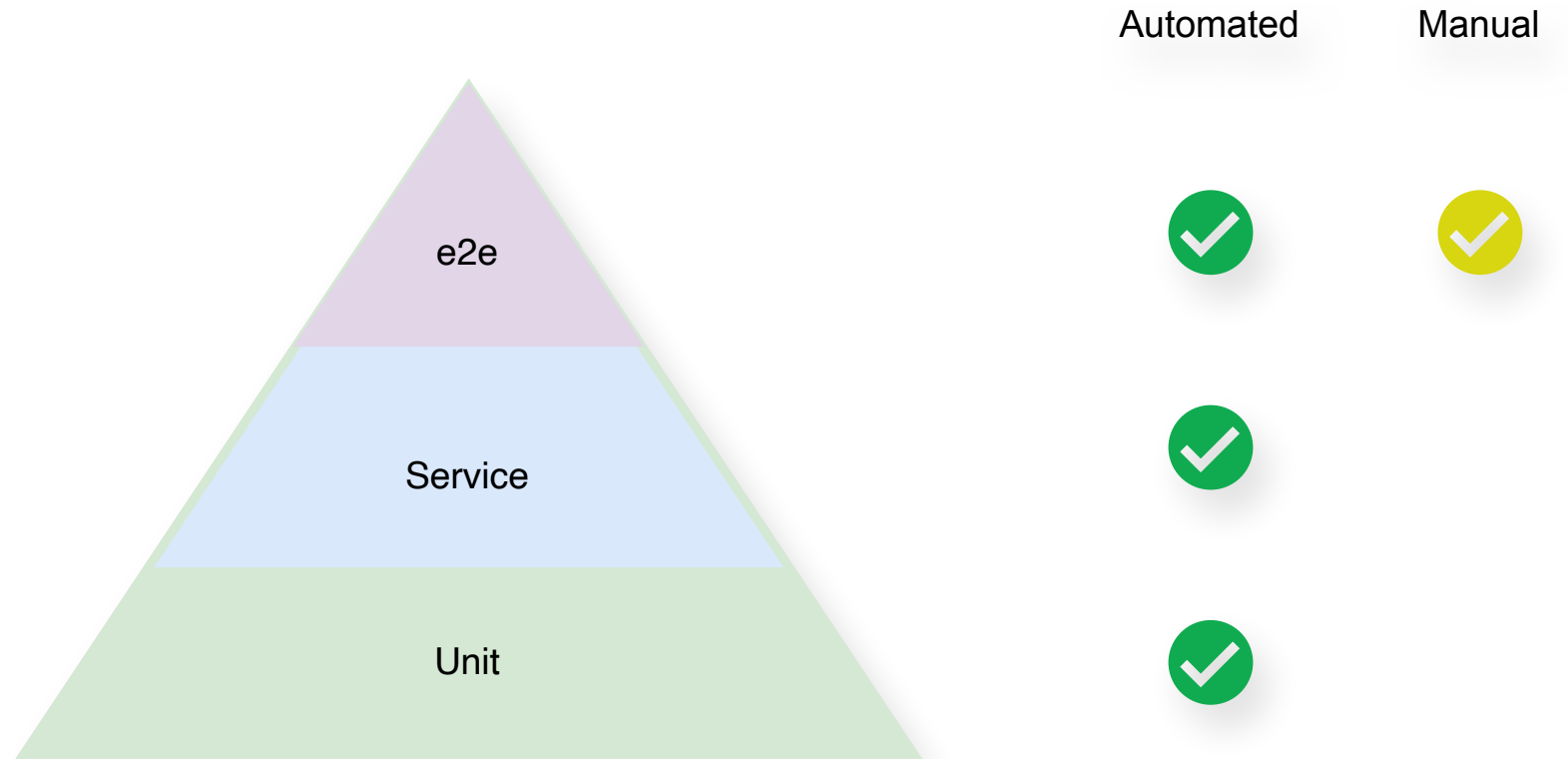
- **automate as much as possible**
- test things on the **right level**
- **code reviews** are mandatory
- in case **manual tests** are needed for regression they must be **well-documented** and easily **reproducible**
- for each **bug** that is found a **test** must be created
- (...)



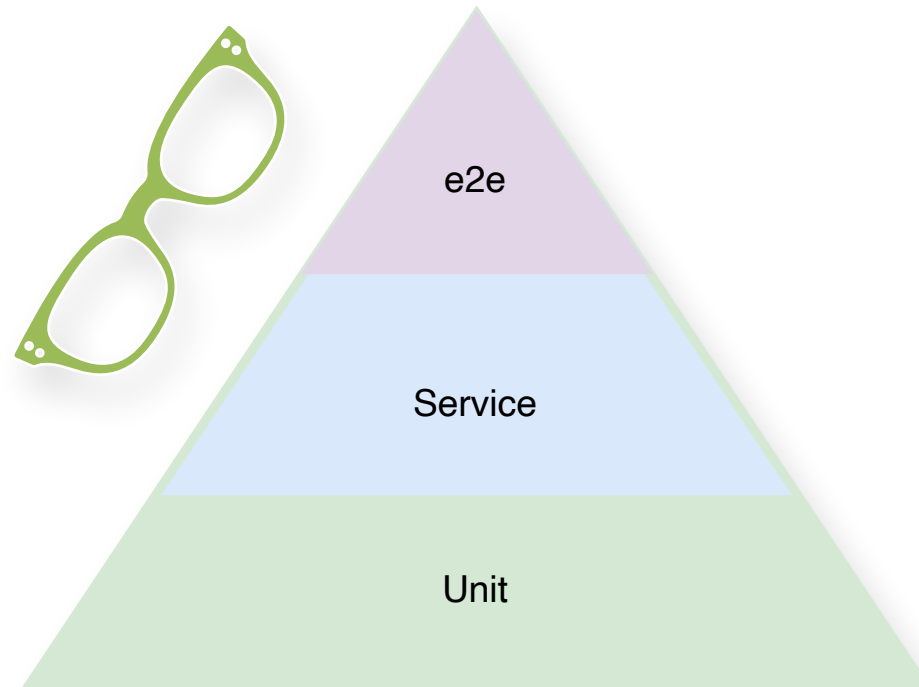
TEST LEVELS



TEST LEVELS



TEST LEVELS



TEST MATRIX

Use Case	Test Case	Unit Test	BE Service Test	FE Service Test	Full Service Test	e2e Test	manual test
Edit Appliance	edit appliance with missing fingerprint	+ for appliance controller + for appliance service	+ add test case (example test case for failure)	+ add test case (test case stops after receiving the fingerprint)	-	-	-
	edit appliance with correct fingerprint	+ for appliance controller + for appliance service	+ add test case (example test case for success)	+ add test case	-	-	-
	edit appliance with incorrect fingerprint	+ for appliance controller + for appliance service	-	+ add test case (test case stops after receiving the error)	-	-	-
	get appliance where the host key has changed and the fingerprint is no longer valid	+ for appliance controller + for appliance service + for import service(s)	(too difficult to simulate)	+ add test case	-	-	test manually

...

QA STRATEGY

Implementation

- **QA Strategy** workshop per team
- Dedicated **QA Engineer** per team
- **QA CoP**



E2E TESTS



E2E TESTS



E2E TESTS



```
1  import "cypress-wait-until"
2
3  >> describe('create a user and log in', () :void => {
4      after(() :void => {
5          cy.programmaticLogout()
6          cy.deleteUser(Cypress.env("new_user"))
7      })
8
9
10 > it('Creating a user on operating system level should work', () :void => {
11     cy.createUser(Cypress.env("new_user"), Cypress.env("temp_password"))
12 })
13
14
15 > it('Login via UI should work and lead to the landing page of OpenSight Asset', () :void => {
16     cy.visit(Cypress.env('redirect_uri'))
17     cy.waitForStableDOM()
18
19     cy.loginViaUI(Cypress.env("new_user"), Cypress.env("temp_password"))
20     cy.updatePassword(Cypress.env("password"))
21     .then(() :void => {
22         // check landing page for validation
23         cy.contains("h2", "Dashboard")
24     })
25 })
26
```

E2E TESTS



```
1  name: e2e tests dev environment
2
3  on:
4    schedule:
5      - cron: '30 00 * * 1-5' # Mon-Fri at 01:30 UTC
6    workflow_dispatch:
7    pull_request:
8      branches: [ main ]
9
10 jobs:
11   e2e-tests:
12     name: Run e2e tests against the dev system
13     runs-on:
14       - self-hosted-generic
15       - self-hosted
16     container:
17       image: ghcr.io/greenbone/opensight-e2e:latest
18       options: --init
19     steps:
20       - name: Checkout
21         uses: actions/checkout@v4
22       - name: Install dependencies
23         run: chown -R root ./ && npm ci
24       - name: Run tests
25         env:
26           CYPRESS_ssh_pass: ${ secrets.SSH_PASSWORD }}
27           CYPRESS_password: ${ secrets.PASSWORD }}
28           CYPRESS_appliance_password: ${ secrets.APPLIANCE_PASSWORD }}
29         run: make start-e2e-tests ENV=dev
30       - name: Upload report folder
31         if: always()
32         uses: actions/upload-artifact@v4
```

E2E TESTS



```
13 runs-on:
14   - self-hosted-generic
15   - self-hosted
16 container:
17   image: ghcr.io/greenbone/opensight-e2e:latest
18   options: --init
19 steps:
20   - name: Checkout
21     uses: actions/checkout@v4
22   - name: Install dependencies
23     run: chown -R root ./ && npm ci
24   - name: Run tests
25     env:
26       CYPRESS_ssh_pass: ${ secrets.SSH_PASSWORD }
27       CYPRESS_password: ${ secrets.PASSWORD }
28       CYPRESS_appliance_password: ${ secrets.APPLIANCE_PASSWORD }
29     run: make start-e2e-tests ENV=dev
30   - name: Upload report folder
31     if: always()
32     uses: actions/upload-artifact@v4
33     with:
34       name: e2e-test-dev-reports
35       path: cypress/reports/
36       retention-days: 4
37   - name: Upload screenshot folder
38     if: always()
39     uses: actions/upload-artifact@v4
40     with:
41       name: e2e-test-dev-screenshots
42       path: cypress/screenshots/
43       retention-days: 4
```

E2E TESTS

⇒ Specs

✓ 4 ✗ 1 ↺

login-logout-workflow cy.ts 00:41

```
11 - type 8YWByCsMI/RHaW9lD4rL2A==
12 get input[id='kc-login'][name='login']
13 - click
    (form sub) --submitting form--
    (page load) --page loaded--
    (new url) https://asset-dev.opensight-asset.greenbone.io/auth/realms/opensight/login-actions/authenticate?session_code=xs7QQY0fgG_5ljYkCBazxD90XIxpFx-B4dkEYmM-lMw&execution=b674ef6e-d0ca-458d-b11a-24441cbf180a&client_id=local-web&tab_id=bkBsny0JeoI
    (new url) https://asset-dev.opensight-asset.greenbone.io/auth/realms/opensight/login-actions/authenticate?execution=b674ef6e-d0ca-458d-b11a-24441cbf180a&client_id=local-web&tab_id=bkBsny0JeoI
14 document
15 log No changes detected for over interval 1400ms!
16 log Continuing with test...
17 wrap <document>
18 get input[id='password-new'] [name='password-new']
```

https://asset-dev.opensight-asset.greenbone.io/auth/realms/opensight/login-actions/authenticate?execution=l

Electron 118 1000x660 (79%)



OpenSight Asset

✗ Invalid user name or password.
Please correct your input and try again.

