10TH ZABBIX SUMMIT 2022

BUG OFF: HUNTING FOR ZABBIX VULNERABILITIES

Alexey Mitrofanov
Cybersecurity Lead

Agenda

1. What are security vulnerabilities?
2. What are main drivers to find vulnerabilities?
3. How have we been seeking solutions?
4. What challenges have we faced?
5. Outcomes.

1.

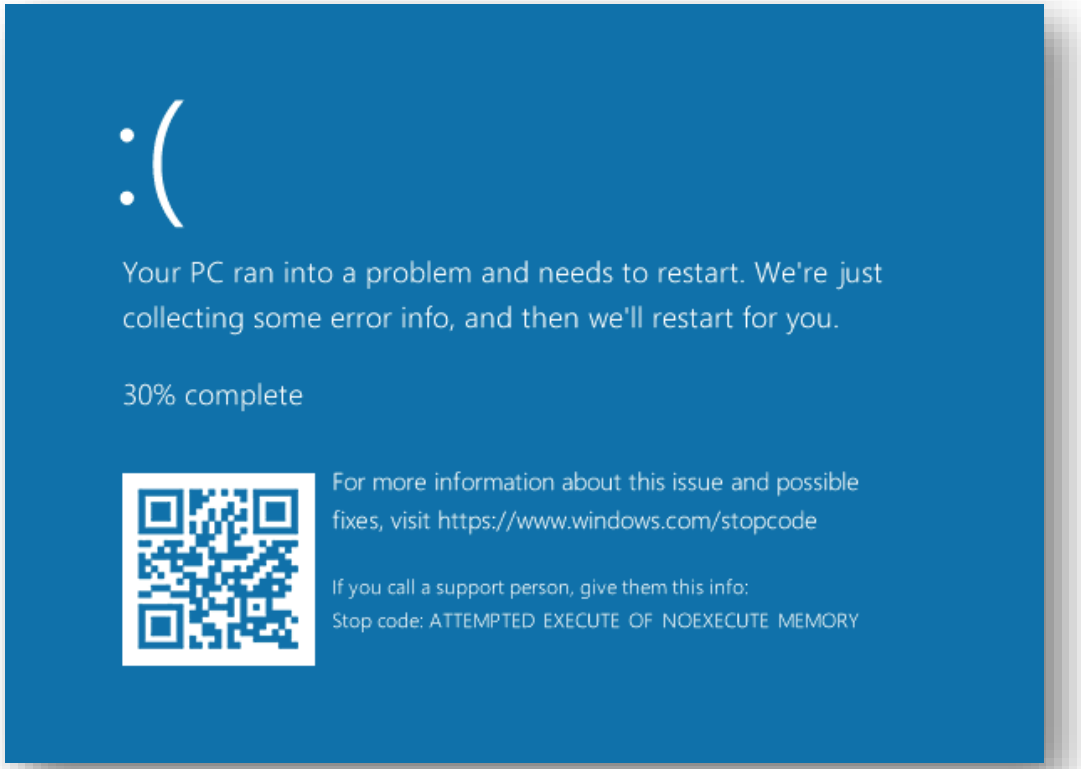
WHAT ARE SECURITY VULNERABILITIES?

"A security vulnerability is an error or flaw within an IT resource that could be exploited by attackers"

https://jfrog.com/

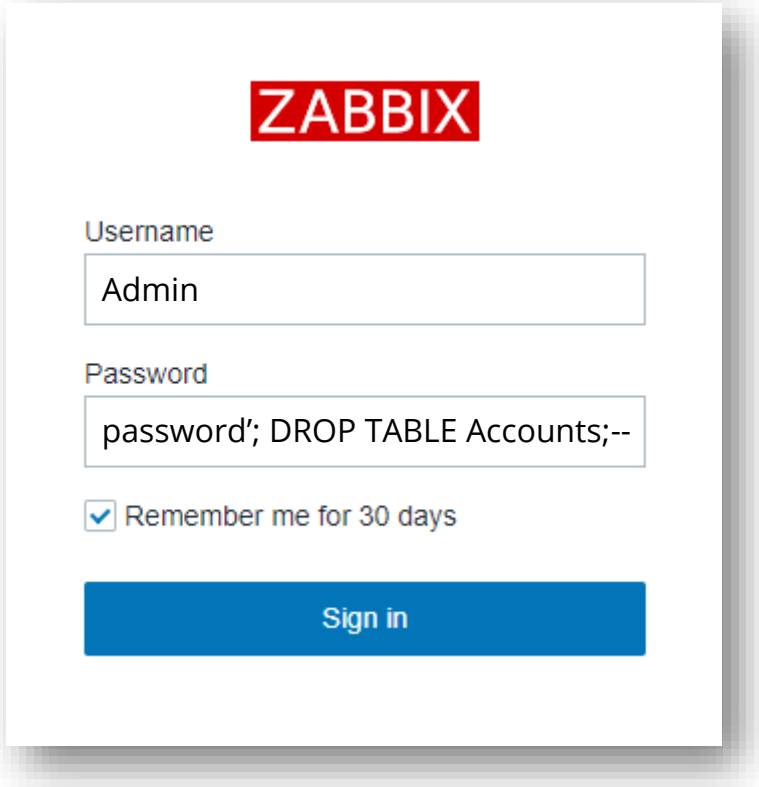
WHAT ARE SECURITY VULNERABILITIES?
Their forms

