CS 6035

Projects / Log4Shell / Intro Flag

Intro Flag

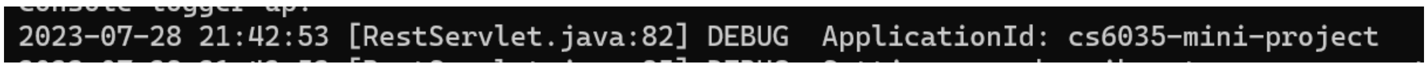
Now that we are set up, let's do a quick rundown of Log4j, how it works at a high level, and test that we are able to successfully call and exploit the application.

As you should know from the background and resource's section, Log4j is an application logging library that outputs user defined program information. An example log statement would look like the following:

```
static Logger log = LogManager.getLogger(RestServlet.class.getName());
log.debug("ApplicationId: {}", applicationId);
```

The log statement above as you can see, defines the class, log level of the message and the actual message. Notice that we are injecting user input into the log message. This is where our vulnerability is.

To be more specific, here is what the output of the message actually will look like:



```
2023-07-28 21:42:53 [RestServlet.java:82] DEBUG  ApplicationId: cs6035-mini-project
```

Now we can map the code above to the logged message. The date/time is defined in configuration files which are out of the scope of this project. Next is where our code starts to map. We have [Classname.java:LineNumber]. This is helpful to know exactly what part of your code is executing and where.

Next, we see DEBUG. This is what is called the log level. Log levels are used for the amount of output you want your application to log, or even to classify errors different from informative messages. The typical log levels are DEBUG, INFO, WARN, ERROR, FATAL. To further explain, if a log level is set to say ERROR, your application will not log anything on WARN, INFO, or DEBUG level. This application is set to DEBUG.

Finally, we get to the actual logged message. As you can see, we log the constant text as well as the injected variable applicationId, which is our applicationId. This is the most important part you will want to pay attention to. Throughout this project, you will try to find messages that log user defined input and inject your malicious string into it.

Now, let’s have some fun and get familiar with the application. To do so, we will call the application normally and then lookup the java version on the application server.

To start we can run a simple inquiry to the services /isAlive endpoint and see what we get back from the logs and see if we can find anything that is exploitable.

GATECH_ID IS A REQUIRED HEADER

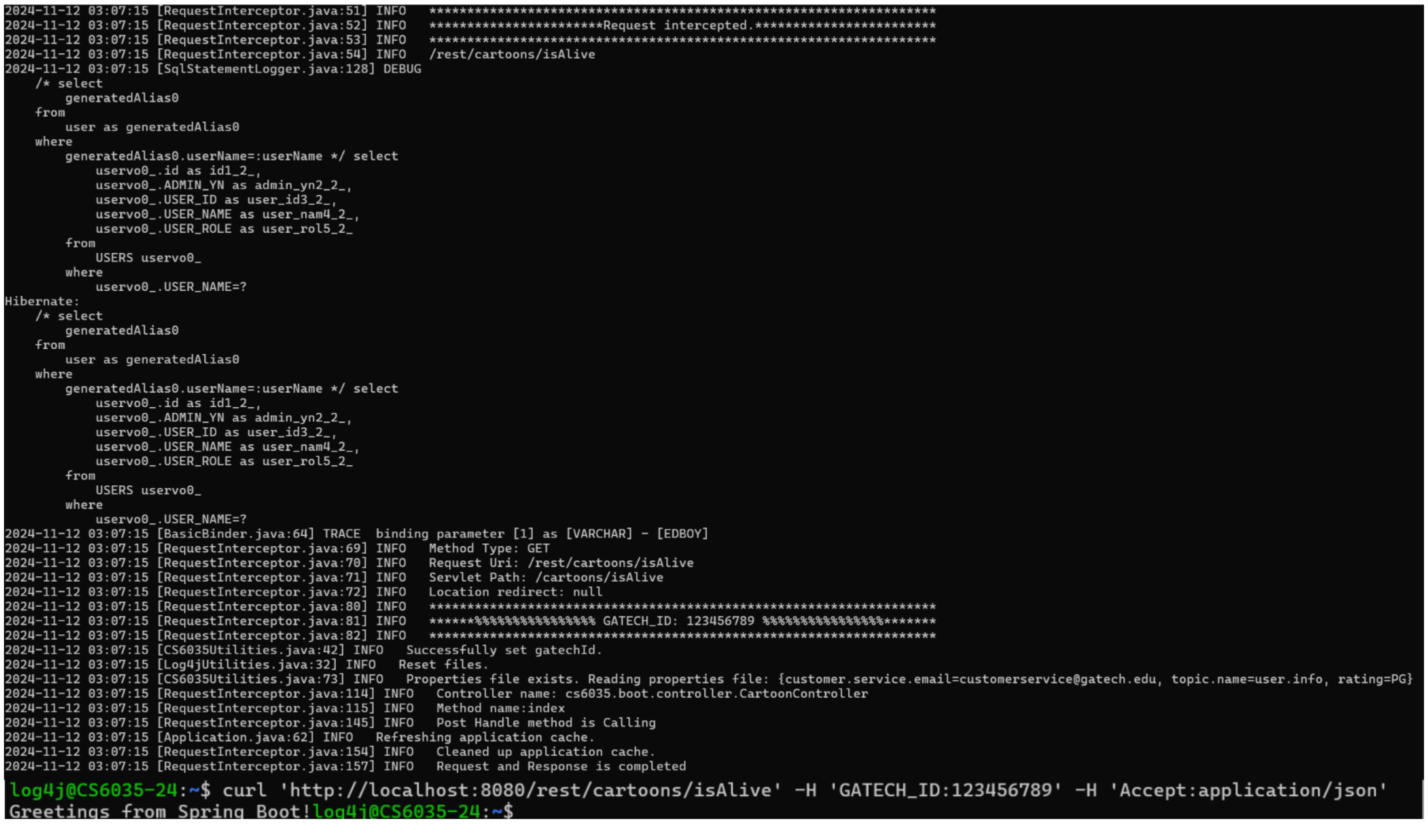
Open a new terminal and run:

```
curl 'http://localhost:8080/rest/cartoons/isAlive' -H 'GATECH_ID:123456789' -H 'Accept:applicatio
```