User name
9a05733a-e778-4d48-b6d1-551e567f2ecc

Password

Sign in

E2E TEST CHALLENGES

⚡ Asynchronism ⚡

- certain computations are done **asynchronously**
 - ➡ e.g. ui rendering, asynchronous messages
 - ➡ **time interval** before results are available **may vary** strongly depending on system and load

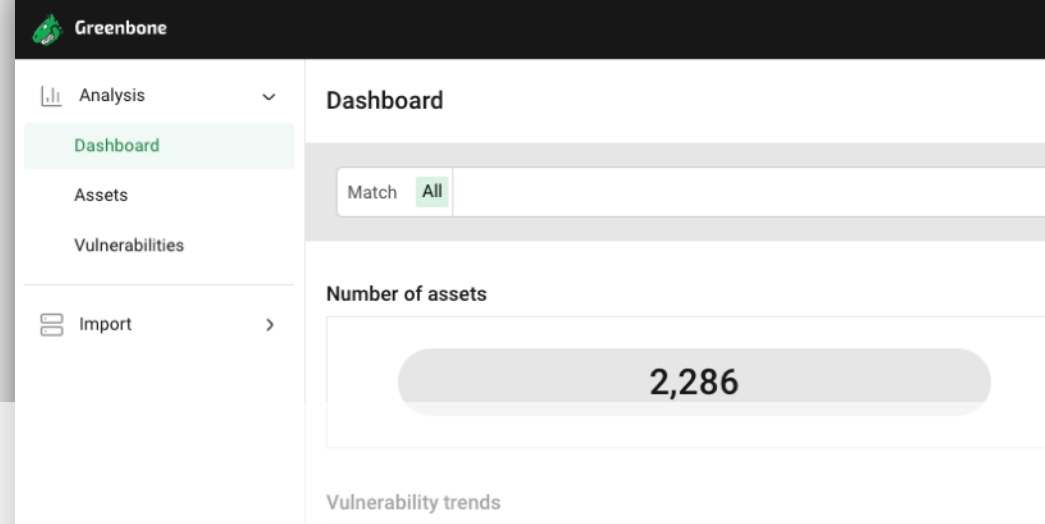


E2E TEST CHALLENGES

⚡ Asynchronism ⚡ 

- certain **computations** are done
 - ➔ e.g. ui rendering, asynchronous r
 - ➔ **time interval** before **results** are a
 - strongly depending on system an

- our approach @ 
 - ➔ **wait for** conditions with **timeout** before checking assertions
 - ➔ **cypress** has **built-in waits** with default timeouts!



```
18 > it('Creating a valid appliance should work and it should be displayed afterwards', () => {  
19     cy.enterPage("Import")  
20     cy.enterPage("Appliances")  
21     cy.get("span").contains("Add appliance").click()  
22     cy.get("form").find("input[name=name]").type(Cypress.env("appliance_name"))  
23     cy.get("form").find("input[name=gmpHost"]').type(Cypress.env("appliance_ip"))  
24     cy.get("form")  
25         .find("input[name=gmpUsername"])  
26         .type(Cypress.env("appliance_user"))  
    })
```

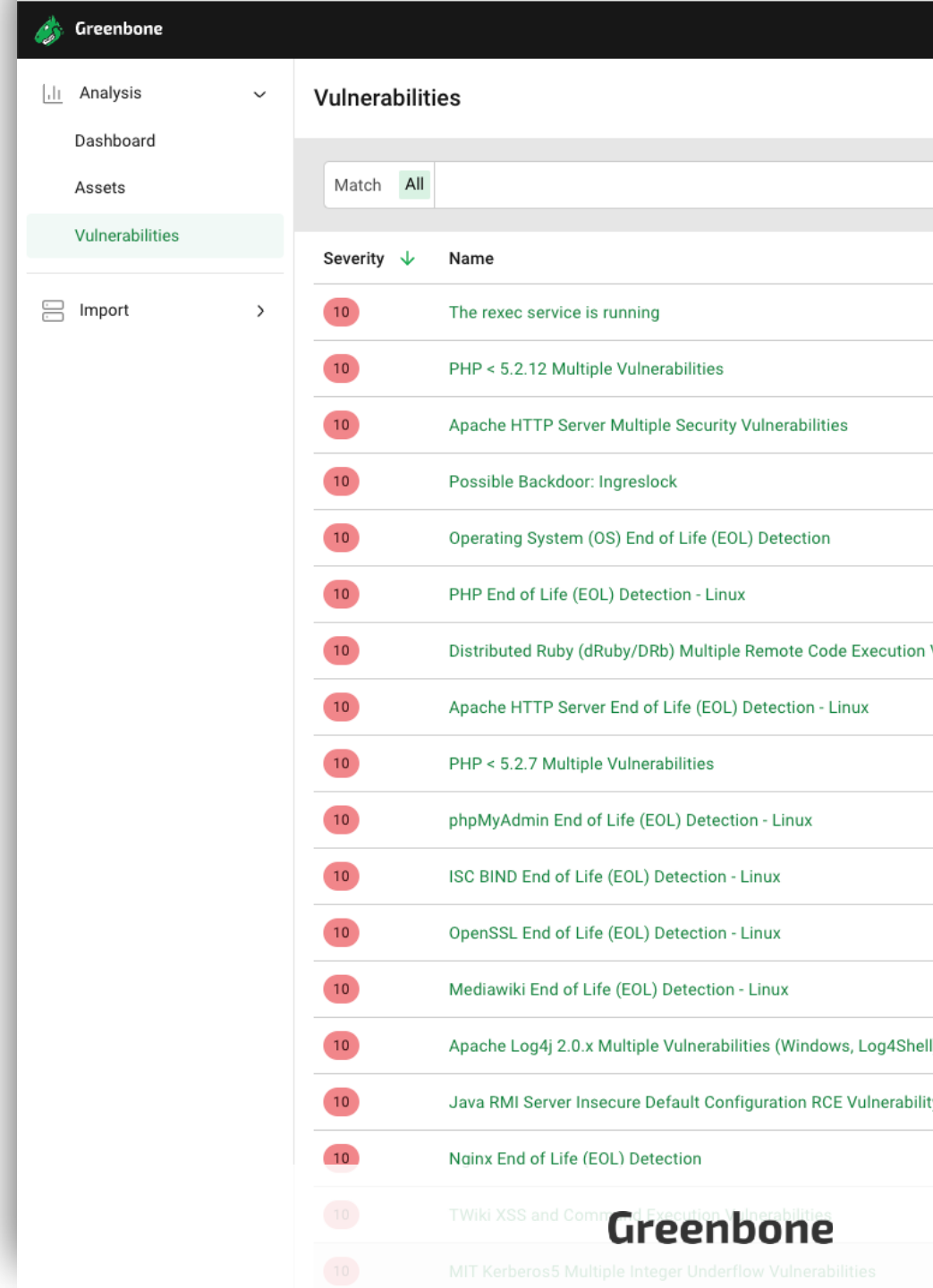
Critical Top 10 vulnerabilities		High Top 10 vulnerabilities	
Name	Assets	Name	Assets
mysql / mariadb default credentials (mysql...	68	vsftpd compromised sou	
distcc rce vulnerability (cve-2004-2687)	66	distcc detection	
postgresql default credentials (postgresq...	59	the login service is runni	
vnc brute force login	52	phpinfo() output reportin	
twiki xss and command execution vulnera...	44	ssh brute force logins wi	
operating system (os) end of life (eol) dete...	11	php-cgi-based setups vul	

Greenbone

E2E TEST CHALLENGES

↻ Changing Data ↻ 

- **data** imported from **real systems** changes throughout time
 - ➔ **how many** and **which** entries will be there?



The screenshot shows the Greenbone Vulnerability Scanner (GVM) interface. The left sidebar contains navigation links: Analysis, Dashboard, Assets, Vulnerabilities (selected), and Import. The main panel displays a table of vulnerabilities. The table has two columns: Severity and Name. All listed vulnerabilities have a severity of 10. The vulnerabilities include:

Severity	Name
10	The rexec service is running
10	PHP < 5.2.12 Multiple Vulnerabilities
10	Apache HTTP Server Multiple Security Vulnerabilities
10	Possible Backdoor: Ingreslock
10	Operating System (OS) End of Life (EOL) Detection
10	PHP End of Life (EOL) Detection - Linux
10	Distributed Ruby (dRuby/DRb) Multiple Remote Code Execution
10	Apache HTTP Server End of Life (EOL) Detection - Linux
10	PHP < 5.2.7 Multiple Vulnerabilities
10	phpMyAdmin End of Life (EOL) Detection - Linux
10	ISC BIND End of Life (EOL) Detection - Linux
10	OpenSSL End of Life (EOL) Detection - Linux
10	Mediawiki End of Life (EOL) Detection - Linux
10	Apache Log4j 2.0.x Multiple Vulnerabilities (Windows, Log4Shell)
10	Java RMI Server Insecure Default Configuration RCE Vulnerability
10	Nginx End of Life (EOL) Detection
10	Twig XSS and Command Execution Vulnerabilities
10	MIT Kerberos5 Multiple Integer Underflow Vulnerabilities

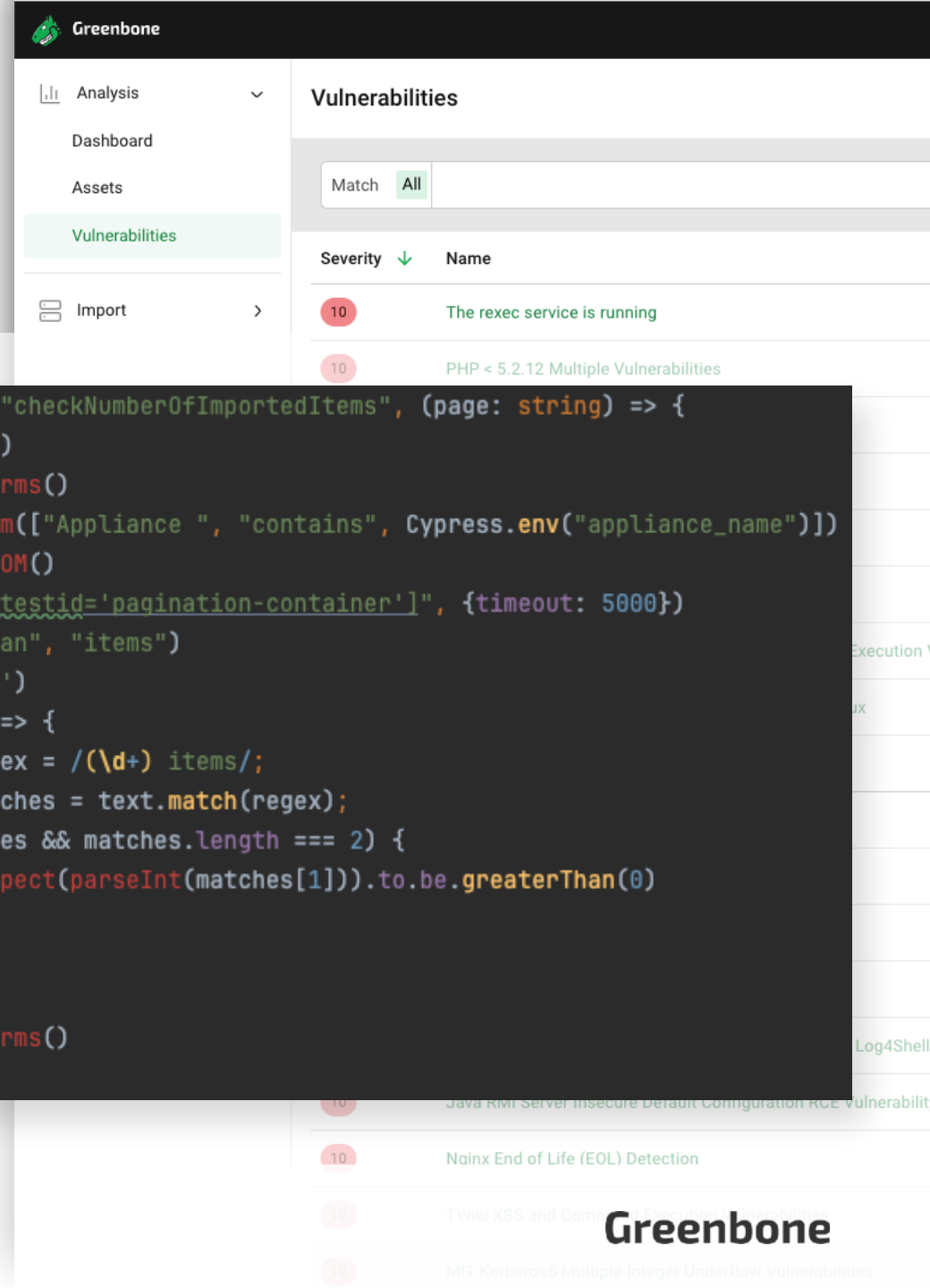
The Greenbone logo is visible in the bottom right corner of the interface.

