PENTEST REPORT
REQUIREMENT YOGI CLOUD

Reference: ARCANSECURITY/REQUIREMENTYOGI/1/1.2/2022
Confidential

Changelog

Date	Version	Comment	Writer	Reviewer
2022-04-26	1.0	First version	F. Pradines	F. Pradines
2022-04-28	1.1	Minor changes	A. Ragot	F. Pradines
2022-07-14	1.2	Verification Audit	F. Pradines	F. Pradines

1. Table of Contents

2. Introduction

2.1. Subject of the document

This document reports the security audit results of the Confluence and Jira plugins “Requirement Yogi Cloud” and “Requirement Yogi for Jira Cloud” developed by the company Requirement Yogi. ArcanSecurity did the audit between the 19th of April 2022 and the 21st of April 2022. Requirement Yogi provided three accounts: Super Admin, Admin and User.

N.B: the results come from 3 days of audit. Thus, they may be only a subset of what an attacker with no time limit can find.
N.B.2: the results have been updated after a verification audit performed early July.

2.2. Report structure

The report is in three parts:

- A summary which presents:
    - Some scenarios and their impacts for Requirement Yogi Cloud
    - Strengths and possible improvements identified during the audit
    - A general conclusion
- A fully detailed description which presents:
    - All the vulnerabilities found and their exploitation
    - The scenarios linked to the vulnerabilities
- A roadmap which synthesizes:
    - The recommendations given in the detailed description
    - The prioritized roadmap

3. Summary of the audit

3.1. Risks

3.1.1.Risk analysis summary

The main risks which lead to this audit are:

- Data leak of a customer
- Unauthorized changes or deletion of customer’s data
- Privilege escalation
- The availability of the platform

During the audit, the auditor has determined four scenarios that could impact the company Requirement Yogi:

- A user dumps data linked to another customer or a space he does not have access to
- A user changes data linked to another customer or a space he does not have access to
- A user performs a restricted action for which he does not have the granted rights
- An attacker performs a distributed denial of service on the platform

3.1.2.Risk assessment

Each scenario is explained in the report and analyzed with the vulnerabilities discovered during the audit.

Grid

Probability of the risk		
Probability		**Description**
4	**Strong**	The environment or context of the company means that, if nothing is done, such a threat will certainly materialize in the short term.
3	**Average**	The environment and the context of the company mean that, if nothing is done, such a threat will materialize in the short term.
2	**Low**	Even in the absence of any security measure, the environment and the context mean that the probability of occurrence of such a threat, in the short or medium term, is low.
1	**Unlikely**	Regardless of any security measures, the probability of occurrence of such a threat is extremely low and negligible.

Impact of the risk		
Impact		**Description**
4	**Strong**	Unsustainable financial, legal, commercial or image impact.
3	**Average**	Significant financial, legal, commercial or image impact.
2	**Low**	Weak financial, legal, commercial or image impact.
1	**Minimal**	Financial, legal, commercial or image impact without significant impact.

Matrix

Strong	4	8	12	16
Average	3	6	9	12
Low	2	4	6	8
Unlikely	1	2	3	4
Probability **Impact**	**Minimal**	**Low**	**Average**	**Strong**

Summary

Scenario	Probability	Impact	Risk	Action to lower the risk
An attacker performs a distributed denial of service on the platform	**2**	**3**	**6**	Implement a rate-limiting system
A user changes data linked to another customer or a space he does not have access to	**1**	**4**	**4**	Harden the overall system Secure the API Make the API more consistent
A user dumps data linked to another customer or a space he does not have access to	**1**	**3**	**3**	Harden the overall system Secure the API Make the API more consistent
A user performs a restricted action for which he does not have the granted rights	**1**	**3**	**3**	Harden the overall system Secure the API

3.2. General overview

3.2.1.Strengths

Good understanding of the cybersecurity risks
Follow the guidelines from Atlassian
Secure development

3.2.2.Possible improvements

Make the API more consistent
Integrate all the good practices in the whole stack of the system
Address the current flaws

3.3. Conclusion

During the audit, no critical or high vulnerabilities were found by the auditor. By focusing on the security picture only, the plugin Requirement Yogi Cloud is at a good level. An attacker will certainly take a considerable amount of time to find and exploit a potential vulnerability in the API. Thus, an attacker would try gaining access to the system by other meanings, like stealing the AWS credentials, doing a phishing attack on employees or other.