- Coding mistake



WHAT ARE SECURITY VULNERABILITIES?
Their forms

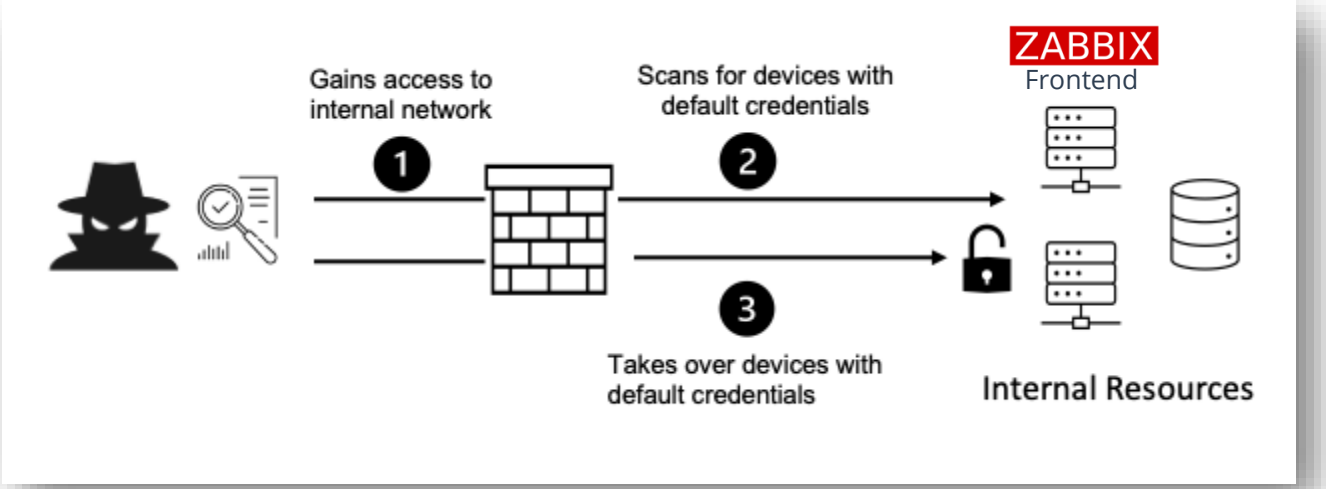
- Coding mistake
- Developer's mistake



https://www.codecademy.com/learn/

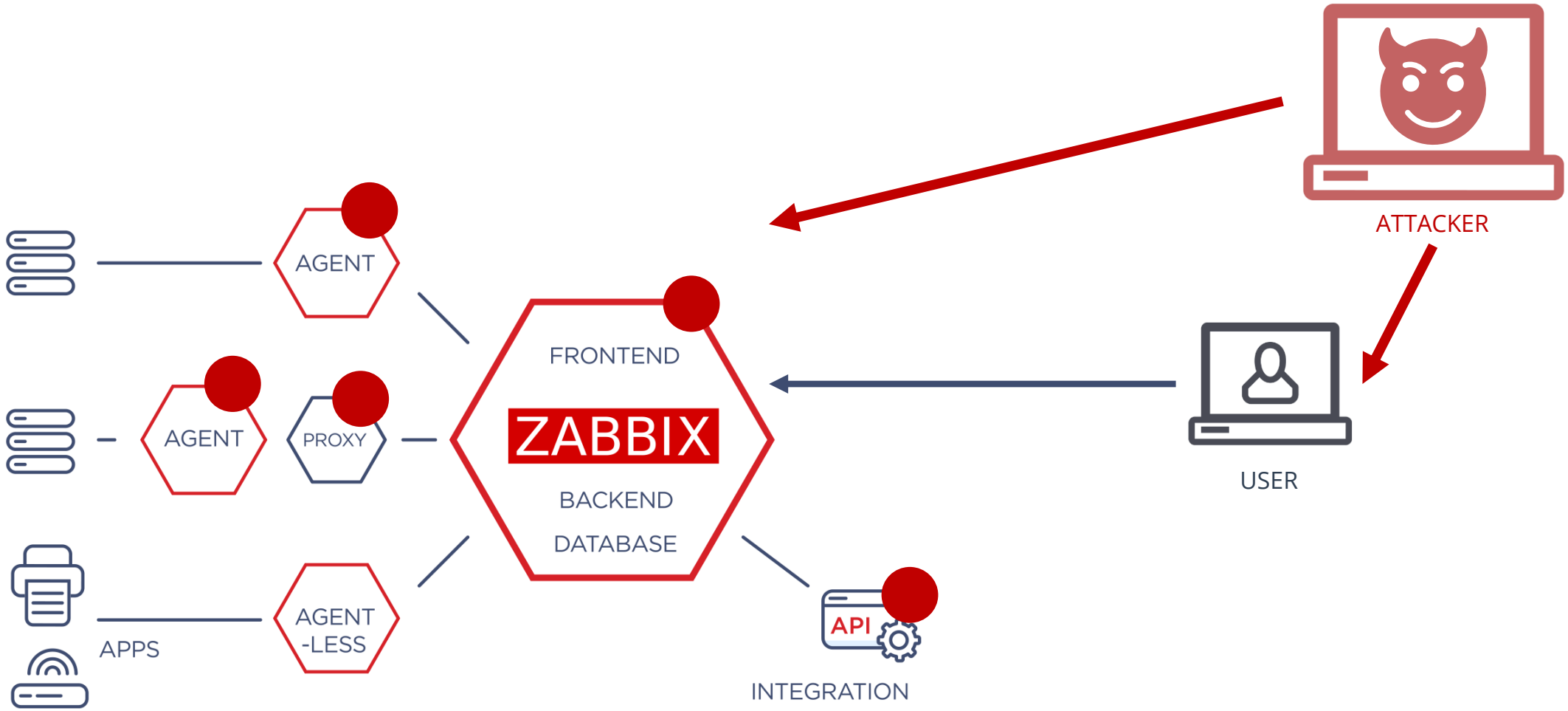
WHAT ARE SECURITY VULNERABILITIES?
Their forms

- Coding mistake
- Developer's mistake
- IT asset misconfiguration
- Many other...



https://support.f5.com/

WHAT ARE SECURITY VULNERABILITIES?
Scope



WHAT ARE SECURITY VULNERABILITIES?
Examples

- The list of publicly disclosed cybersecurity vulnerabilities in MITRE CVE*

  (https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=zabbix)