E2E TEST CHALLENGES

↻ Changing Data ↻ 

- data imported from **real systems** throughout time
 - ➔ how many and **which** entries will be
- our approach @ 
 - ➔ **assert** only what **you can know**
 - ➔ validate **structure**

```
35 Cypress.Commands.add("checkNumberOfImportedItems", (page: string) => {
36   cy.enterPage(page)
37   cy.deleteFilterTerms()
38   cy.enterFilterTerm(["Appliance ", "contains", Cypress.env("appliance_name")])
39   cy.waitForStableDOM()
40   cy.get("div[data-testid='pagination-container']", {timeout: 5000})
41     .contains("span", "items")
42     .invoke('text')
43     .then((text) => {
44       const regex = /(\d+) items/;
45       const matches = text.match(regex);
46       if (matches && matches.length === 2) {
47         cy.expect(parseInt(matches[1])).to.be.greaterThan(0)
48       }
49     })
50
51   cy.deleteFilterTerms()
52 })
```



E2E TEST CHALLENGES

⚡ Interference in Parallel Runs ⚡



- **same resources** / entities / identifiers used for **each run**
 - ➡ e.g. same user
 - ➡ **test data** can be **influenced by another test run**

Europe/Berlin | 🇬🇧 | ⚙️ | John Doe

+ Add appliance

Version ↑↓	Import status ↑↓	Last import ↑↓	Next import ↑↓	Action
22.5	Done	Apr 19, 2024, 12:59:05 AM	Apr 20, 2024, 12:59:00 AM	⋮
22.5	Done	Apr 19, 2024, 06:30:03 AM	Apr 20, 2024, 06:30:00 AM	⋮
22.5	Done	Apr 19, 2024, 06:31:02 AM		⋮

< 1 >

E2E TEST CHALLENGES

⚡ Interference in Parallel Runs ⚡



- **same resources** / entities / identifiers used for **each run**

➡ e.g. same user

➡ **test data** can be **influenced** by another test

- our approach @ 🦊

➡ **generate** entities with random identifiers

➡ and **delete** them after **each run**

Europe/Berlin | 🇬🇧 | ⚙️ | John Doe

+ Add appliance

Version	Import status	Last import	Next import	Action
22.5	Done	Apr 19, 2024, 12:59:05 AM	Apr 20, 2024, 12:59:00 AM	⋮
22.5	Done	Apr 19, 2024, 06:30:03 AM	Apr 20, 2024, 06:30:00 AM	⋮
22.5	Done	Apr 19, 2024, 06:31:02 AM		⋮

< 1 >

```
1 #!/bin/bash
2
3 random_username=$(uuidgen)
4 export CYPRESS_new_user=$random_username
5
6 echo CYPRESS_new_user="$CYPRESS_new_user"
7
8 temp_password=$(head -c 16 /dev/urandom | base64)
9 export CYPRESS_temp_password=$temp_password
```

```
2
3 describe('create a user and log in', () : void => {
4   after(() : void => {
5     cy.programmaticLogout()
6     cy.deleteUser(Cypress.env("new_user"))
7   })
8
9   it('Creating a user on operating system level should work', () : void => {
10    cy.createUser(Cypress.env("new_user"), Cypress.env("temp_password"))
11  })
12 }
```

E2E TEST CHALLENGES

⚡ Interference in Parallel Runs ⚡



- **same resources** / entities / identifiers used for **each run**

➡ e.g. same user

➡ **test data** can be **influenced by another test run**

- our approach @

➡ **generate** entities with random identifiers

➡ and **delete** them after **each run**

Version	Import status	Last import	Next import	Action
22.5	Done	Apr 19, 2024, 12:59:05 AM	Apr 20, 2024, 12:59:00 AM	⋮
22.5	Done	Apr 19, 2024, 06:30:03 AM	Apr 20, 2024, 06:30:00 AM	⋮
22.5	Done	Apr 19, 2024, 06:31:02 AM		⋮

- ⚡ **unique resources** ⚡

➡ e.g. single machine

- possible approaches

➡ provide set and **lend instances to test runs**

➡ write test that **adapt to situation** (e.g. start import if none is running)

E2E TESTS: WRAP-UP

- asynchronism: use **wait for** constructs
- no control of data: **validate structure**, assert what you can know
- parallel runs: **generate entities** with random identifiers, delete them after each run
- keep **test durations** low
 - ➔ e.g. use **API calls** for test setup
- **expect efforts** for maintenance



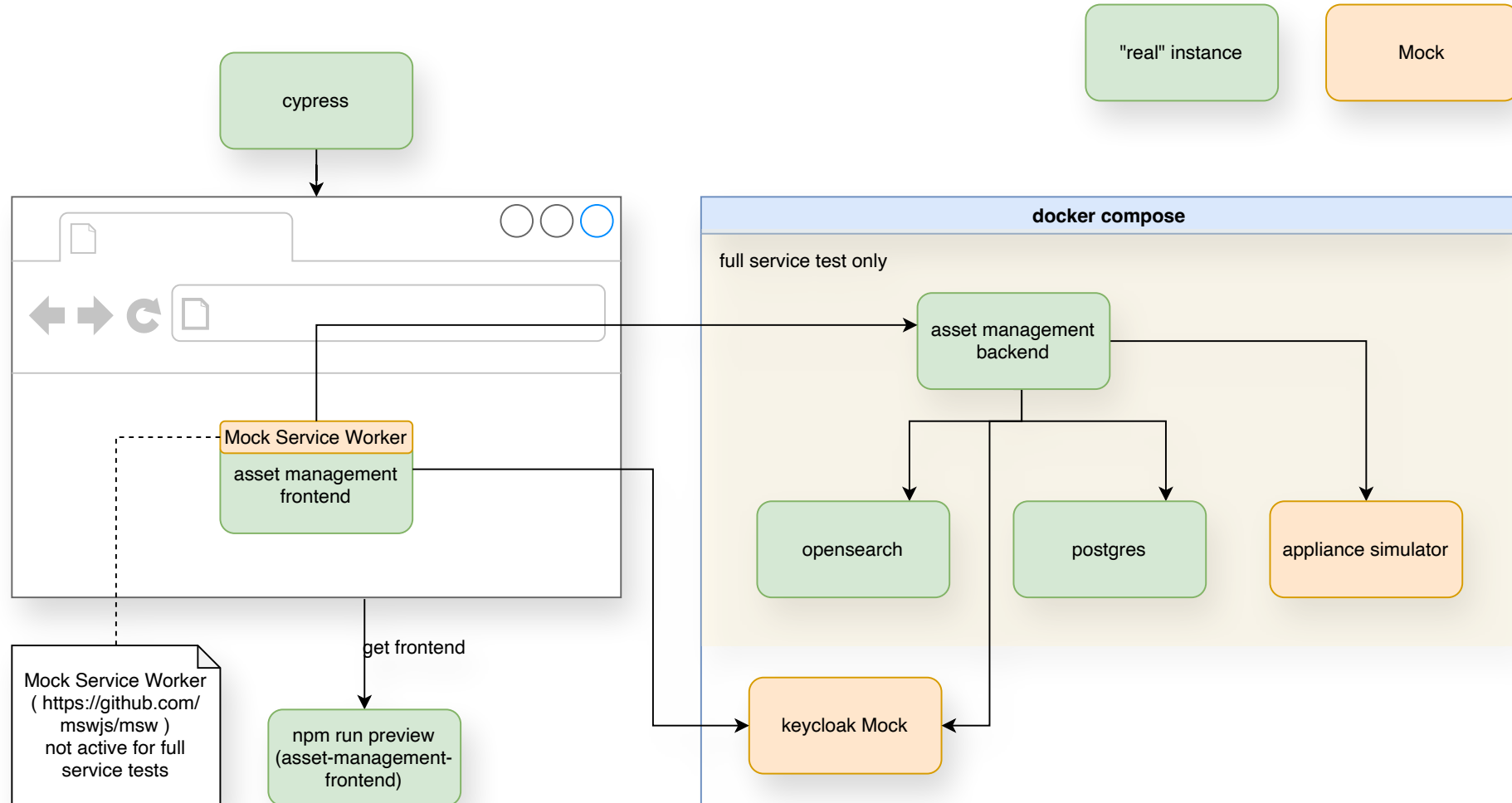
FRONTEND SERVICE TESTS



FRONTEND SERVICE TESTS



FRONTEND SERVICE TESTS



FRONTEND SERVICE TESTS



Page Object Pattern

```
class VulnerabilityDetailsPage {
  elements = {
    breadcrumbCurrentPage: (text: string) =>
      cy.contains("div", `Vulnerability details: ${text}`),
    breadcrumbPreviousPage: () =>
      cy
        .contains("div", "Vulnerability details: ")
        .parent()
        .siblings("a"),
    vulnerabilityName: (text: string) => cy.contains("div", text),
    divContainer: () => cy.get("div"),
    linkToAffectedAssets: () => cy.contains("a", "here"),
  }

  verifyThatBreadcrumbContainsText(text: string) {
    this.elements.breadcrumbCurrentPage(text).should("exist")
  }

  verifyThatVulnerabilityNameContainsText(text: string) {
    this.elements.vulnerabilityName(text).should("exist")
  }
}
```

FRONTEND SERVICE TESTS



Page Object Pattern

```
1  describe("Frontend service tests for checking the vulnerabilities details page", () => {
2      const vulnerabilityDetailsPage = require("../pages/vulnerabilityDetailsPage")
3      const vulnerabilityOverviewPage = require("../pages/vulnerabilityOverviewPage")
4      const pageLayout = require("../pages/pageLayout")
5      const vulnerabilityName = "The rexec service is running"
6
7      before(() => {
8          pageLayout.visitRootPage()
9      })
10
11     beforeEach(() => {
12         pageLayout.enterPage("/vulnerabilities")
13         vulnerabilityOverviewPage.cleanFilterContainer()
14         vulnerabilityOverviewPage.navigateToVulnDetailsByIndex(0)
15     })
16
17     it("vulnerability details page should show correct information", () => {
18         vulnerabilityDetailsPage.verifyThatBreadcrumbContainsText(
19             vulnerabilityName
20         )
21         pageLayout.verifyThatPageTitleExists("Vulnerability details")
22         vulnerabilityDetailsPage.verifyThatVulnerabilityNameContainsText(
23             vulnerabilityName
24         )
25     })
26 }
```

FRONTEND SERVICE TESTS



```
10 jobs:
11   run-service-test:
12     runs-on: "ubuntu-latest"
13     strategy:
14       matrix:
15         node-version:
16           - 18.x
17           - 20.x
```

...

```
36   - name: setup service test environment
37     run: make start-service-test-environment
38
39   - name: run service tests
40     run: |
41       npm run preview&
42       sleep 5s
43       ./node_modules/.bin/cypress run --reporter mochawesome --spec "service-tests/cypress/tests/service-tests/*.cts"
```

...

```
45   - name: Upload report folder
46     if: always()
47     uses: actions/upload-artifact@v4
48     with:
49       name: FrontendServiceTestReports_${{ matrix.node-version }}
50       path: service-tests/cypress/reports/
51       retention-days: 1
```

...