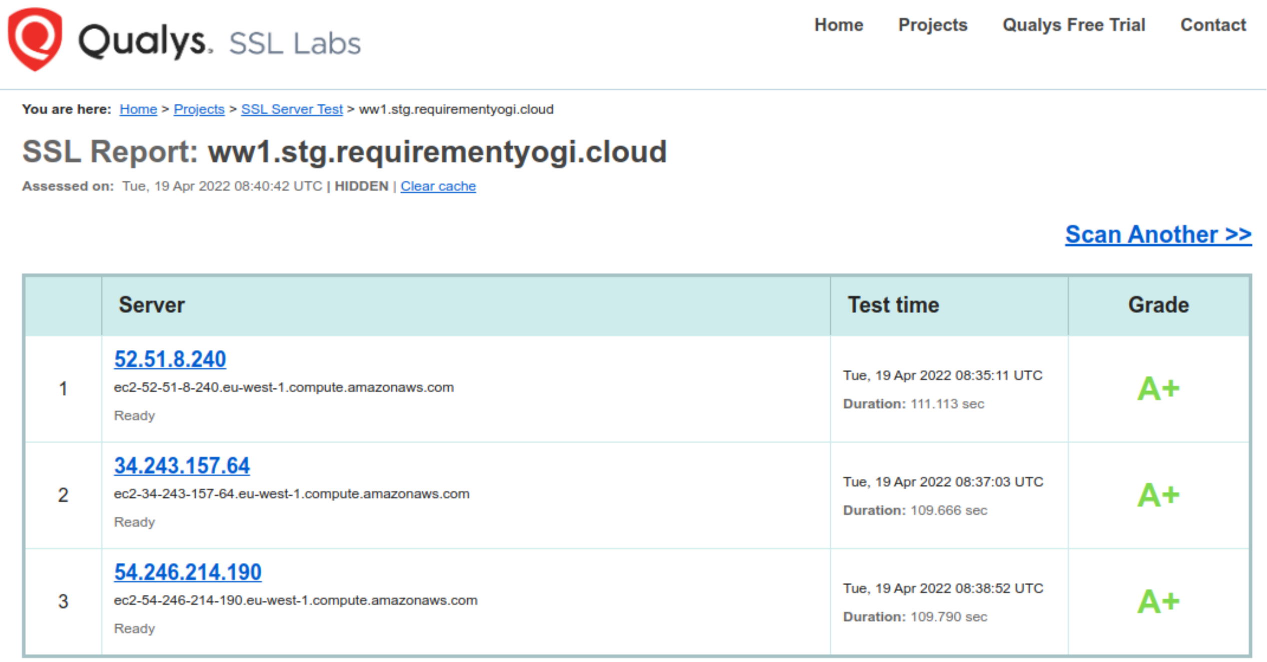
Among that, let’s not forget that on a bigger picture, the system needs some work to make the API fully consistent, less verbose, and more robust against denial of service. By taking all of the steps mentioned in the given roadmap, the probability of an attack could be reduced, and so the risk too.

4. Detailed audit

4.1. Recon

4.1.1.SSL Certificates

The auditor has started by the analysis of the SSL configuration for the domain **ww1.stg.requirementyogi.cloud** with help of the tool *ssltest* provided by *ssllabs*.



You are here: Home > Projects > SSL Server Test > ww1.stg.requirementyogi.cloud

SSL Report: ww1.stg.requirementyogi.cloud

Assessed on: Tue, 19 Apr 2022 08:40:42 UTC | HIDDEN | Clear cache

	Server	Test time	Grade
1	52.51.8.240 ec2-52-51-8-240.eu-west-1.compute.amazonaws.com Ready	Tue, 19 Apr 2022 08:35:11 UTC Duration: 111.113 sec	A+
2	34.243.157.64 ec2-34-243-157-64.eu-west-1.compute.amazonaws.com Ready	Tue, 19 Apr 2022 08:37:03 UTC Duration: 109.666 sec	A+
3	54.246.214.190 ec2-54-246-214-190.eu-west-1.compute.amazonaws.com Ready	Tue, 19 Apr 2022 08:38:52 UTC Duration: 109.790 sec	A+

Trace 1: SSL configuration ranking A+

The result is ok, with a ranking A+. This is as expected since the load balancer is managed by Amazon Web Services. Yet, the chosen configuration is not allowing for insecure algorithms.

4.1.2.Port scan

Then, the auditor did a port scan to try discovering some services.

```
% sudo nmap -p- ww1.stg.requirementyogi.cloud
Starting Nmap 7.80 ( https://nmap.org ) at 2022-04-19 14:03 CEST
Nmap scan report for ww1.stg.requirementyogi.cloud (34.243.157.64)
Host is up (0.12s latency).
Other addresses for ww1.stg.requirementyogi.cloud (not scanned): 52.51.8.240 54.246.214.190
rDNS record for 34.243.157.64: ec2-34-243-157-64.eu-west-1.compute.amazonaws.com
Not shown: 65534 filtered ports
PORT    STATE SERVICE
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 608.06 seconds
```

Trace 2: Port scan

Again, the result is as expected with only one port open. The app is behind an application load balancer (ALB) provided by AWS (Amazon Web Services).

4.1.3.HTTP headers sent by the server

Then, the auditor performed an analysis of the HTTP headers sent by the server to identify some services and eventually flaws.

```
% curl -I "https://ww1.stg.requirementyogi.cloud/"
HTTP/2 302
date: Thu, 14 Jul 2022 13:39:56 GMT
content-length: 0
server: nginx
strict-transport-security: max-age=31536000; includeSubDomains; preload
content-security-policy: default-src none; script-src 'self' https://connect-cdn.atl-paas.net
https://cdnjs.cloudflare.com https://confluence-v1.prod.atl-paas.net; font-src
https://cdnjs.cloudflare.com/ajax/libs/aui/;connect-src 'self';style-src 'unsafe-inline' 'self'
https://connect-cdn.atl-paas.net https://unpkg.com/@atlaskit/ https://cdnjs.cloudflare.com
https://confluence-v1.prod.atl-paas.net ;img-src 'self' https://*.atlassian.net
https://api.media.atlassian.com;frame-ancestors https://*.atlassian.net;
x-frame-options: DENY
location: /atlassian-connect.json
x-content-type-options: nosniff
x-xss-protection: 1; mode=block
cache-control: no-cache, no-store, max-age=0, must-revalidate
pragma: no-cache
expires: 0
referrer-policy: origin-when-cross-origin
content-language: en
```

Trace 3: HTTP headers sent by the server

We can see that almost all of the security headers are configured on the web server which is a good practice. But the auditor noticed that the “server” header is returning the name of the webserver in use: “nginx”.

R-1

It is recommended to hide the server used.