You should see your logs log some messages in the log tailing terminal window. Let’s inspect it.

Go back to the terminal window you are tailing the logs in. When the server intercepts a request, it logs “Request intercepted” to alert the user where the request starts. As we can see, there is not much going on in terms of useful information, but we can see the service did log a message that could be exploitable.

Voila! In the highlighted message, we see that it is logging the Method Type: GET, the URL: /rest/cartoons/isAlive, and some headers that are being logged. This is a good indicator that we can exploit this service by sending lookups through a header.



```
2024-11-12 03:07:15 [RequestInterceptor.java:51] INFO   ******************************************************************
2024-11-12 03:07:15 [RequestInterceptor.java:52] INFO   **********************Request intercepted.***********************
2024-11-12 03:07:15 [RequestInterceptor.java:53] INFO   ******************************************************************
2024-11-12 03:07:15 [RequestInterceptor.java:54] INFO   /rest/cartoons/isAlive
2024-11-12 03:07:15 [SqlStatementLogger.java:128] DEBUG
    /* select
        generatedAlias0
    from
        user as generatedAlias0
    where
        generatedAlias0.userName=:userName */ select
            uservo0_.id as id1_2_,
            uservo0_.ADMIN_YN as admin_yn2_2_,
            uservo0_.USER_ID as user_id3_2_,
            uservo0_.USER_NAME as user_nam4_2_,
            uservo0_.USER_ROLE as user_rol5_2_
        from
            USERS uservo0_
        where
            uservo0_.USER_NAME=?
Hibernate:
    /* select
        generatedAlias0
    from
        user as generatedAlias0
    where
        generatedAlias0.userName=:userName */ select
            uservo0_.id as id1_2_,
            uservo0_.ADMIN_YN as admin_yn2_2_,
            uservo0_.USER_ID as user_id3_2_,
            uservo0_.USER_NAME as user_nam4_2_,
            uservo0_.USER_ROLE as user_rol5_2_
        from
            USERS uservo0_
        where
            uservo0_.USER_NAME=?
2024-11-12 03:07:15 [BasicBinder.java:64] TRACE  binding parameter [1] as [VARCHAR] - [EDBOY]
2024-11-12 03:07:15 [RequestInterceptor.java:69] INFO   Method Type: GET
2024-11-12 03:07:15 [RequestInterceptor.java:70] INFO   Request Uri: /rest/cartoons/isAlive
2024-11-12 03:07:15 [RequestInterceptor.java:71] INFO   Servlet Path: /cartoons/isAlive
2024-11-12 03:07:15 [RequestInterceptor.java:72] INFO   Location redirect: null
2024-11-12 03:07:15 [RequestInterceptor.java:80] INFO   ******************************************************************
2024-11-12 03:07:15 [RequestInterceptor.java:81] INFO   ******%%%%%%%%%%%%%%%%%% GATECH_ID: 123456789 %%%%%%%%%%%%%%%%%%*******
2024-11-12 03:07:15 [RequestInterceptor.java:82] INFO   ******************************************************************
2024-11-12 03:07:15 [CS6035Utilities.java:42] INFO   Successfully set gatechId.
2024-11-12 03:07:15 [Log4jUtilities.java:32] INFO   Reset files.
2024-11-12 03:07:15 [CS6035Utilities.java:73] INFO   Properties file exists. Reading properties file: {customer.service.email=customerservice@gatech.edu, topic.name=user.info, rating=PG}
2024-11-12 03:07:15 [RequestInterceptor.java:114] INFO   Controller name: cs6035.boot.controller.CartoonController
2024-11-12 03:07:15 [RequestInterceptor.java:115] INFO   Method name:index
2024-11-12 03:07:15 [RequestInterceptor.java:145] INFO   Post Handle method is Calling
2024-11-12 03:07:15 [Application.java:62] INFO   Refreshing application cache.
2024-11-12 03:07:15 [RequestInterceptor.java:154] INFO   Cleaned up application cache.
2024-11-12 03:07:15 [RequestInterceptor.java:157] INFO   Request and Response is completed
log4j@CS6035-24:~$ curl 'http://localhost:8080/rest/cartoons/isAlive' -H 'GATECH_ID:123456789' -H 'Accept:application/json'
Greetings from Spring Boot!log4j@CS6035-24:~$
```