- Zabbix Ticket management system

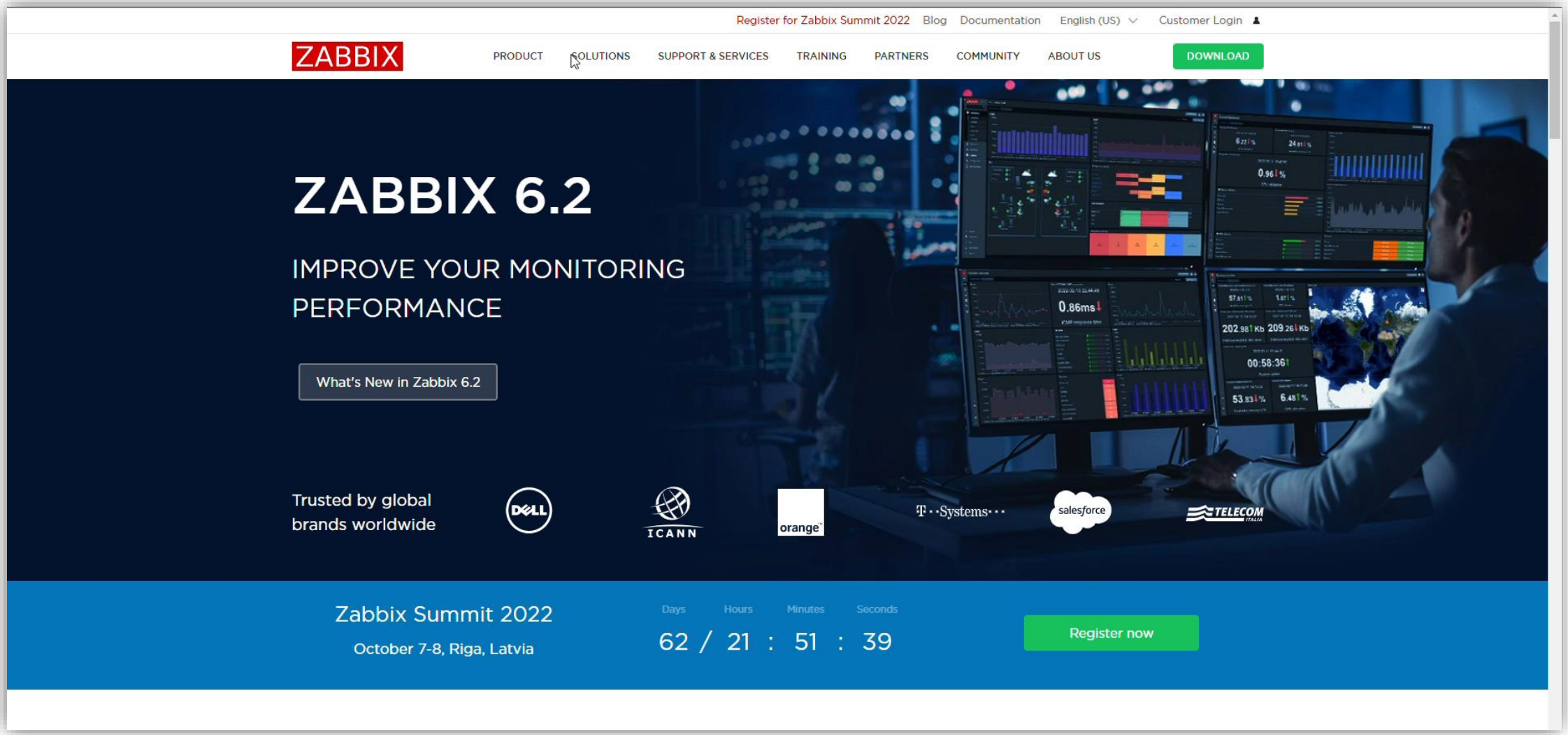
  (https://support.zabbix.com/)

- Zabbix Security Advisories and CVE database

  (https://www.zabbix.com/security_advisories)