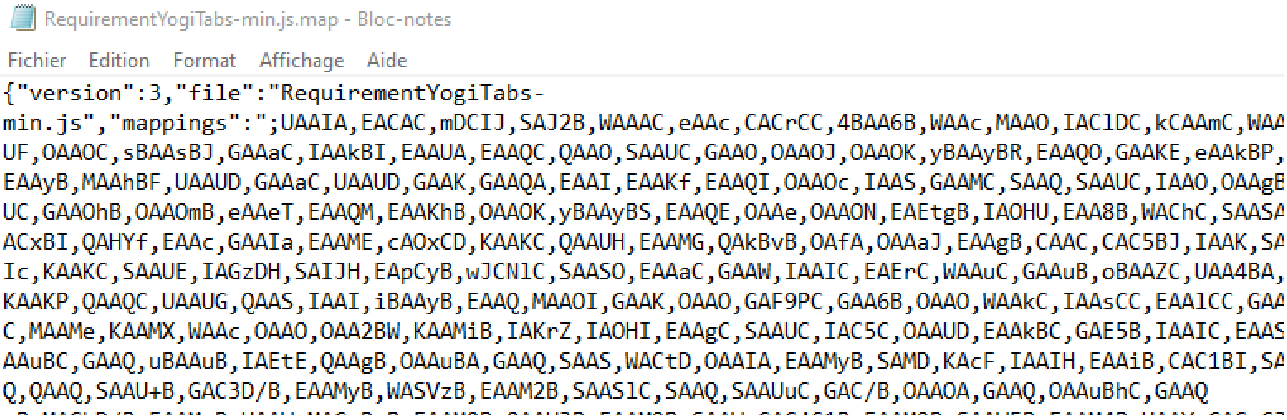
4.2. Web interface

4.2.1. Common vulnerabilities

During the pentest, the auditor tried to exploit some common vulnerabilities like SQL injections, JWT forging, XSS, CSRF… All of them were unsuccessful, which is a good point. It is as expected since the webapp is running under Spring Boot, ReactJS and is using the library provided by Atlassian for authentication.

4.2.2. JavaScript files

On the main page (https://ww1.stg.requirementyogi.cloud/web/search), the auditor noticed a javascript file (/js/RequirementYogiTabs-min.js?version=XXX). The file is minified and uglified, but the source mapping is available and accessible to anyone in the production environment.



```
{"version":3,"file":"RequirementYogiTabs-
min.js","mappings":";UAAIA,EACAC,mDCIJ,SAJ2B,WAAAC,eAAc,CACrCC,4BAA6B,WAAc,MAAO,IAC1DC,kCAAmC,WAA
UF,OAAOC,sBAAsBJ,GAAaC,IAAkBI,EAAUA,EAAQC,QAAO,SAAUC,GAAO,OAAOJ,OAAOK,yBAAyBR,EAAQO,GAAKE,eAAkBP,
EAAyB,MAAhBF,UAAUD,GAAaC,UAAUD,GAAK,GAAQA,EAAI,EAAKf,EAAQI,OAAOc,IAAS,GAAMC,SAAQ,SAAUC,IAAO,OAAgE
UC,GAAOhB,OAAOmB,eAAeT,EAAQM,EAAKhB,OAAOK,yBAAyBS,EAAQE,OAAe,OAAON,EAEtgB,IAOHU,EAA8B,WAChC,SAASA
ACxBI,QAHYf,EAAc,GAAIa,EAAME,cAOxCD,KAAKC,QAAUH,EAAMG,QAkBvB,OAfA,OAAaJ,EAAgB,CAAC,CAC5BJ,IAAK,SA
Ic,KAAKC,SAAUE,IAGzDH,SAIJH,EApCyB,wJCN1C,SAASO,EAAaC,GAAAW,IAAIC,EAErC,WAAuC,GAAuB,oBAAZC,UAA4BA,
KAAKP,QAAQC,UAAUG,QAAS,IAAI,iBAAyB,EAAQ,MAAOI,GAAK,OAAO,GAF9PC,GAA6B,OAAO,WAAkC,IAAsCC,EAA1CC,GAA
C,MAAMe,KAAMX,WAAc,OAAO,OAA2BW,KAAMiB,IAKrZ,IAOHI,EAAgC,SAAUC,IAC5C,OAAUD,EAAkBC,GAE5B,IAAIC,EAAS
AAuBC,GAAQ,uBAAuB,IAEtE,QAAgB,OAAuBA,GAAQ,SAAS,WACtD,OAAIA,EAAMyB,SAMD,KAcF,IAAIH,EAAiB,CAC1BI,SA
Q,QAAQ,SAAU+B,GAC3D/B,EAAMyB,WASVzB,EAAM2B,SAAS1C,SAAQ,SAAUuC,GAC/B,OAAOA,GAAQ,OAAuBhC,GAAQ
```

Trace 4: Map file linked to minified JavaScript file

R-2

It is recommended to restrict the access to the JavaScript mapping files.

4.3. API

The core component of Requirement Yogi Cloud is an API managing all the resources in a centralized way.

4.3.1.Multiple internal server errors

The auditor found numerous API endpoints that were crashing with an Internal Server Error instead of a Bad Request when the parameter does not meet the expected type, for instance a String instead of an Integer. Sometimes, the server can even return HTML code instead of JSON.

```
% curl
'https://ww1.stg.requirementyogi.cloud/rest/search?query=1&spaceKey=SCRUM&includeArchived=false
&limit=AAA' \
  -H 'authorization: JWT THE_TOKEN'
{"message":"The server met an unexpected
error.","httpStatus":"INTERNAL_SERVER_ERROR","timeStamp":"2022-04-
19T10:17:19.394868917","requestMethod":"GET","endpoint":"/rest/search"}

% curl 'https://ww1.stg.requirementyogi.cloud/rest/admin/queue-job?offset=1 or
1=1&jobStatus=&order=1&limit=1&eventType=&spaceKey=SCRUM' \
  -H 'authorization: JWT THE_TOKEN'
<!doctype html><html lang="en"><head><title>HTTP Status 400 – Bad Request</title><style
type="text/css">body {font-family:Tahoma,Arial,sans-serif;} h1, h2, h3, b {color:white;background-
color:#525D76;} h1 {font-size:22px;} h2 {font-size:16px;} h3 {font-size:14px;} p {font-size:12px;} a
{color:black;} .line {height:1px;background-
color:#525D76;border:none;}</style></head><body><h1>HTTP Status 400 – Bad
Request</h1></body></html>
```