FE S TEST CHALLENGES

↻ Changing Data ↻ 

- **data** imported from **real systems changes** throughout time
 - ➔ **how many** and **which** entries will be there?

Greenbone

Analysis

Dashboard

Assets

Vulnerabilities

Import

MatchAll

Severity

↓

Name

The rexec service is running

10

PHP < 5.2.12 Multiple Vulnerabilities

10

Apache HTTP Server Multiple Security Vulnerabilities

10

Possible Backdoor: Ingreslock

10

Operating System (OS) End of Life (EOL) Detection

10

PHP End of Life (EOL) Detection - Linux

10

Distributed Ruby (dRuby/DRb) Multiple Remote Code Execution

10

Apache HTTP Server End of Life (EOL) Detection - Linux

10

PHP < 5.2.7 Multiple Vulnerabilities

10

phpMyAdmin End of Life (EOL) Detection - Linux

10

ISC BIND End of Life (EOL) Detection - Linux

10

OpenSSL End of Life (EOL) Detection - Linux

10

Mediawiki End of Life (EOL) Detection - Linux

10

Apache Log4j 2.0.x Multiple Vulnerabilities (Windows, Log4Shell)

10

Java RMI Server Insecure Default Configuration RCE Vulnerability

10

Nginx End of Life (EOL) Detection

10

TWiki XSS and Command Execution Vulnerabilities

10

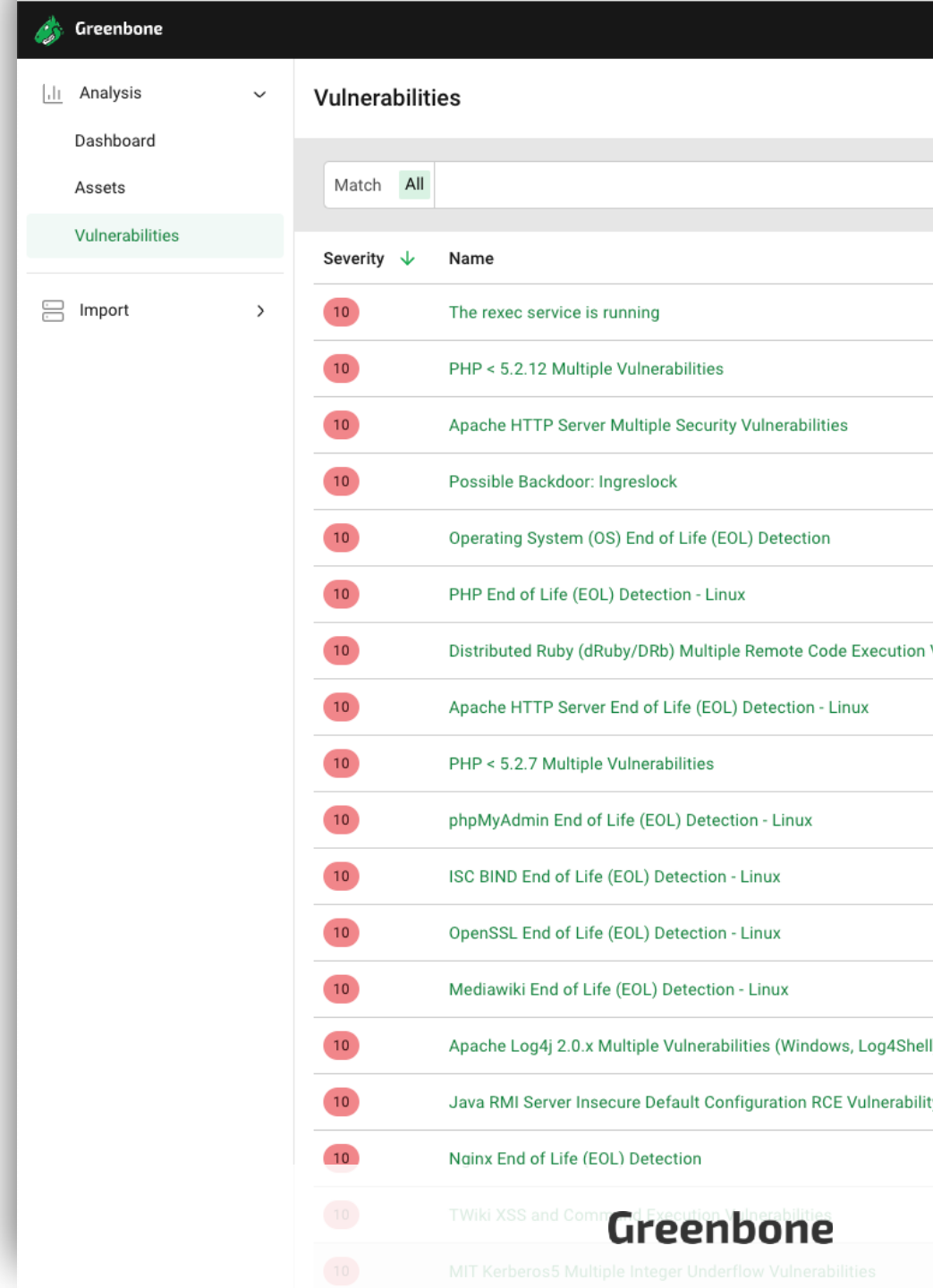
MIT Kerberos5 Multiple Integer Underflow Vulnerabilities

Greenbone

FE S TEST CHALLENGES

↯ Changing Data ↯

- data imported from **real systems** changes throughout time
 - ➡ how many and which entries will be there?
- our approach @ 
 - ➡ **mock** external systems
 - ➡ simplifies test setup
 - ➡ makes test shorter & more stable (reliable testdata!)



The screenshot shows the Greenbone web interface. On the left is a sidebar with navigation links: Analysis, Dashboard, Assets, Vulnerabilities (highlighted), and Import. The main content area is titled 'Vulnerabilities' and features a search bar with 'Match' and 'All' buttons. Below this is a table of vulnerabilities, each with a severity score of 10 (indicated by a red circle) and a descriptive name. The vulnerabilities listed include: 'The rexec service is running', 'PHP < 5.2.12 Multiple Vulnerabilities', 'Apache HTTP Server Multiple Security Vulnerabilities', 'Possible Backdoor: Ingreslock', 'Operating System (OS) End of Life (EOL) Detection', 'PHP End of Life (EOL) Detection - Linux', 'Distributed Ruby (dRuby/DRb) Multiple Remote Code Execution', 'Apache HTTP Server End of Life (EOL) Detection - Linux', 'PHP < 5.2.7 Multiple Vulnerabilities', 'phpMyAdmin End of Life (EOL) Detection - Linux', 'ISC BIND End of Life (EOL) Detection - Linux', 'OpenSSL End of Life (EOL) Detection - Linux', 'Mediawiki End of Life (EOL) Detection - Linux', 'Apache Log4j 2.0.x Multiple Vulnerabilities (Windows, Log4Shell)', 'Java RMI Server Insecure Default Configuration RCE Vulnerability', 'Nginx End of Life (EOL) Detection', 'TWiki XSS and Command Execution Vulnerabilities', and 'MIT Kerberos5 Multiple Integer Underflow Vulnerabilities'. The Greenbone logo is visible in the bottom right corner of the interface.

Severity	Name
10	The rexec service is running
10	PHP < 5.2.12 Multiple Vulnerabilities
10	Apache HTTP Server Multiple Security Vulnerabilities
10	Possible Backdoor: Ingreslock
10	Operating System (OS) End of Life (EOL) Detection
10	PHP End of Life (EOL) Detection - Linux
10	Distributed Ruby (dRuby/DRb) Multiple Remote Code Execution
10	Apache HTTP Server End of Life (EOL) Detection - Linux
10	PHP < 5.2.7 Multiple Vulnerabilities
10	phpMyAdmin End of Life (EOL) Detection - Linux
10	ISC BIND End of Life (EOL) Detection - Linux
10	OpenSSL End of Life (EOL) Detection - Linux
10	Mediawiki End of Life (EOL) Detection - Linux
10	Apache Log4j 2.0.x Multiple Vulnerabilities (Windows, Log4Shell)
10	Java RMI Server Insecure Default Configuration RCE Vulnerability
10	Nginx End of Life (EOL) Detection
10	TWiki XSS and Command Execution Vulnerabilities
10	MIT Kerberos5 Multiple Integer Underflow Vulnerabilities

FE S TEST CHALLENGES

↯ Code Maintainability Decreases ↯

- code may become **difficult to read**
 - ➡ often selectors on the DOM are **not self-explanatory**
- code changes may become **cumbersome**
 - ➡ often **selectors** are **repeated** throughout the code
 - ➡ can be a **lot of effort** to **change** all locations

```
28 it("sorting assets by appliance name should work correctly", () => {
29   cy.contains("h4", "Appliance name").click()
30   cy.waitForStableDOM()
31   cy.get("tbody>tr")
32     .eq(0)
33     .should("contain", Cypress.env("appliance_name"))
34   cy.contains("h4", "Appliance name").click()
35   cy.waitForStableDOM()
36   cy.get("tbody>tr")
37     .eq(0)
38     .should("contain", Cypress.env("appliance_name"))
39 })
40
41 it("sorting assets by last scan should work correctly", () => {
42   cy.contains("h4", "Last scan").click()
43   cy.waitForStableDOM()
44   cy.get("tbody>tr").eq(0).should("contain", "Feb. 07, 2023 08:42:02am")
45   cy.contains("h4", "Last scan").click()
46   cy.waitForStableDOM()
47   cy.get("tbody>tr").eq(0).should("contain", "Feb. 07, 2023 09:36:46am")
48 })
49
50 it("the pagination should work for different sizes and pages", () => {
51   cy.contains("span", "1-10 of 15 items").should("exist")
52   cy.get("tbody>tr").then((rows) => {
53     cy.expect(rows.length).to.equal(10)
54   })
55   cy.get("input[data-testid='paging-step-select'][value='10']").should(
56     "exist"
57   )
58
59   // navigate to page 2
60   cy.get("div[data-testid='pagination-container']").within(() => {
61     cy.contains("button", "2").click().waitForStableDOM()
62   })
63   cy.contains("span", "11-15 of 15 items").should("exist")
64   cy.get("tbody>tr").then((rows) => {
65     cy.expect(rows.length).to.equal(5)
66   })
67   cy.get("tbody>tr").eq(0).should("contain", "192.168.123.36")
68
69   // change paging size to 20
```