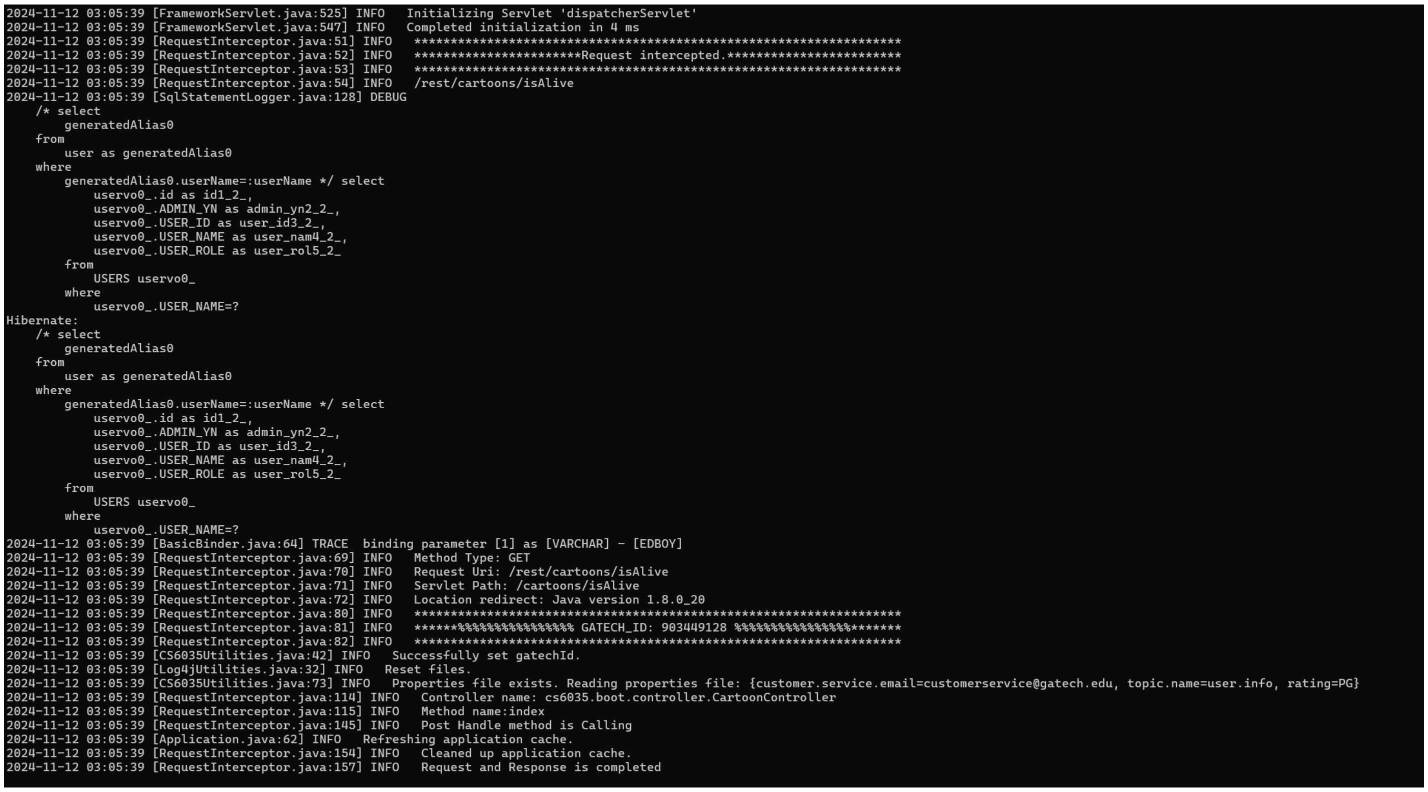
**Note: You can zoom in by using ctrl + scroll

Take some time to inspect the logged messages and try to understand what the program is doing and the flow of it.