Trace 5: Internal Server Errors with crafted parameters

The same behavior was observed on some endpoints when the user does not have permissions to create, read, update, or delete a given resource.

```
% curl 'https://ww1.stg.requirementyogi.cloud/rest/traceability/SCRUM/saved-queries/137' \
  -X 'DELETE' \
  -H 'authorization: JWT THE_TOKEN' \
  -H 'content-type: application/json;charset=UTF-8' \
{"message":"The server met an unexpected
error.","httpStatus":"INTERNAL_SERVER_ERROR","timeStamp":"2022-04-
19T14:52:03.689527824","requestMethod":"DELETE","endpoint":"/rest/traceability/SCRUM/saved-
queries/137"}
```

Trace 6: Internal Server Errors on ungranted resources

Here, the user has been granted read permission on the saved query because it was shared with him, but he cannot remove it. The endpoint should have returned a FORBIDDEN http result.
One of the risks is to miss some real bugs in logs because of these “fake” uncaught exceptions.

R-3

It is recommended to handle the exceptions on the API instead of letting the endpoint crash.

4.3.2.Authentication token scope

Depending on the context, Atlassian (which is the authority taking care of the authentication) is sending a JWT with a context or not. As an example, when the URL of the page contains a space, a JWT will be generated to be used in that context, and when you are in the administration panel of the plugin, a more general JWT is given.

The application doesn’t consider the context and thus, allows a JWT generated for a particular action to be used on another context (like the administration panel), which breaks the concept of “attack surface reduction” provided by Atlassian.

JWT 1: generated in the SCRUM space

```
{
  "sub": "625d66d1ab7a1800708a419e",
  "qsh": "context-qsh",
  "iss": "974b8b0d-7338-3db9-bf6e-209ff6e98ff4",
  "context": {
    "license": {
      "active": true
    },
    "confluence": {
      "space": {
        "key": "SCRUM",
        "id": "5898250"
      },
      "content": {
        "plugin": "ac:ry-cloud:requirements",
        "type": "custom"
      }
    }
  },
  "exp": 1650448868,
  "iat": 1650447968
}
```

JWT 2: generated in the general context

```
{
  "sub": "6258104a0630bd007076b845",
  "qsh": "context-qsh",
  "iss": "974b8b0d-7338-3db9-bf6e-
209ff6e98ff4",
  "context": {
    "license": {
      "active": true
    }
  },
  "exp": 1650881800,
  "iat": 1650880900
}
```

Trace 7: different JWT for different contexts

```
% curl 'https://ww1.stg.requirementyogi.cloud/rest/admin/queue-job?offset=0&jobStatus=&order=&limit=10&eventType=&spaceKey=HOME' \
  -H 'authorization: JWT THE_TOKEN'
{"results":[{"spaceKey":"HOME", ...},
...],"offset":0,"limit":10,"total":15,"humanReadable":null,"sql":null,"nextPageOffset":11}
```

Trace 8: using context-space JWT on general-context endpoint

R-4

It is recommended to verify the context of the token to reduce the scope of an attacker in case the JWT is compromised.

4.3.3.Expired or invalid license

As a paid plugin with a subscription, Requirement Yogi Cloud get the license status from Atlassian. The auditor noticed that when a JWT is generated, even if the license switches its state from *‘licensed’* to *‘unlicensed’*, this state is ignored, and the API continues to serve the data.

```
% curl 'https://ww1.stg.requirementyogi.cloud/rest/search?query=&spaceKey=SCRUM&includeArchived=false' \
  -H 'authorization: JWT THE_TOKEN'
{...}
```

Trace 9: API accepts requests even when the plugin is not licensed

R-5

It is recommended to verify the state of the license to avoid unauthorized use of Requirement Cloud Yogi.