FE S TEST CHALLENGES

↯ Code Maintainability Decreases ↯

- code may become **difficult to read**
 - ➡ often selectors on the DOM are **not self-explanatory**
- code changes may become **cumbersome**
 - ➡ often **selectors** are **repeated** throughout the code
 - ➡ can be a **lot of effort** to **change** all locations
- our approach @ 
 - ➡ use **page object pattern**

```
28 it("sorting assets by appliance name should work correctly", () => {
29   cy.contains("h4", "Appliance name").click()
30   cy.waitForStableDOM()
31   cy.get("tbody>tr")
32     .eq(0)
33     .should("contain", Cypress.env("appliance_name"))
34   cy.contains("h4", "Appliance name").click()
35   cy.waitForStableDOM()
36   cy.get("tbody>tr")
37     .eq(0)
38     .should("contain", Cypress.env("appliance_name"))
39 })
40
41 it("sorting assets by last scan should work correctly", () => {
42   cy.contains("h4", "Last scan").click()
43   cy.waitForStableDOM()
44   cy.get("tbody>tr").eq(0).should("contain", "Feb. 07, 2023 08:42:02am")
45   cy.contains("h4", "Last scan").click()
46   cy.waitForStableDOM()
47   cy.get("tbody>tr").eq(0).should("contain", "Feb. 07, 2023 09:36:46am")
48 })
49
50 it("the pagination should work for different sizes and pages", () => {
51   cy.contains("span", "1-10 of 15 items").should("exist")
52   cy.get("tbody>tr").then((rows) => {
53     cy.expect(rows.length).to.equal(10)
54   })
55   cy.get("input[data-testid='paging-step-select'][value='10']").should(
56     "exist"
57   )
58
59   // navigate to page 2
60   cy.get("div[data-testid='pagination-container']").within(() => {
61     cy.contains("button", "2").click().waitForStableDOM()
62   })
63   cy.contains("span", "11-15 of 15 items").should("exist")
64   cy.get("tbody>tr").then((rows) => {
65     cy.expect(rows.length).to.equal(5)
66   })
67   cy.get("tbody>tr").eq(0).should("contain", "192.168.123.36")
68
69   // change paging size to 20
```

FE S TEST CHALLENGES

↩ Code Maintainability Decreases ↩

■ code

→ often

■ code

→ often

→ can

■ our a

→ use

```
38 ▶ it("sorting assets by last scan should work correctly", () => {
39     affectedAssetsPage.tableController().sortHeader("Last scan")
40     affectedAssetsPage.verifyTharFirstRowsContainsText(
41         "Feb 07, 2023, 08:42:02 AM"
42     )
43     affectedAssetsPage.tableController().sortHeader("Last scan")
44     affectedAssetsPage.verifyTharFirstRowsContainsText(
45         "Feb 07, 2023, 09:36:46 AM"
46     )
47 })
48
49 ▶ it("the pagination should work for different sizes and pages", () => {
50     affectedAssetsPage.paginationTextEquals("1-10 of 15 items")
51     affectedAssetsPage.tableController().verifyRowAmountIsEqualTo(10)
52     affectedAssetsPage.tableController().verifySelectedPageSize(10)
53
54     affectedAssetsPage.tableController().gotoPage(2)
55     affectedAssetsPage.paginationTextEquals("11-15 of 15 items")
56     affectedAssetsPage.tableController().verifyRowAmountIsEqualTo(5)
57     affectedAssetsPage.verifyTharFirstRowsContainsText("192.168.123.36")
58
59     affectedAssetsPage.tableController().setPageSize(20)
60     affectedAssetsPage.paginationTextEquals("1-15 of 15 items")
61     affectedAssetsPage.tableController().verifySelectedPageSize(20)
62 }
```

tory

e

```
28 it("sorting assets by appliance name should work correctly", () => {
29     cy.contains("h4", "Appliance name").click()
30     cy.waitForStableDOM()
31     cy.get("tbody>tr")
32         .eq(0)
33         .should("contain", Cypress.env("appliance_name"))
34     cy.contains("h4", "Appliance name").click()
35     cy.waitForStableDOM()
36     cy.get("tbody>tr")
37         .eq(0)
38         .should("contain", Cypress.env("appliance_name"))
39 })
40
41 it("sorting assets by last scan should work correctly", () => {
42     cy.contains("h4", "Last scan").click()
43     cy.waitForStableDOM()
44     cy.get("tbody>tr").eq(0).should("contain", "Feb. 07, 2023 08:42:02am")
45     cy.contains("h4", "Last scan").click()
46     cy.waitForStableDOM()
47     cy.get("tbody>tr").eq(0).should("contain", "Feb. 07, 2023 09:36:46am")
48 })
49
50 it("the pagination should work for different sizes and pages", () => {
51     cy.contains("span", "1-10 of 15 items").should("exist")
52     cy.get("tbody>tr").then((rows) => {
53         cy.expect(rows.length).to.equal(10)
54     })
55     cy.get("input[data-testid='paging-step-select'][value='10']").should(
56         "exist"
57     )
58
59     // navigate to page 2
60     cy.get("div[data-testid='pagination-container']").within(() => {
61         cy.contains("button", "2").click().waitForStableDOM()
62     })
63     cy.contains("span", "11-15 of 15 items").should("exist")
64     cy.get("tbody>tr").then((rows) => {
65         cy.expect(rows.length).to.equal(5)
66     })
67     cy.get("tbody>tr").eq(0).should("contain", "192.168.123.36")
68
69     // change paging size to 20
```

FE S TEST CHALLENGES

⚡ time and translation varies ⚡



- **i18n**: different time zones, date formats, translations
 - ➔ may **differ** between systems
 - ➔ **tests may fail** when checking for date, time or translations

Europe/Berlin 📍 | 🇬🇧 | ☀️ | John Doe 👤

+ Add appliance

Next import ⬆️⬆️	Action
Apr 25, 2024, 06:30:00 AM	⋮
Apr 25, 2024, 06:31:00 AM	⋮
Apr 25, 2024, 12:59:00 AM	⋮

< 1 >

Europe/Berlin 📍 | 🇩🇪 | ☀️ | John Doe 👤

+ Appliance hinzufügen

Nächster Import ⬆️⬆️	Aktion
26.04.2024, 06:30:00	⋮
25.04.2024, 18:31:00	⋮
26.04.2024, 00:59:00	⋮