***MITRE CVE** - Massachusetts Institute of Technology Research and Engineering Common Vulnerabilities and Exposures

WHAT ARE SECURITY VULNERABILITIES?
Examples

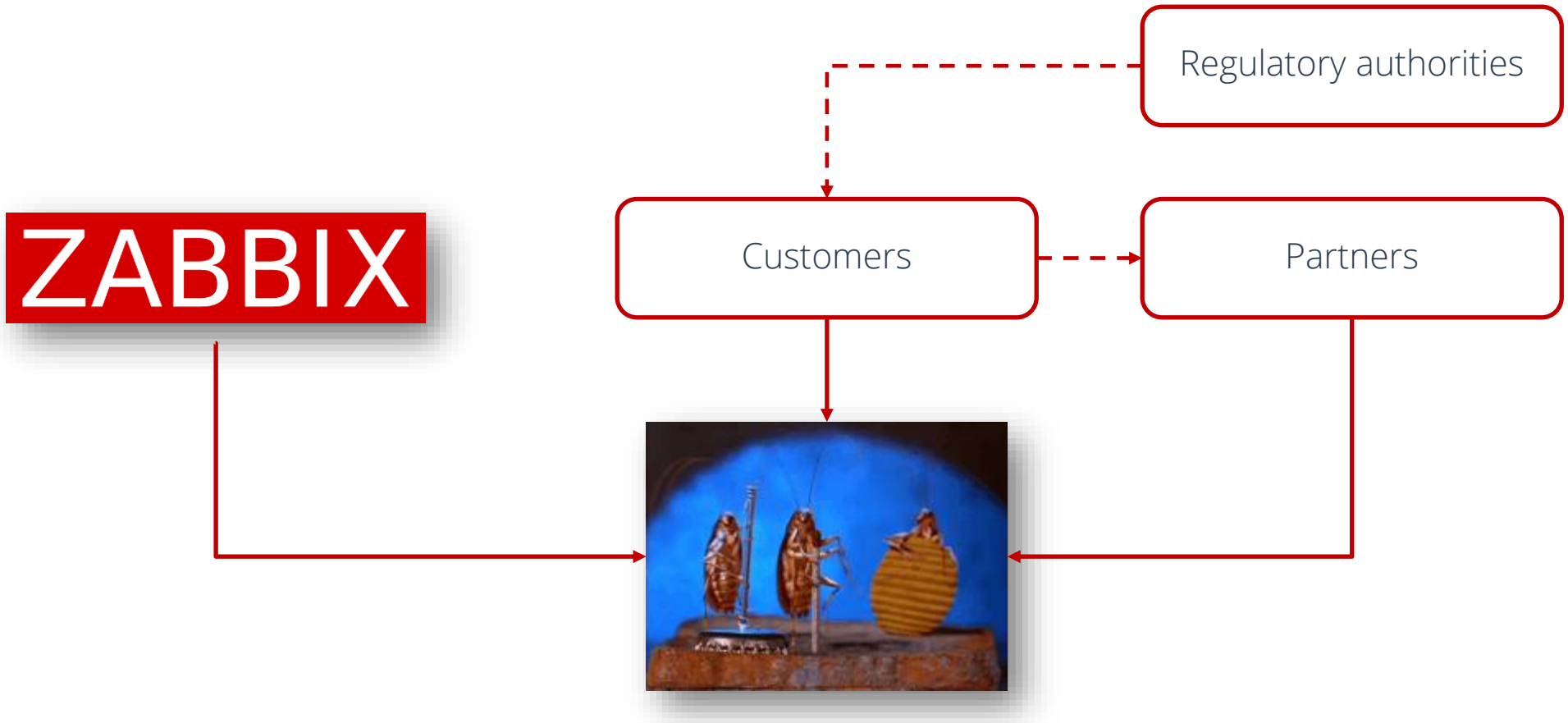


2.

WHAT ARE MAIN DRIVERS TO FIND VULNERABILITIES?

Regulatory authorities? Customers? Someone else?

WHAT ARE MAIN DRIVERS TO FIND VULNERABILITIES?

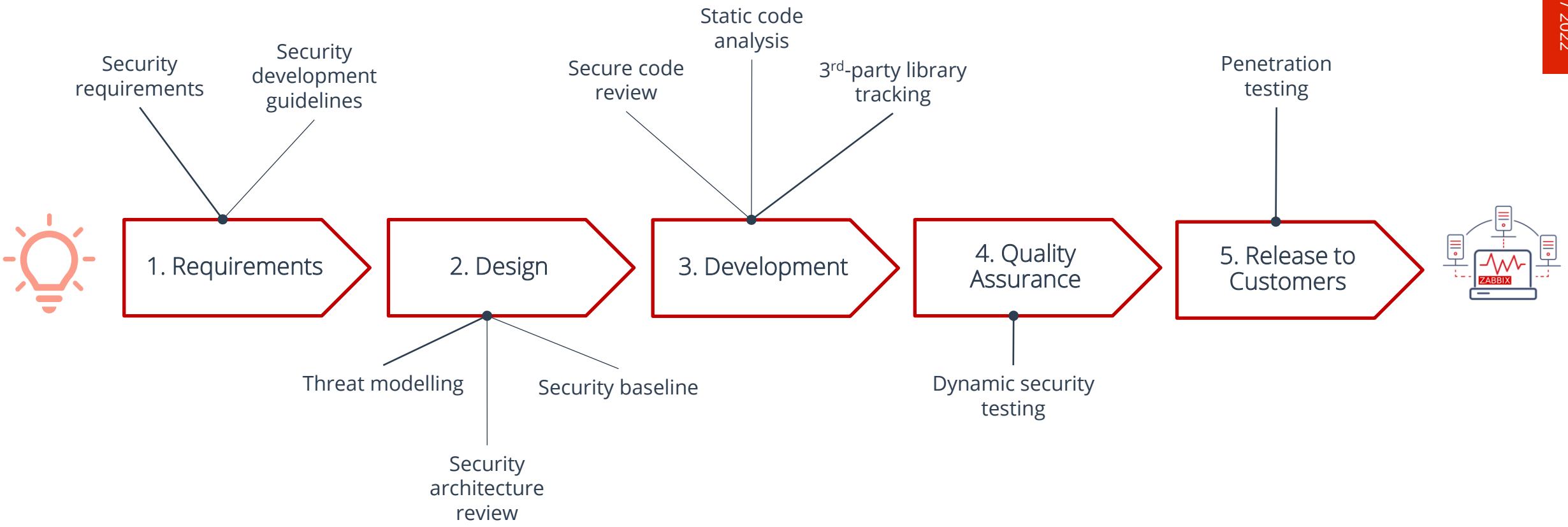


Who wants to see the bugs fixed?