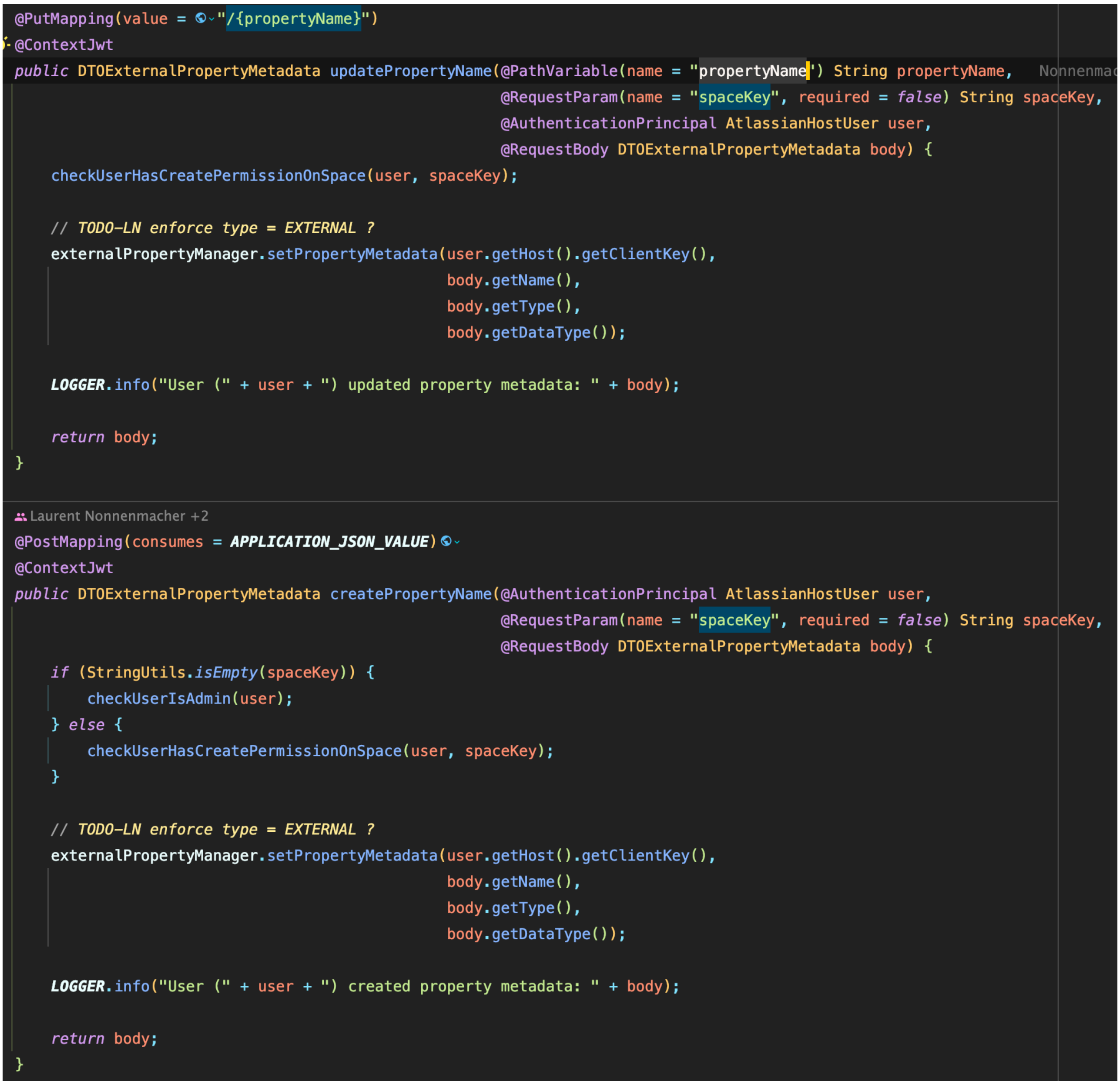
4.4. (Bonus) Quick source code audit

The company Requirement Yogi allowed us to use some time primarily planned for the pentest, to perform a quick source code audit to confirm the vulnerability found by the auditor and try to identify more flaws.

4.4.1.Endpoints doing the same action with not the same permissions

External properties have a PUT endpoint to create and a POST endpoint to update. Yet, there are two problems:

- First one is that the two endpoints do the exact same thing
- But they are not doing the same permissions verification



```
@PutMapping(value = "/{propertyName}")
@ContextJwt
public DTOExternalPropertyMetadata updatePropertyName(@PathVariable(name = "propertyName") String propertyName,   Nonnenmac
                                                      @RequestParam(name = "spaceKey", required = false) String spaceKey,
                                                      @AuthenticationPrincipal AtlassianHostUser user,
                                                      @RequestBody DTOExternalPropertyMetadata body) {
    checkUserHasCreatePermissionOnSpace(user, spaceKey);

    // TODO-LN enforce type = EXTERNAL ?
    externalPropertyManager.setPropertyMetadata(user.getHost().getClientKey(),
                                                body.getName(),
                                                body.getType(),
                                                body.getDataType());

    LOGGER.info("User (" + user + ") updated property metadata: " + body);

    return body;
}

Laurent Nonnenmacher +2
@PostMapping(consumes = APPLICATION_JSON_VALUE)
@ContextJwt
public DTOExternalPropertyMetadata createPropertyName(@AuthenticationPrincipal AtlassianHostUser user,
                                                      @RequestParam(name = "spaceKey", required = false) String spaceKey,
                                                      @RequestBody DTOExternalPropertyMetadata body) {
    if (StringUtils.isEmpty(spaceKey)) {
        checkUserIsAdmin(user);
    } else {
        checkUserHasCreatePermissionOnSpace(user, spaceKey);
    }

    // TODO-LN enforce type = EXTERNAL ?
    externalPropertyManager.setPropertyMetadata(user.getHost().getClientKey(),
                                                body.getName(),
                                                body.getType(),
                                                body.getDataType());

    LOGGER.info("User (" + user + ") created property metadata: " + body);

    return body;
}
```

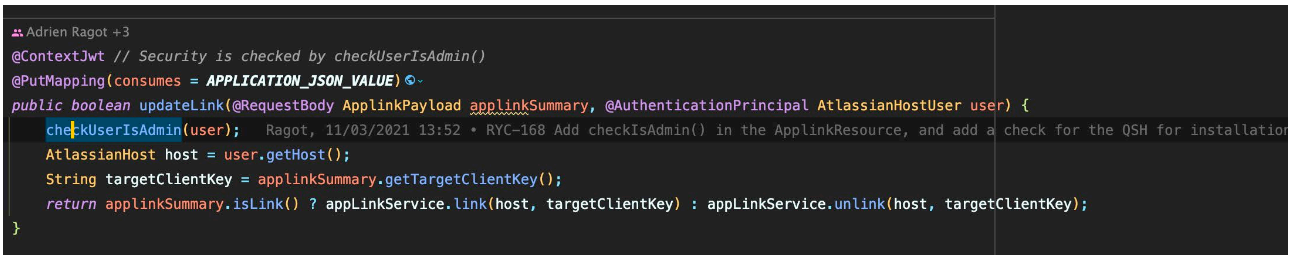
Trace 10: same logic with different permissions

R-6

It is recommended to verify the permissions on the endpoints and the logic applied on them.