Greenbone < 1 >

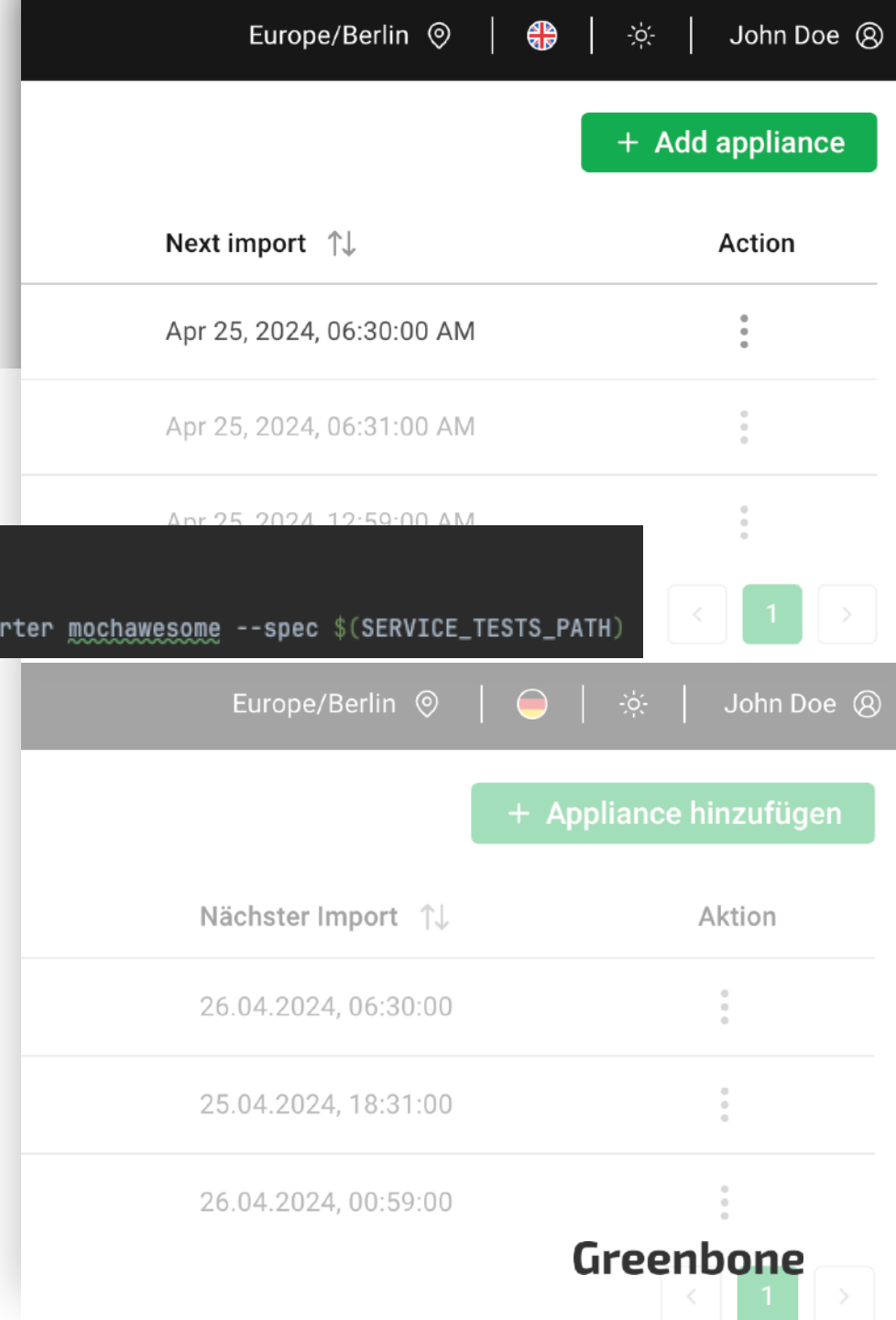
FE S TEST CHALLENGES

⚡ time and translation varies ⚡



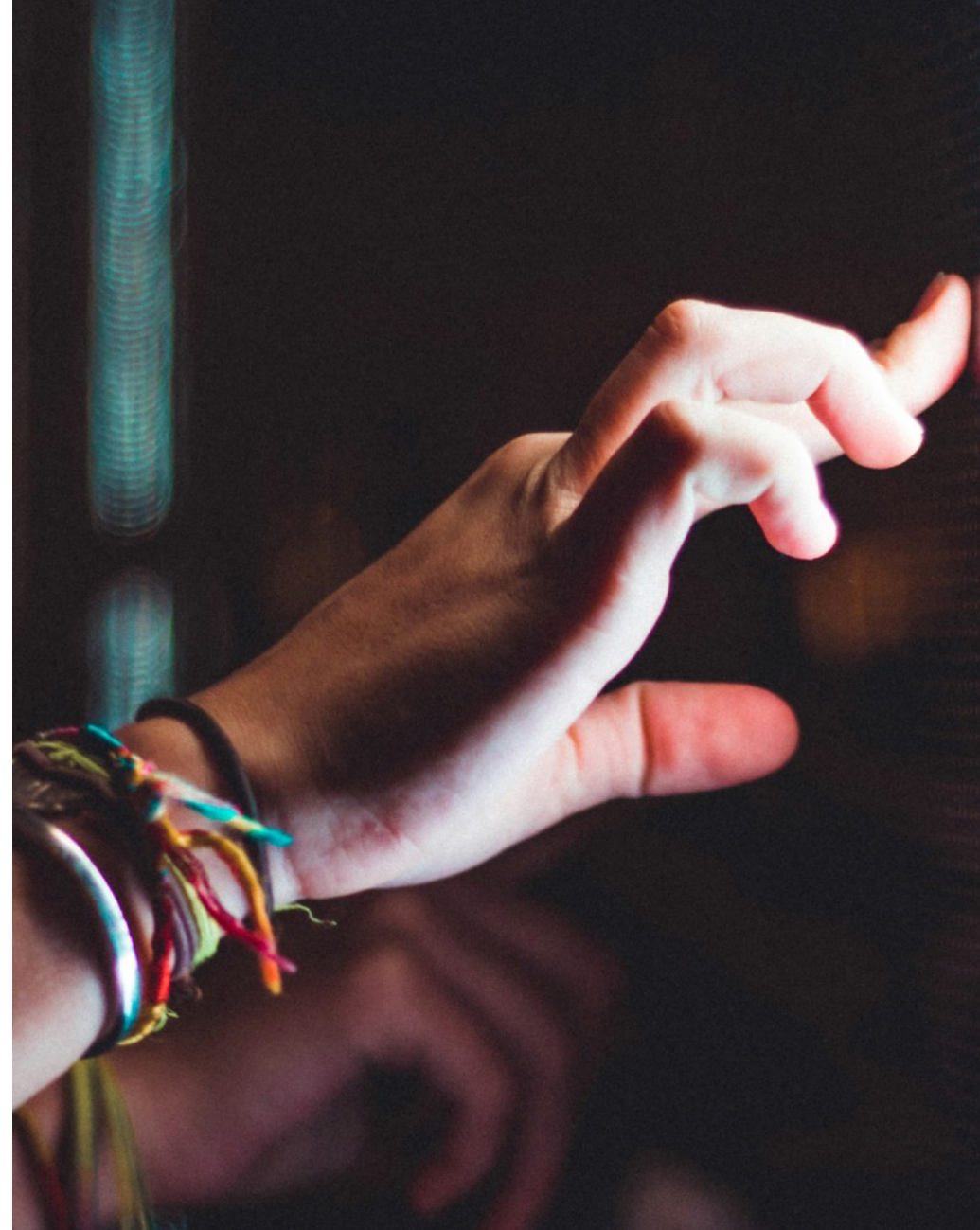
- i18n: different time zones, date formats, translations
 - ➔ may differ between
 - ➔ tests may fail when checking for date, time or translations
- our approach @ 🦊
 - ➔ **set time zone** explicitly
 - ➔ **select locale** explicitly (or **rely on default**)

```
56 .PHONY: start-service-tests
57 start-service-tests:
58     TZ=UTC ./node_modules/.bin/cypress run --reporter mochawesome --spec $(SERVICE_TESTS_PATH)
```



FE S TESTS: WRAP-UP

- consider **full service tests** (with backend)
- **mock** systems outside scope (except data stores)
- use **page object** pattern
- set **time zone / locale** explicitly
- like e2e
 - ➔ asynchronism: use **wait for** constructs
 - ➔ keep **test durations** low
 - ➔ **expect efforts** for maintenance



BACKEND SERVICE TESTS



BACKEND SERVICE TESTS



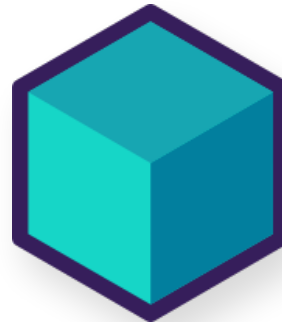
GitHub



python™

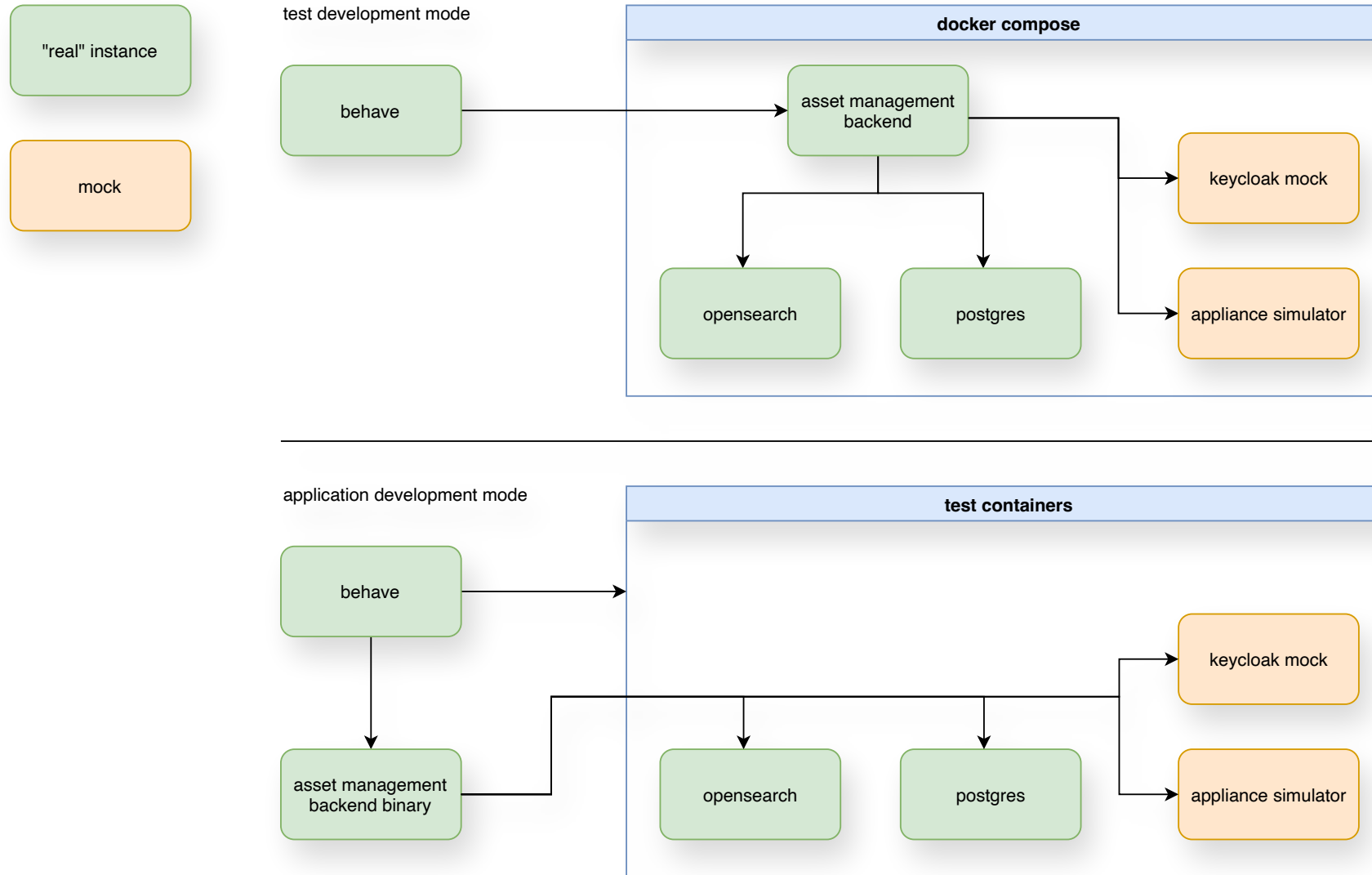


docker®



Testcontainers

BACKEND SERVICE TESTS



BACKEND SERVICE TESTS

```
98 @fixture.appliance.creation
99 @fixture.appliance.cleanup
100 >> Scenario: Updating an existing appliance with invalid ssh key fingerprint should deliver a 400 status and the expected error (via PUT /appliances/{applianceId})
101     Given the user is logged in
102     And the appliance 'Prepared Test Appliance' is available
103     When the existing appliance 'Prepared Test Appliance' is updated using invalid fingerprint
104     Then a 400 status is delivered
105     And the response has the text 'greenbone/asset/host-key-invalid' for the field 'type'
106     And the response has the text 'SHA256:yyq1KasQ622bc5Ez5nH/fh1Mcse2ZFhewcG4VH3Mir4' for the custom field 'retrievedSshFingerprint'
```

```
36 @step(u'a (?P<status_code>\\d+) status is delivered')
37 def step_check_error_code(context, status_code):
38     wait.wait_for_context_response(context, condition=lambda r: assert_that(r.status_code)
39     .is_equal_to(int(status_code)))
```

BE S TESTS: WRAP-UP

- like frontend service tests:
 - ➔ **mock systems** outside scope (except data stores)
 - ➔ asynchronism: use **wait for** constructs
 - ➔ keep **test durations** low
 - ➔ **expect efforts** for maintenance



OUTLOOK

- Sometimes we get "We detected that the **Electron** Renderer process just **crashed**." in the CI 
 - ➔ we need to investigate
- run e2e test after each deployment
- synchronize QA approach in whole company
- automated cross browser test
- contract driven testing (e.g. pact)
- metrics for quality (e.g. escaped bugs)



EXPLORATORY TESTING

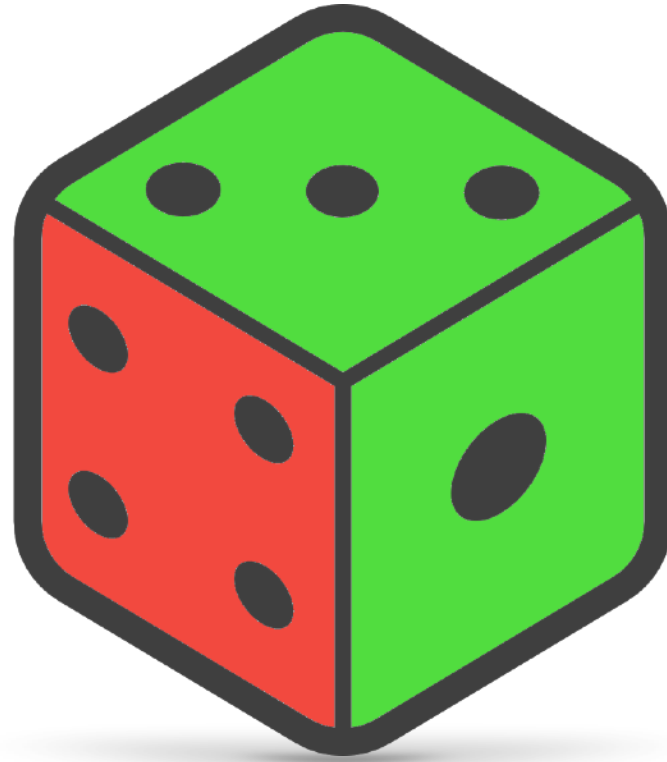
- **Scripted** (espec. automated) **testing is important**
e.g. for regression and possibly short feedback cycles
- But think of your scripted tests as **threads of a net**
- They catch whatever you thought of, but there are always **things** that are **not caught**
- Given the **unlimited space of possibilities** it is impossible to weave the net fine enough to catch every possibility
- This is where **exploratory testing** comes in handy!