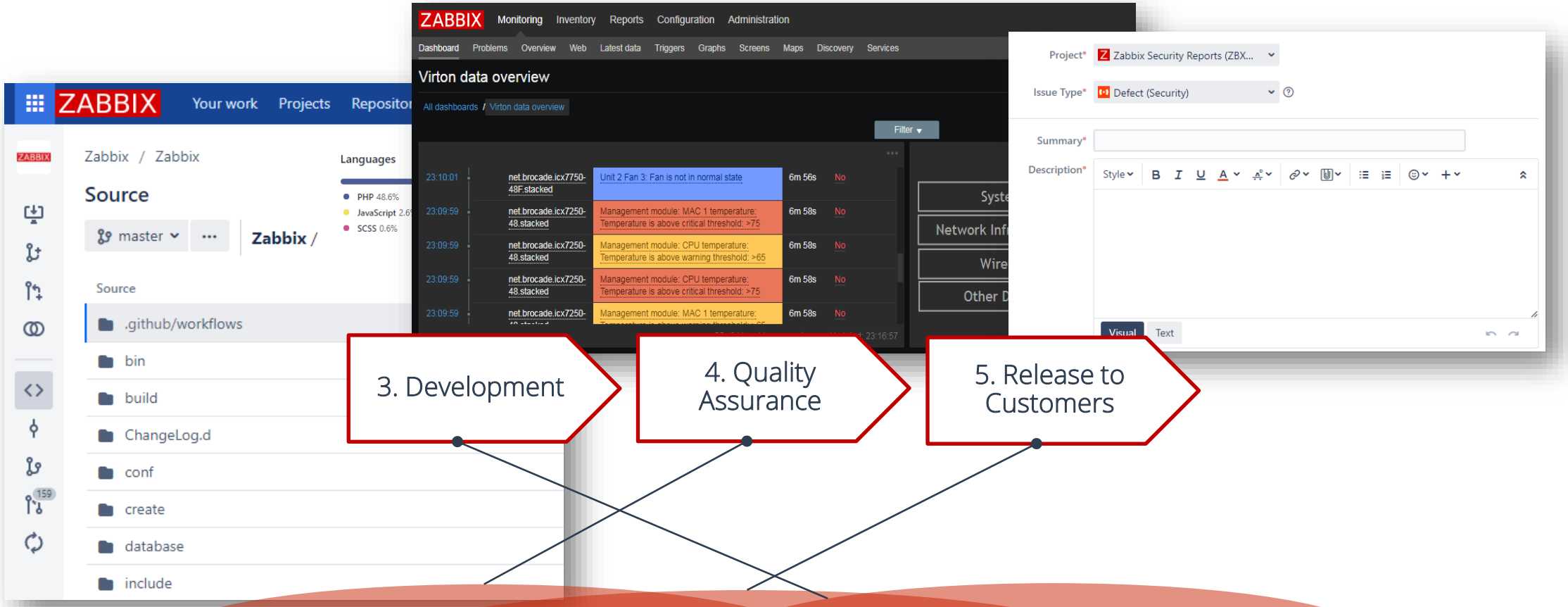
3.

HOW HAVE WE BEEN SEEKING SOLUTIONS?

HOW HAVE WE BEEN SEEKING SOLUTIONS?
Software Development Lifecycle (SDLC)

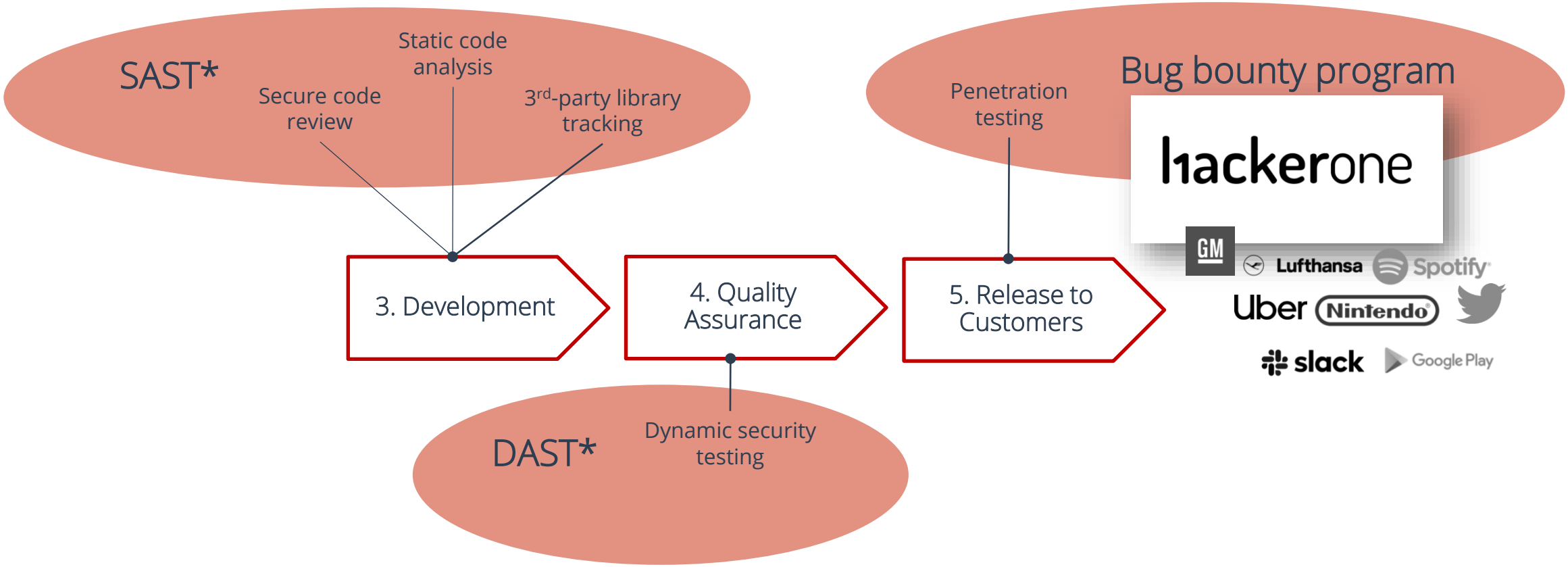


HOW HAVE WE BEEN SEEKING SOLUTIONS?
Zabbix cybersecurity program hotspots



HOW HAVE WE BEEN SEEKING SOLUTIONS?

Significant improvements this year



*SAST - Static Application Security Testing Tool
*DAST - Dynamic Application Security Testing Tool

HOW HAVE WE BEEN SEEKING SOLUTIONS?
Significant improvements this year

hackerone

ZABBIX

Zabbix

Zabbix is a universal all-in-one solution that offers real-time monitoring of your entire infrastructure and is completely open-source.

http://zabbix.com · @zabbix

Bug Bounty Program
Launched on Sep 2022

Managed by HackerOne
Includes retesting

Bookmark Subscribe

Rewards

Low	Medium	High	Critical
https://www.zabbix.com/download_sources			
$150	$500	$1,000	$2,000

Rewards

Our rewards are based on severity per CVSS (the Common Vulnerability Scoring Standard). Please note these are general guidelines, and reward decisions are up to the discretion of Zabbix.

Last updated on September 5, 2022. View changes

4.

WHAT CHALLENGES HAVE WE FACED?