4.4.2.Create annotations for permissions

Almost all endpoints are protected with checks of the user’s permission. This check is generally the first thing executed on the endpoint method. This could be improved by using an annotation instead to keep clarity with *@ContextJwt* that is used the same way.



```
Adrien Ragot +3
@ContextJwt // Security is checked by checkUserIsAdmin()
@PutMapping(consumes = APPLICATION_JSON_VALUE)
public boolean updateLink(@RequestBody ApplinkPayload applinkSummary, @AuthenticationPrincipal AtlassianHostUser user) {
    checkUserIsAdmin(user);   Ragot, 11/03/2021 13:52 • RYC-168 Add checkIsAdmin() in the ApplinkResource, and add a check for the QSH for installatior
    AtlassianHost host = user.getHost();
    String targetClientKey = applinkSummary.getTargetClientKey();
    return applinkSummary.isLink() ? appLinkService.link(host, targetClientKey) : appLinkService.unlink(host, targetClientKey);
}
```

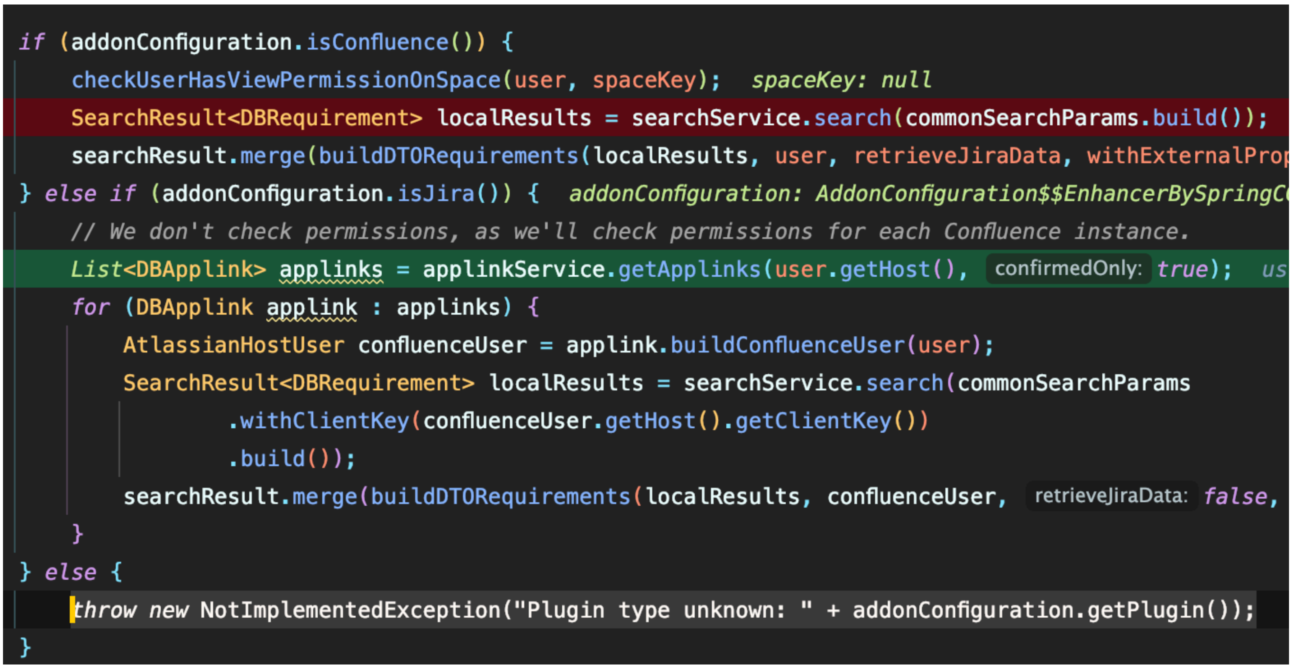
Trace 11: code checking the user’s permission

It is recommended to use annotations for authentication and permissions purposes.

R-7

4.4.3.Exceptions in controller

Earlier in the report, we talked about endpoints that were returning some Internal Server Error instead of a Bad Request, Forbidden… Below is an example in the code of an exception that is thrown instead of returning a Bad Request.

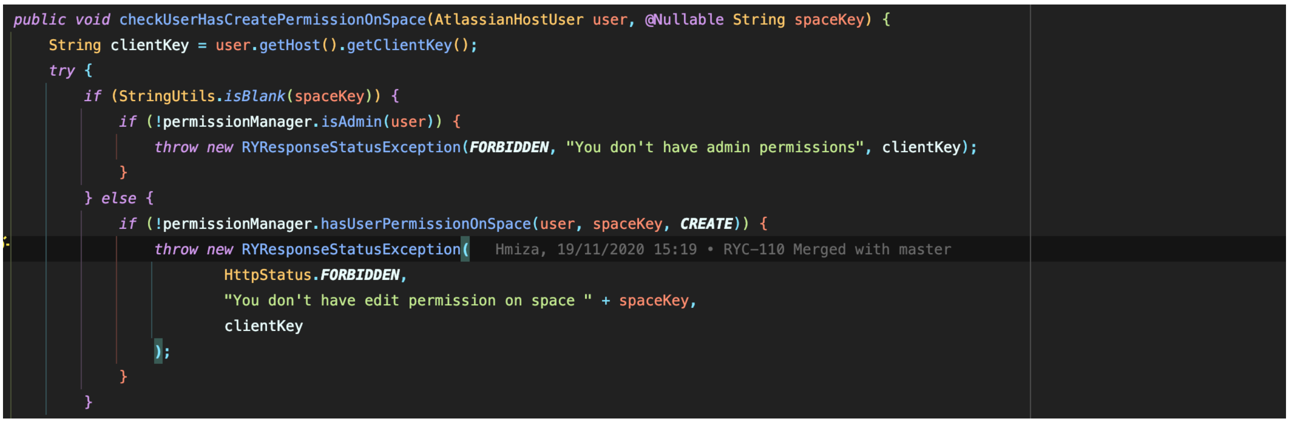


```
if (addonConfiguration.isConfluence()) {
    checkUserHasViewPermissionOnSpace(user, spaceKey);   spaceKey: null
    SearchResult<DBRequirement> localResults = searchService.search(commonSearchParams.build());
    searchResult.merge(buildDTORequirements(localResults, user, retrieveJiraData, withExternalProp
} else if (addonConfiguration.isJira()) {  addonConfiguration: AddonConfiguration$$EnhancerBySpringC
    // We don't check permissions, as we'll check permissions for each Confluence instance.
    List<DBApplink> applinks = applinkService.getApplinks(user.getHost(),  confirmedOnly: true);   us
    for (DBApplink applink : applinks) {
        AtlassianHostUser confluenceUser = applink.buildConfluenceUser(user);
        SearchResult<DBRequirement> localResults = searchService.search(commonSearchParams
                .withClientKey(confluenceUser.getHost().getClientKey())
                .build());
        searchResult.merge(buildDTORequirements(localResults, confluenceUser,  retrieveJiraData: false,
    }
} else {
    throw new NotImplementedException("Plugin type unknown: " + addonConfiguration.getPlugin());
}
```