<https://www.goodreads.com/book/show/15980494-explore-it>



BEWARE OF THE DICE

don't ignore flaky tests!



Thank You!

Questions?

LET'S CREATE A FUTURE TOGETHER

We are more than just a business: We are a team of creative thinkers, innovative problem solvers, dedicated professionals – and passionate humans.

Be part of the Solution: find our job postings here



Our key benefits:

Remote Work options: 100% remote work possible

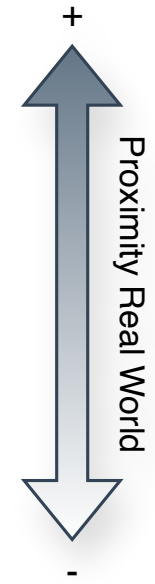
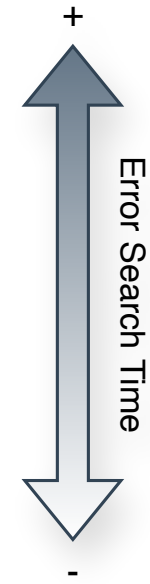
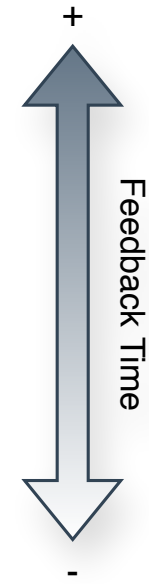
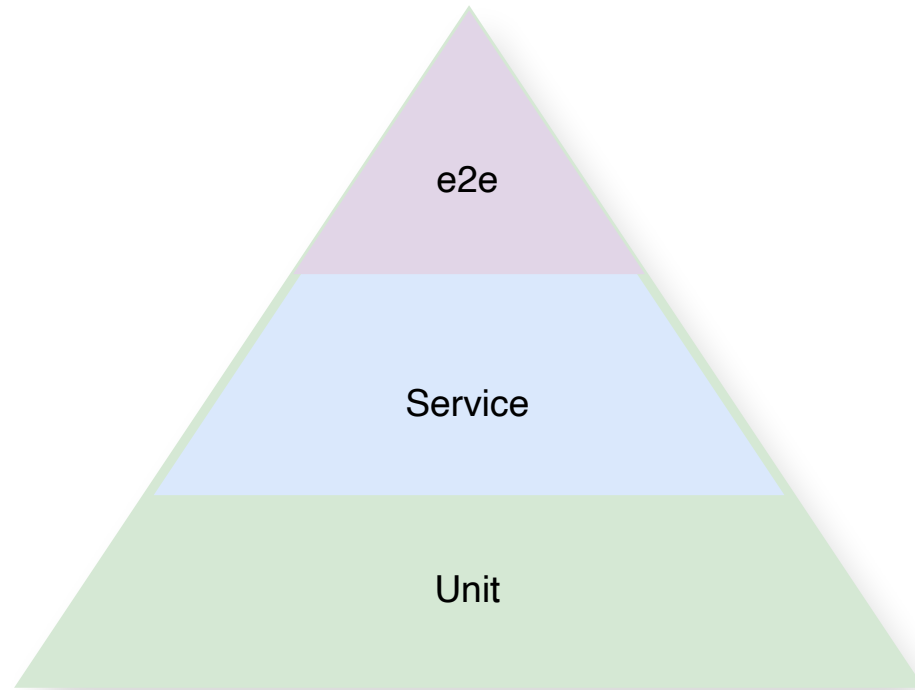
Flexible working hours

Opportunities for participation in company decisions

Backup

Greenbone

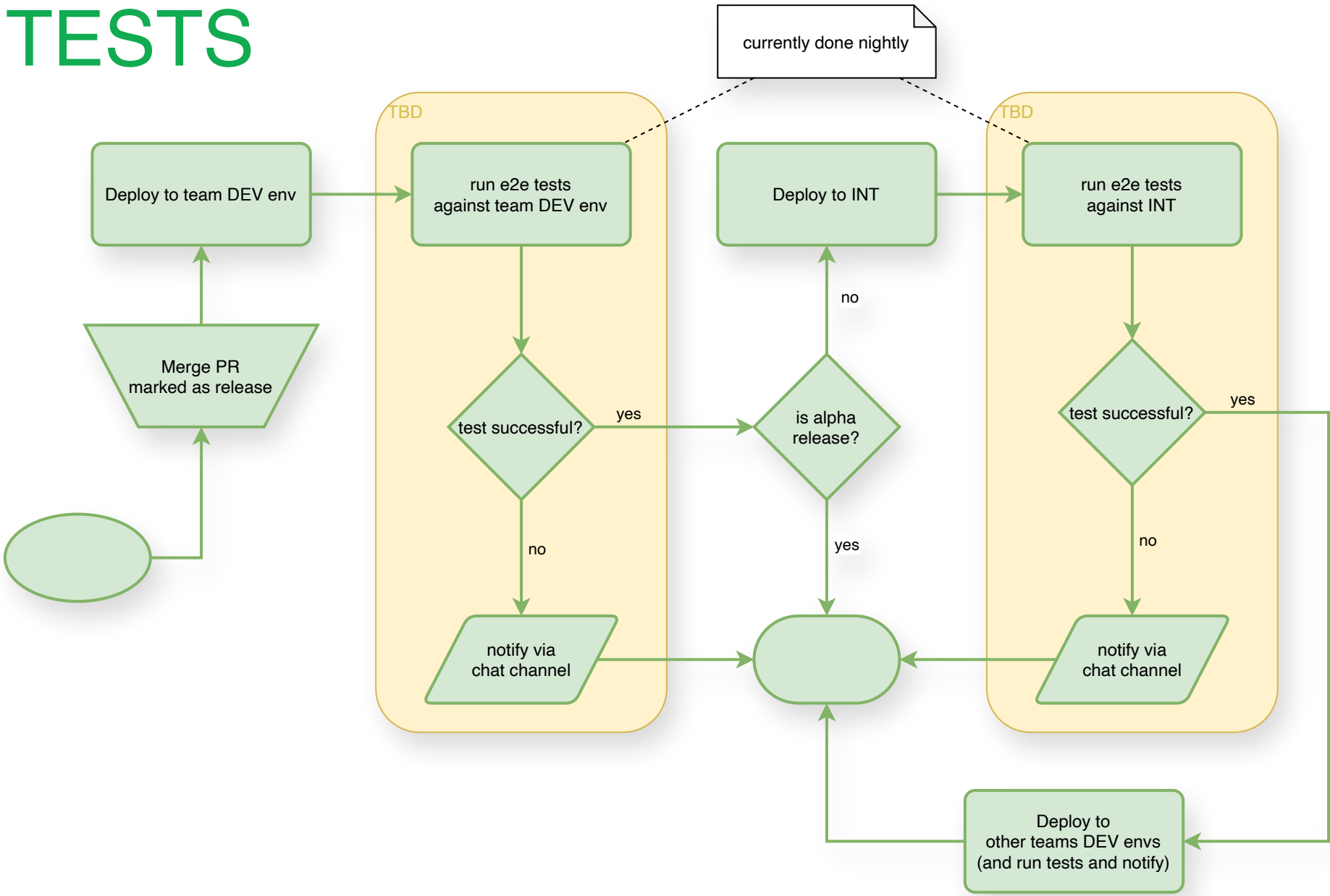
TEST PYRAMID



E2E TESTS



E2E TESTS



E2E TEST CHALLENGES

↯ Active Users on Test System ↯ 

- **test systems** are also used for **manual testing** and presentations
- ➡ test outcomes can be **influenced** by **user actions**

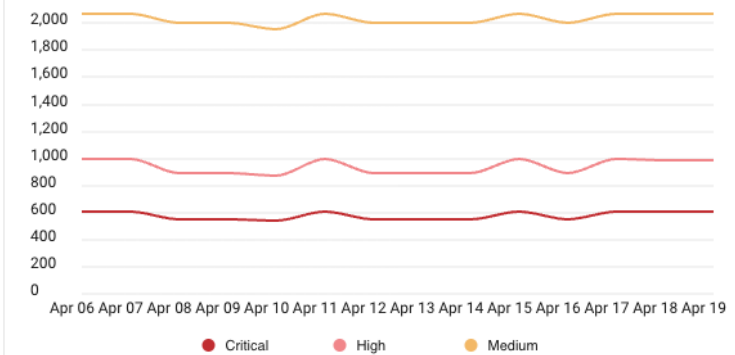
Dashboard

Match All

Number of assets

2,286

Vulnerability trends



Top 10 most vulnerable a

target-gos-qa-debia...
gos-qa-metasploitab...
target-macos.qm-mai...
WIN10CIS1809B.gb1.I...
target-gos-qa-debia...
vhosts-target-3.sup...
vhosts-target-1a.su...
tls13-qm.devel.gree...
ZANXC6FA1Y7 (2691c...
VF95ES4C0UQX (72f61...

Critical Top 10 vulnerabilities

Name	Assets
mysql / mariadb default credentials (mysql...	68
distcc rce vulnerability (cve-2004-2687)	66
postgresql default credentials (postgresql ...	59
vnc brute force login	52
twiki xss and command execution vulnera...	44
operating system (os) end of life (eol) dete...	11

High Top 10 vulnerabilities

Name	Assets
vsftpd compromised source packages bac...	115
distcc detection	60
the rlogin service is running	59
phpinfo() output reporting (http)	57
ssh brute force logins with default credenti...	52
php-cgi-based setups vulnerability when p...	50