WHAT CHALLENGES HAVE WE FACED?
Choosing the right tool

Static Application Security Testing vs **Dynamic Application Security Testing**

Developer view

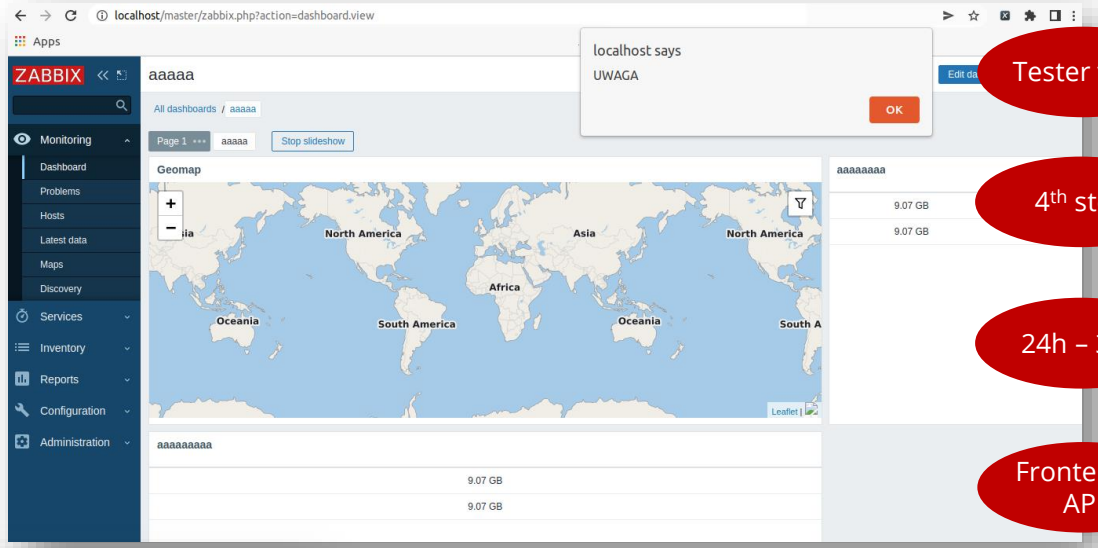
3rd step

10 min – 1h

All code

```
271
        page->alloc = INIT_PERF_REST_SIZE;
        page->data = (char *)zbx_malloc(NULL, page->alloc);
        page->url = zbx_strdup(NULL, url);
        zbx_rtrim(page->url, "/");
276     url_sz = strlen(page->url);
277
        if (url_sz < ZBX_CONST_STRLEN("/sdk") || 0 != strcmp(&page->url[url_sz - ZBX_CONST_STRLEN("/sdk")], "/sdk"))
        {
            *error = zbx_strdup(*error, "cannot initialize rest service url");
            goto out;
282     }
283
        memcpy(&page->url[url_sz - ZBX_CONST_STRLEN("api")], "api", ZBX_CONST_STRLEN("api"));
        *headers = curl_slist_append(*headers, ZBX_XML_HEADER1);
        *headers = curl_slist_append(*headers, ZBX_XML_HEADER2);

        if (CURLE_OK != (err = curl_easy_setopt(*easyhandle, opt = CURLOPT_HTTPHEADER, *headers)) ||
289             CURLE_OK != (err = curl_easy_setopt(*easyhandle, opt = CURLOPT_COOKIEFILE, "")) ||
290             CURLE_OK != (err = curl_easy_setopt(*easyhandle, opt = CURLOPT_FOLLOWLOCATION, 1L)) ||
            CURLE_OK != (err = curl_easy_setopt(*easyhandle, opt = CURLOPT_WRITEFUNCTION, curl_write_cb)) ||
            CURLE_OK != (err = curl_easy_setopt(*easyhandle, opt = CURLOPT_WRITEDATA, page)) ||
            CURLE_OK != (err = curl_easy_setopt(*easyhandle, opt = CURLOPT_PRIVATE, page)) ||
            CURLE_OK != (err = curl_easy_setopt(*easyhandle, opt = CURLOPT_HEADERFUNCTION,
```



Tester view

4th step

24h – 30h

Frontend + API


MAYBE NOTHING...

WHAT CHALLENGES HAVE WE FACED?
SAST and DAST together

One eye is good, but two are much better

WHAT CHALLENGES HAVE WE FACED?
Which one is first?

- DAST* is the 1st tool, because:
  - it is capable of covering a larger scope of reported vulnerabilities
  - it takes less effort to get started

*DAST - Dynamic Application Security Testing Tool

WHAT CHALLENGES HAVE WE FACED?

Which one is better?