Trace 12: Exception thrown in controller

4.4.4.Incorrect error message

The method used to check if the user has the create permission on a space may check if the user is admin if no space key was provided. In this check, if the user is not an administrator, the returned message says that the user does not have the admin permissions. This is correct, but for a developer or an implementer, it may be disturbing and difficult to understand that the user can indeed have the permission if he provides a space key.



```
public void checkUserHasCreatePermissionOnSpace(AtlassianHostUser user, @Nullable String spaceKey) {
    String clientKey = user.getHost().getClientKey();
    try {
        if (StringUtils.isBlank(spaceKey)) {
            if (!permissionManager.isAdmin(user)) {
                throw new RYResponseStatusException(FORBIDDEN, "You don't have admin permissions", clientKey);
            }
        } else {
            if (!permissionManager.hasUserPermissionOnSpace(user, spaceKey, CREATE)) {
                throw new RYResponseStatusException(   Hmiza, 19/11/2020 15:19 • RYC-110 Merged with master
                        HttpStatus.FORBIDDEN,
                        "You don't have edit permission on space " + spaceKey,
                        clientKey
                );
            }
        }
```

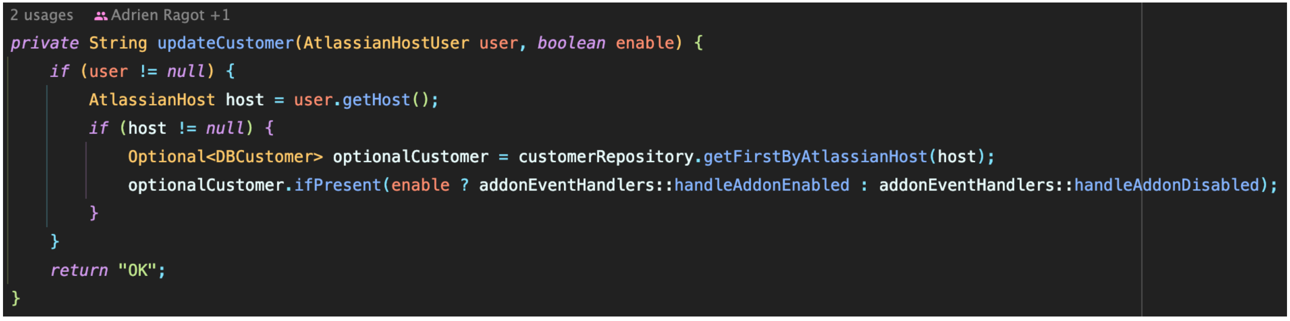
Trace 13: admin error message on check user permission

It is recommended to make the API return consistent error messages.

R-8

4.4.5.Endpoint returns a success even in case of failure

There are some endpoints that are returning a success, even if the routine was not executed due to insufficient permissions or other. Thus, a developer can think that everything web smooth whereas it is not the case. This can lead to inconsistent data or security issues.