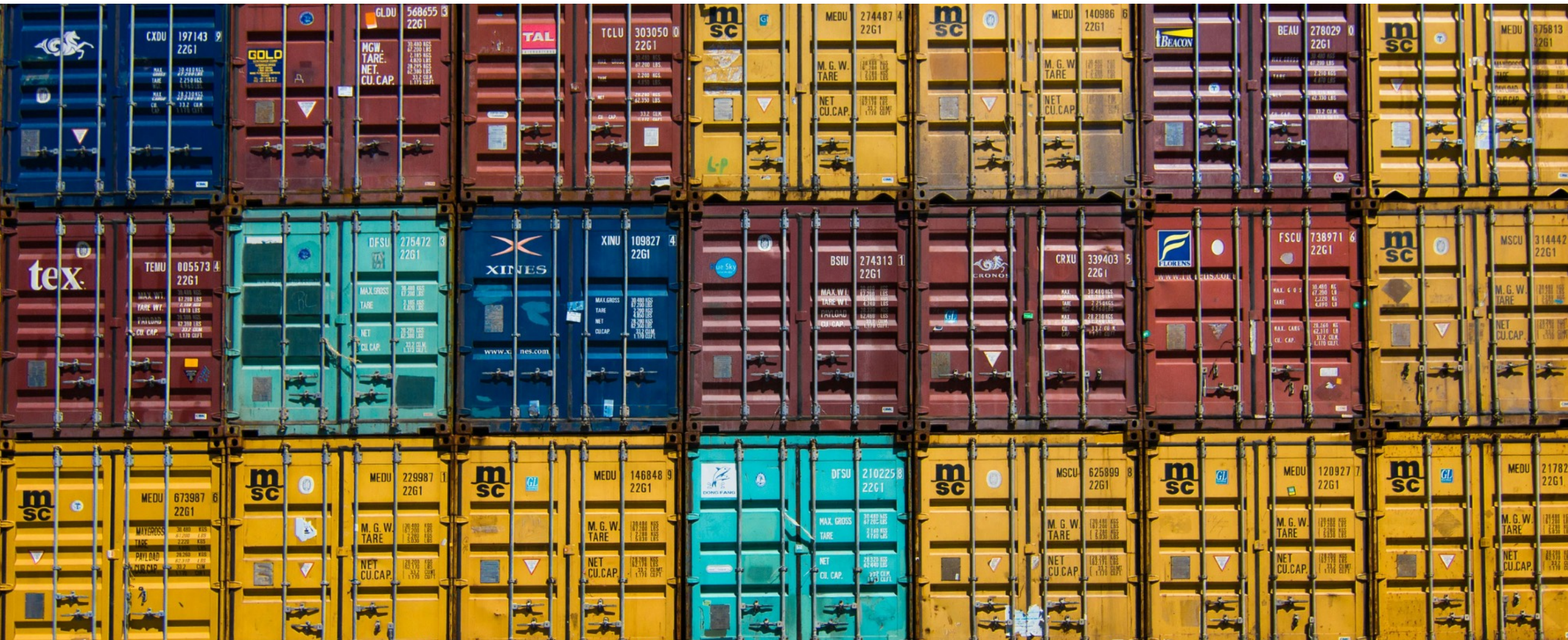
E2E TEST CHALLENGES

↯ Active Users on Test System ↯ 

- **test systems** are also used for **manual testing** and presentations
 - ➔ test outcomes can be **influenced** by **user actions**
- our approach @ 
 - ➔ **label** test data and **filter** for it
 - ➔ conventions on **what** (not) **to change**
 - ➔ alternative: separate envs (\$\$\$), create separate accounts



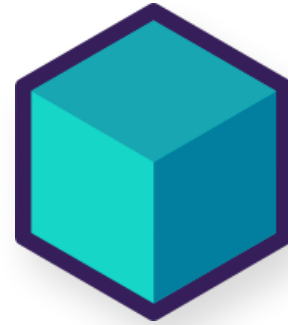
BACKEND UNIT TESTS



BACKEND UNIT TESTS



GitHub



Testcontainers



docker®

BACKEND UNIT TESTS

```
327 func TestVulnerabilityExport(t *testing.T) {
328     testCases := map[string]testCase{
329         "happy path": {
330             acceptHeader:      "text/csv",
331             requestBody:          vulnerabilityRequest(),
332             expectedStatusCode:    http.StatusOK,
333             mockedVulnerabilitySearch: getMockedVulnerabilitySearch(t),
334             bodyValidation: func(mvc rest_test.MockMvc) {
335                 mvc.ExpectBody( body: "Severity,Name,QoD,Total Assets,Solution type\n" +
336                     "7.5,FTP Brute Force Logins Reporting,95,1,Mitigation\n")
337             },
338         },
339         "invalid media type": {
340             acceptHeader:      "application/json",
341             expectedStatusCode: http.StatusNotAcceptable,
342             bodyValidation: func(mvc rest_test.MockMvc) {
343                 mvc.ExpectJson( json: "{\"title\":\"Invalid media type was specified in the Accept header (should be text/csv)\", \"type\":\"greenbone/generic-error\"}")
344             },
345         },
346         "invalid body": {
```

BACKEND UNIT TESTS

```
for name := range testCases {  
    t.Run(name, func(t *testing.T) {  
        mockMvc := setupExport(testCases[name])  
        mockMvc = mockMvc.  
            Perform(t, rest_test.Post( url: "/api/asset-management/export/vulnerabilities").  
                Header( key: "Accept", []string{  
                    testCases[name].acceptHeader,  
                }).  
                Content(testCases[name].requestBody)).  
                ExpectStatusCode(testCases[name].expectedStatusCode)  
        testCases[name].bodyValidation(mockMvc)  
    })  
}
```

BACKEND UNIT TESTS - OPENSEARCH

```
56 ▶ "paging and sorting": {
57     oid: "ba312986-9667-413a-95e8-030dcb039b37",
58     resultSelector: query.ResultSelector{
59         Filter: &filter.Request{},
60         Sorting: &sorting.Request{
61             SortColumn: "name",
62             SortDirection: "asc",
63         },
64         Paging: &paging.Request{
65             PageIndex: 0,
66             PageSize: 2,
67         },
68     },
69     expectedRequest: string(readFile(t, name: "testJsons/vulnerabilityByAssetRequestWithPagingAndSorting.json")),
70     expectedResponse: string(readFile(t, name: "testJsons/vulnerabilityByAssetResponse.json")),
71     expectedResults: vulnerabilitiesByAssets(),
72     expectedTotalRowCount: 286,
73     expectedError: nil,
74 }
```

BACKEND UNIT TESTS - OPENSEARCH

```
56 ▶ "paging and sorting": {
57     oid: "ba312986-9667-413a-95e8-030dcb039b37",
58     resultSelector: query.ResultSelector{
59         Filter: &filter.Request{},
60         Sorting: &sorting.Request{
61             SortColumn: "name",
62             SortDirection: "asc",
63         },
64         Paging: &paging.Request{
65             PageIndex: 0,
66             PageSize: 2,
67         },
68     },
69     expectedRequest: string(readFile(t, name: "testJsons/vulnerability")),
70     expectedResponse: string(readFile(t, name: "testJsons/vulnerability")),
71     expectedResults: vulnerabilitiesByAssets(),
72     expectedTotalRowCount: 286,
73     expectedError: nil,
74 },
```

```
1  {
2      "from": 0,
3      "query": {
4          "bool": {
5              "filter": [
6                  {
7                      "term": {
8                          "asset.id": {
9                              "value": "ba312986-9667-413a-95e8-030dcb039b37"
10                          }
11                      }
12                  }
13              ]
14          }
15      },
16      "size": 2,
17      "sort": [
18          {
19              "vulnerabilityTest.name.keyword": {
20                  "order": "asc"
21              }
22          }
23      ]
24  }
```

BE UNIT TESTS: WRAP-UP

- Make use of **data tables** / data driven testing
- make it possible to **verify test expectations**
- use **testcontainers** only **where unavoidable**
- like backend service tests:
 - ➔ **mock functions** outside scope (except data stores)
 - ➔ asynchronism: use **wait for** constructs
 - ➔ keep **test durations** low

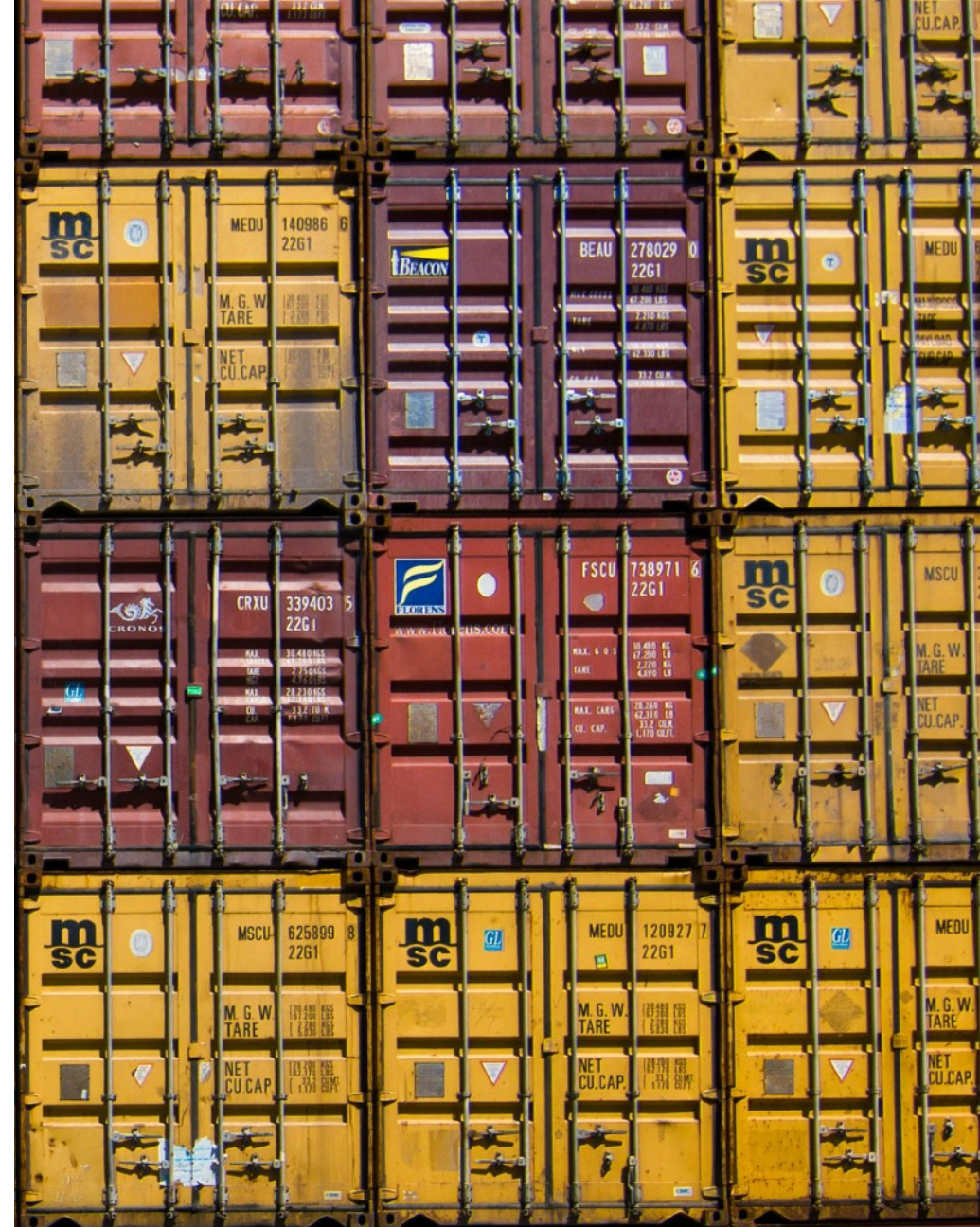


IMAGE CREDITS



Foto von Nicholas Green auf Unsplash



Foto von Jonathan Sifa auf Unsplash



Foto von Denys Nevozhai auf Unsplash



Foto von Nicolas Hoizey auf Unsplash



Foto von Jukan Tateisi auf Unsplash



Foto von Aideal Hwa auf Unsplash



Foto von Dipesh Shrestha auf Unsplash



Foto von NASA auf Unsplash



Foto von Killian Cartignies auf Unsplash

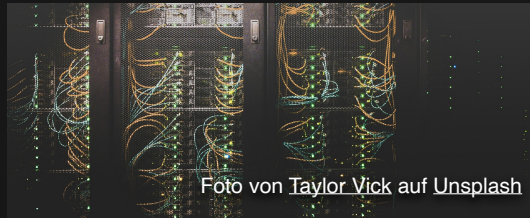


Foto von Taylor Vick auf Unsplash



Foto von Guillaume Bolduc auf Unsplash



Foto von Jonas Jacobsson auf Unsplash