Lets try to get the java version on the server now.

The checked headers for this application are content-type, location, and X-NetworkUserId, although not all are always checked/exploitable.

Construct a malicious payload using one of the logged headers that will return the java version of the host of the web application. You should see something like the screenshot below if successful:



```
2024-11-12 03:05:39 [FrameworkServlet.java:525] INFO   Initializing Servlet 'dispatcherServlet'
2024-11-12 03:05:39 [FrameworkServlet.java:547] INFO   Completed initialization in 4 ms
2024-11-12 03:05:39 [RequestInterceptor.java:51] INFO   *******************************************************************
2024-11-12 03:05:39 [RequestInterceptor.java:52] INFO   ***********************Request intercepted.************************
2024-11-12 03:05:39 [RequestInterceptor.java:53] INFO   *******************************************************************
2024-11-12 03:05:39 [RequestInterceptor.java:54] INFO   /rest/cartoons/isAlive
2024-11-12 03:05:39 [SqlStatementLogger.java:128] DEBUG
    /* select
        generatedAlias0
    from
        user as generatedAlias0
    where
        generatedAlias0.userName=:userName */ select
            uservo0_.id as id1_2_,
            uservo0_.ADMIN_YN as admin_yn2_2_,
            uservo0_.USER_ID as user_id3_2_,
            uservo0_.USER_NAME as user_nam4_2_,
            uservo0_.USER_ROLE as user_rol5_2_
        from
            USERS uservo0_
        where
            uservo0_.USER_NAME=?
Hibernate:
    /* select
        generatedAlias0
    from
        user as generatedAlias0
    where
        generatedAlias0.userName=:userName */ select
            uservo0_.id as id1_2_,
            uservo0_.ADMIN_YN as admin_yn2_2_,
            uservo0_.USER_ID as user_id3_2_,
            uservo0_.USER_NAME as user_nam4_2_,
            uservo0_.USER_ROLE as user_rol5_2_
        from
            USERS uservo0_
        where
            uservo0_.USER_NAME=?
2024-11-12 03:05:39 [BasicBinder.java:64] TRACE  binding parameter [1] as [VARCHAR] - [EDBOY]
2024-11-12 03:05:39 [RequestInterceptor.java:69] INFO   Method Type: GET
2024-11-12 03:05:39 [RequestInterceptor.java:70] INFO   Request Uri: /rest/cartoons/isAlive
2024-11-12 03:05:39 [RequestInterceptor.java:71] INFO   Servlet Path: /cartoons/isAlive
2024-11-12 03:05:39 [RequestInterceptor.java:72] INFO   Location redirect: Java version 1.8.0_20
2024-11-12 03:05:39 [RequestInterceptor.java:80] INFO   *******************************************************************
2024-11-12 03:05:39 [RequestInterceptor.java:81] INFO   ******%%%%%%%%%%%%%%%% GATECH_ID: 903449128 %%%%%%%%%%%%%%%%*******
2024-11-12 03:05:39 [RequestInterceptor.java:82] INFO   *******************************************************************
2024-11-12 03:05:39 [CS6035Utilities.java:42] INFO   Successfully set gatechId.
2024-11-12 03:05:39 [Log4jUtilities.java:32] INFO   Reset files.
2024-11-12 03:05:39 [CS6035Utilities.java:73] INFO   Properties file exists. Reading properties file: {customer.service.email=customerservice@gatech.edu, topic.name=user.info, rating=PG}
2024-11-12 03:05:39 [RequestInterceptor.java:114] INFO   Controller name: cs6035.boot.controller.CartoonController
2024-11-12 03:05:39 [RequestInterceptor.java:115] INFO   Method name:index
2024-11-12 03:05:39 [RequestInterceptor.java:145] INFO   Post Handle method is Calling
2024-11-12 03:05:39 [Application.java:62] INFO   Refreshing application cache.
2024-11-12 03:05:39 [RequestInterceptor.java:154] INFO   Cleaned up application cache.
2024-11-12 03:05:39 [RequestInterceptor.java:157] INFO   Request and Response is completed
```

**Note: You might need to scroll up to see this output.

Do you see the same output? LIGHTWEIGHT BABY! Muahaha! If not, try to research the log4shell exploit more and learn how to exploit the lookup.

Our hunch was correct and we have successfully exploited the vulnerability. You can play around with this if you like and see what other lookups you can perform. It is possible to lookup system settings, environment variables and much more just with this.

Be sure to save your work outside of the VM in case the VM crashes or some other unforeseen issue arises. This will ensure you are not losing your work.