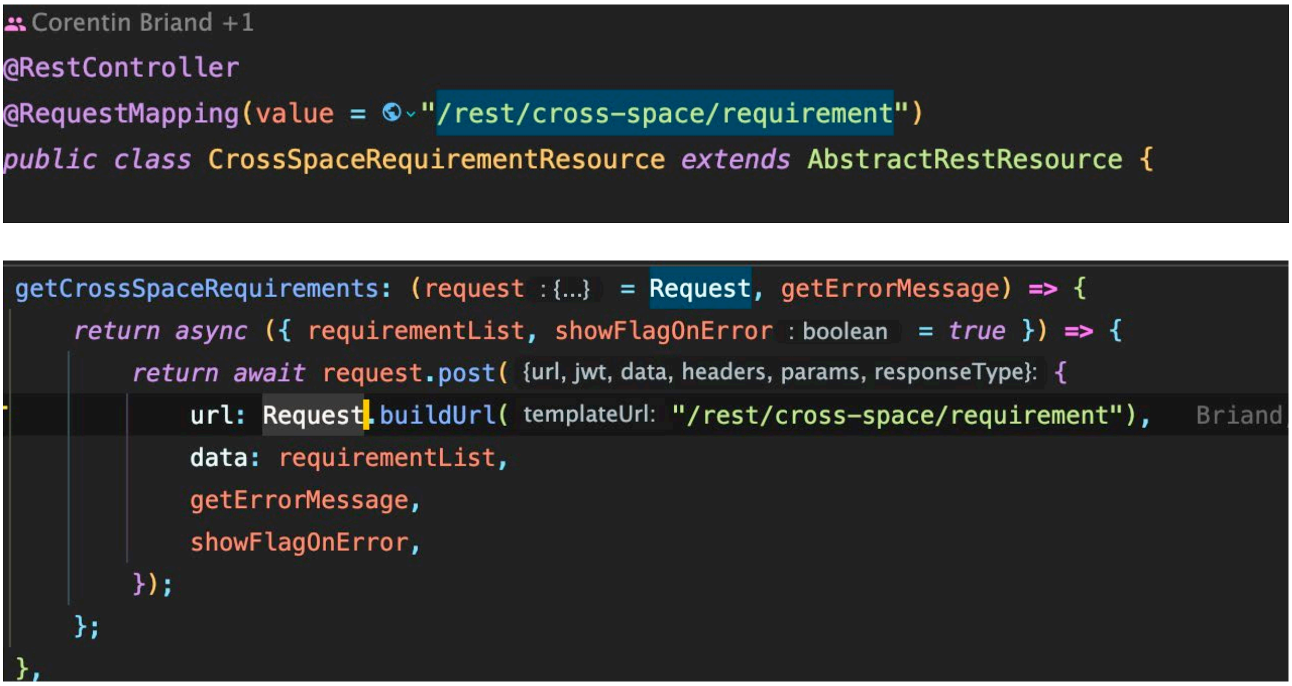


```
2 usages   Adrien Ragot +1
private String updateCustomer(AtlassianHostUser user, boolean enable) {
    if (user != null) {
        AtlassianHost host = user.getHost();
        if (host != null) {
            Optional<DBCustomer> optionalCustomer = customerRepository.getFirstByAtlassianHost(host);
            optionalCustomer.ifPresent(enable ? addonEventHandlers::handleAddonEnabled : addonEventHandlers::handleAddonDisabled);
        }
    }
    return "OK";
}
```

Trace 14: endpoint returns success even in case of failure

4.4.6.URLs duplicated between Spring Boot and ReactJS

ReactJS uses the endpoint defined by Spring Boot. Problem is that these URLs are not shared in a common way but are rather duplicated from Spring Boot into ReactJS. If a route changes in Spring Boot and it is not updated in the ReactJS code, it will lead to incorrect paths.



```
Corentin Briand +1
@RestController
@RequestMapping(value = "/rest/cross-space/requirement")
public class CrossSpaceRequirementResource extends AbstractRestResource {
```

```
getCrossSpaceRequirements: (request :{...} = Request, getErrorMessage) => {
    return async ({ requirementList, showFlagOnError : boolean = true }) => {
        return await request.post( {url, jwt, data, headers, params, responseType}: {
            url: Request.buildUrl( templateUrl: "/rest/cross-space/requirement"),    Briand
            data: requirementList,
            getErrorMessage,
            showFlagOnError,
        });
    };
},
```

Trace 15: URLs duplicated

R-9

It is recommended to implement a system that allows sharing the URLs between the different parts of the stack.

4.4.7.No rate-limiting

The API is not protected by a rate-limiting system to avoid consuming too many resources on the backend side. A system like this would secure the database and API from intensive and repetitive computations if a user stresses it.

R-10

It is recommended to implement a rate-limiting system.

4.5. Exploitation scenarios

Four main scenarios were determined prior to the audit:

- User dumps data linked to another customer or a space he does not have access to
- User changes data linked to another customer or a space he does not have access to
- User performs a restricted action for which he does not have the granted rights
- An attacker performs a distributed denial of service on the platform

For all these scenarios, the auditor was not able to do a privilege escalation. This results in a good maturity for the system when we talk about security.

The main risk here is a broken logic or an error while checking permissions, which could let an attacker exploit the two first scenarios.

The second risk is the use of many dependencies. If a library is vulnerable at some points (like Log4Shell vulnerability), the app would eventually suffer from it and could lead to a leak of the secrets, reverse shell, and so on…

Last but not least, the lack of protection against DDOS attack could affect the availability of the platform.

5. Roadmap

5.1. Summary of the recommendations

The recommendations shown here are in the order of appearance in the document.

R-1	It is recommended to hide the server used.
R-2	It is recommended to restrict the access to the JavaScript mapping files.
R-3	It is recommended to handle the exceptions on the API instead of letting the endpoint crash.
R-4	It is recommended to verify the context of the token to reduce the scope of an attacker in case the JWT is compromised.
R-5	It is recommended to verify the state of the license to avoid unauthorized use of Requirement Cloud Yogi.
R-6	It is recommended to verify the permissions on the endpoints and the logic applied on them.
R-7	It is recommended to use annotations for authentication and permissions purposes.
R-8	It is recommended to make the API return consistent error messages.
R-9	It is recommended to implement a system that allows sharing the URLs between the different parts of the stack.
R-10	It is recommended to implement a rate-limiting system.

Legend:

🟩 Level 1: notification only. This can be or not be implemented.
🟦 Level 2: this should be implemented on a medium frame period.
🟧 Level 3: this should be fixed on a short frame period.
🟥 Level 4: this should be fixed immediately.

5.2. Summary of the roadmap

The following roadmap is shown in the prioritized order recommended by ArcanSecurity. This roadmap considers the criticality of the vulnerabilities, the difficulty required to exploit them and the complexity of the fix. By adding all this, the most critical and easy to fix actions appear at the top.

Action	Description	Linked recommendations
A-1	Harden the overall system	**R-1, R-2**
A-2	Secure the API	**R-4, R-5, R-6, R-7**
A-3	Make the API more consistent	**R-3, R-8, R-9**
A-4	Implement a rate-limiting system	**R-10**

5.3. Verification audit

Once the roadmap is implemented, we recommend that you carry out a verification audit in order to verify that the fixes are correctly implemented, and they did not introduce other flaws.