Pentesting vs Bug bounty

Limited group of security researchers

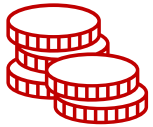
Thousands of security researchers

More formal, methodology driven

More creative

Limited in time

Continuous



Paid testing time

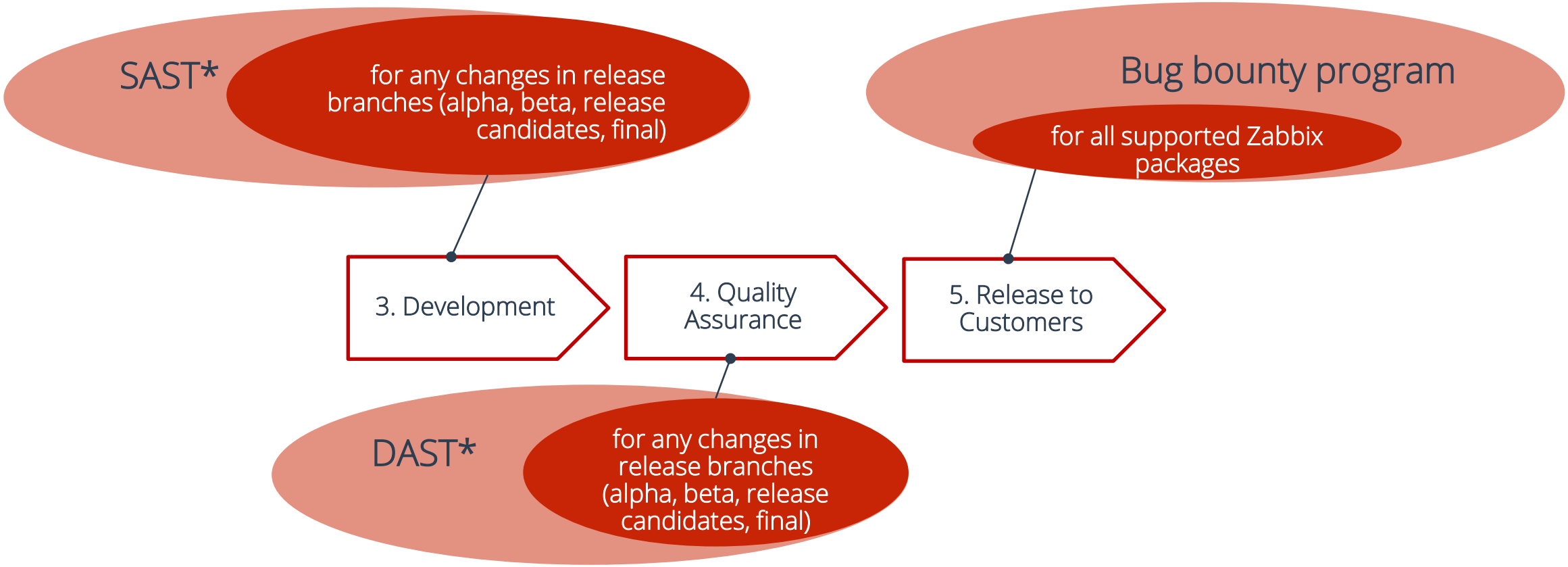
Paid vulnerabilities based on criticality

WHAT CHALLENGES HAVE WE FACED?
Pentesting vs Bug bounty

- The choice fell on the Bug bounty, because it is closer to the Zabbix culture:
  - the number of releases is quite large; hence testing should be constant
  - the openness of the code and the openness to what will be found
  - the more research, the more secure is the core.

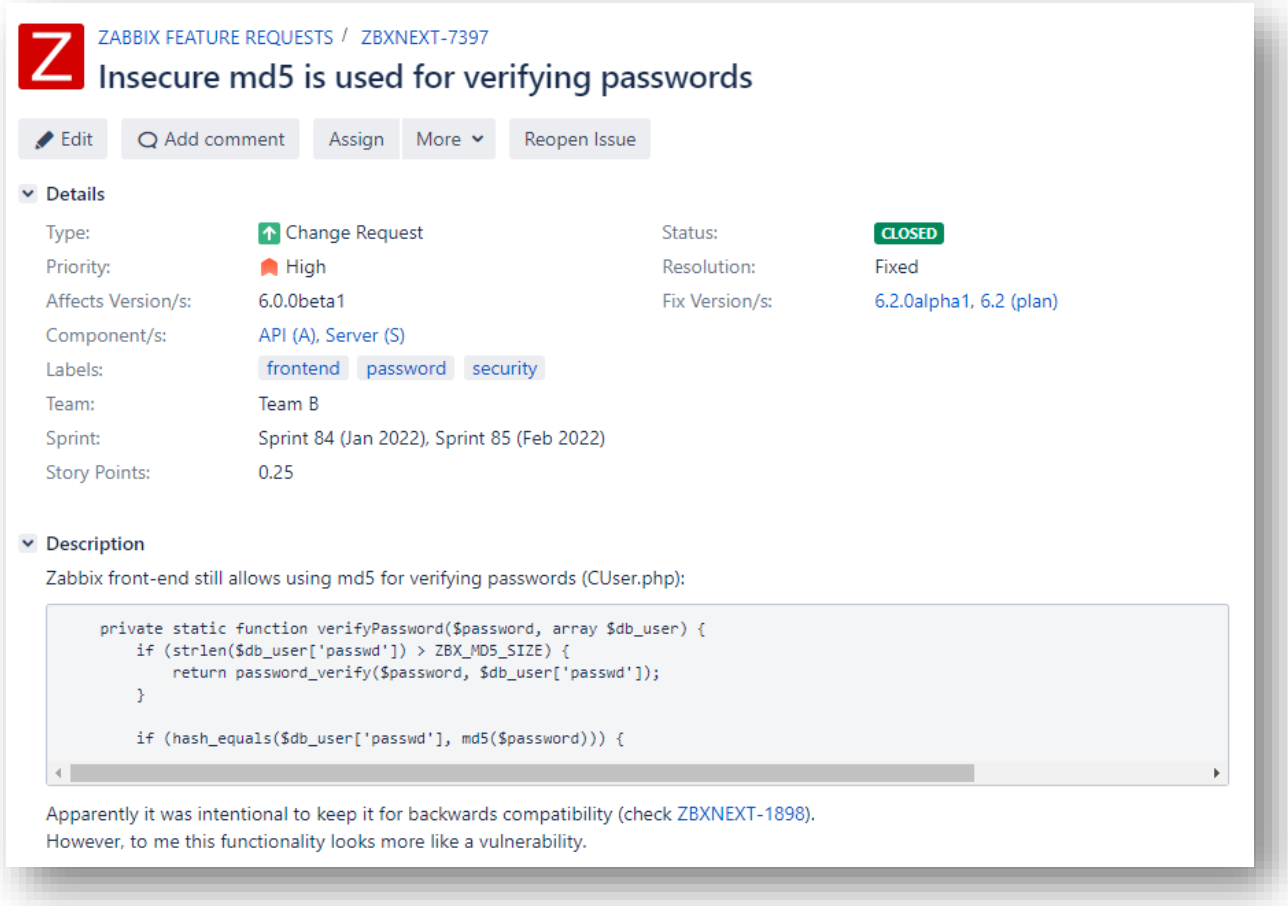
At this moment, our Bug bounty is private!

WHAT CHALLENGES HAVE WE FACED?
What we have now



***SAST** - Static Application Security Testing Tool
***DAST** - Dynamic Application Security Testing Tool

WHAT CHALLENGES HAVE WE FACED?
Vulnerability identified by SAST



ZABBIX FEATURE REQUESTS / ZBXNEXT-7397

Insecure md5 is used for verifying passwords

Edit | Add comment | Assign | More | Reopen Issue

Details

Type:	Change Request	Status:	CLOSED
Priority:	High	Resolution:	Fixed
Affects Version/s:	6.0.0beta1	Fix Version/s:	6.2.0alpha1, 6.2 (plan)
Component/s:	API (A), Server (S)		
Labels:	frontend password security		
Team:	Team B		
Sprint:	Sprint 84 (Jan 2022), Sprint 85 (Feb 2022)		
Story Points:	0.25		

Description

Zabbix front-end still allows using md5 for verifying passwords (CUser.php):

```
    private static function verifyPassword($password, array $db_user) {
        if (strlen($db_user['passwd']) > ZBX_MD5_SIZE) {
            return password_verify($password, $db_user['passwd']);
        }

        if (hash_equals($db_user['passwd'], md5($password))) {
```

Apparently it was intentional to keep it for backwards compatibility (check ZBXNEXT-1898).
However, to me this functionality looks more like a vulnerability.

***SAST** - Static Application Security Testing Tool

WHAT CHALLENGES HAVE WE FACED?

Vulnerability identified by DAST

CVE/Advisory number:	CVE-2022-40626
Synopsis:	Reflected XSS in action configuration window of Zabbix Frontend
Description:	An unauthenticated user can create a link with reflected Javascript code inside the backurl parameter and send it to other authenticated users.
Known Attack Vectors:	When prepared link with malicious code is sent to a user with privileged rights in Zabbix and the user follows the link, the XSS payload will create a fake account with predefined login, password and role in Zabbix Frontend.
Resolution:	To remediate this vulnerability, apply the updates listed in the 'Fixed Version' section to appropriate products.
Workaraunds:	The vulnerability can be exploited only by authenticated users. If an immediate update is not possible, review user access rights to your Zabbix Frontend, be attentive to browser warnings and always check any links you can receive via email or other means of communication, which lead to Zabbix Frontend and contain suspicious parameters with special symbols.
Acknowledgements:	-

Component/s	Affected version/s	Fix version/s	CVSS score	Zabbix severity	Tickets
Frontend	6.0.0-6.0.6 6.2.0	=>6.0.7rc1 =>6.2.1rc1	4.8	Medium	ZBX-21350

***DAST** - Dynamic Application Security Testing Tool

5.

Outcomes

OUTCOMES

Security has become a continues concern during all Secure SDLC*. It is changing the culture.

***SDLC** - Software Development Lifecycle

OUTCOMES

- We have become more proactive in finding vulnerabilities and detecting them early; before the product release
- Undetected vulnerabilities in SDLC* may be found by external security researchers in the bug bounty program

***SDLC** - Software Development Lifecycle

OUTCOMES

Partners and customers are first informed about vulnerabilities that needed to be fixed

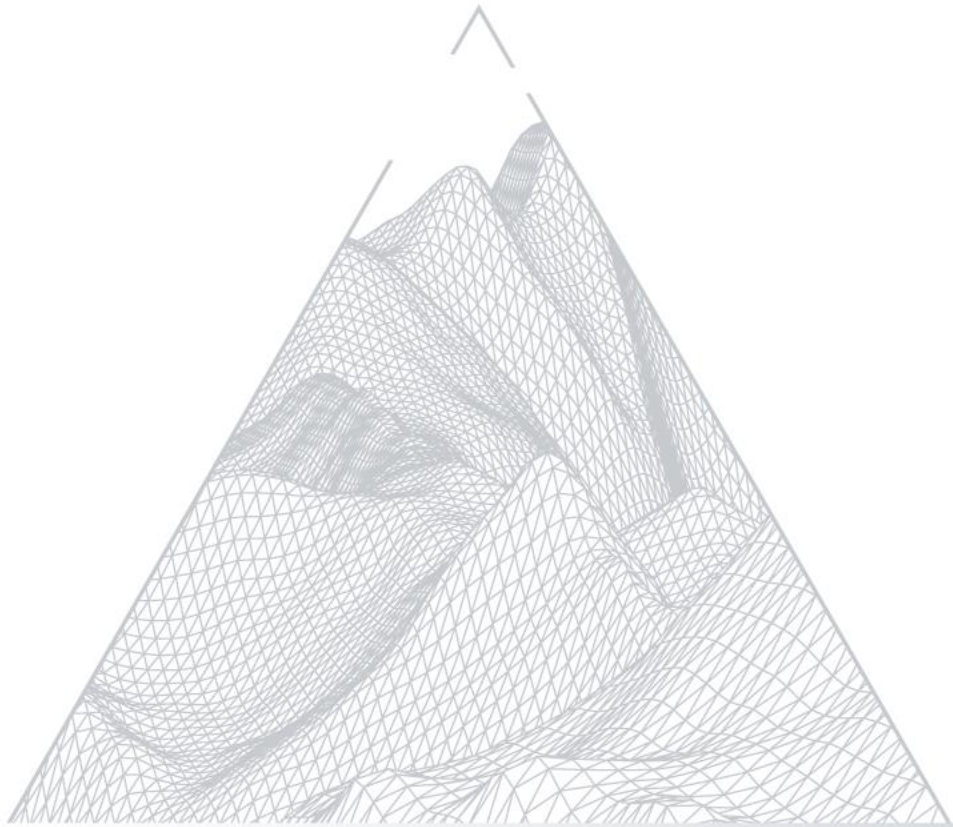
OUTCOMES

Reduces costs for users and Zabbix

OUTCOMES

By cleaning vulnerabilities up, we make Zabbix user environment more secure and less exploitable by hackers

10TH ZABBIX SUMMIT 2022



THANKS A BUNCH!