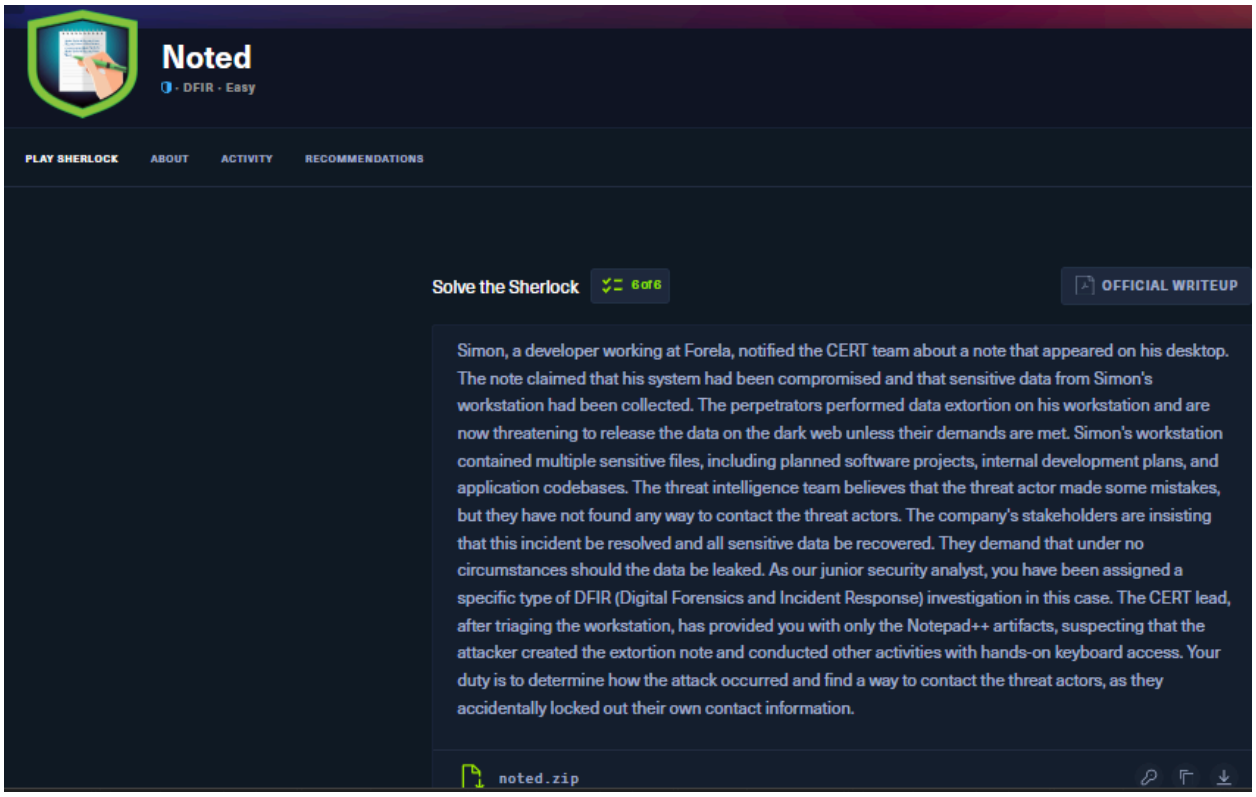
🕵️ HTB Sherlock: Noted — DFIR Write-Up



Evidence Acquisition

This Sherlock challenge was provided as a compressed archive downloaded from Hack The Box.

Downloaded Artifact

noted.zip

Extracted Contents

After extracting the ZIP file, the following artifacts were identified and used during the investigation:

config

session

backup/

YOU HAVE BEEN HACKED.txt@2023-07-24_150548

LootAndPurge.java@2023-07-24_145332

Artifact Relevance

- **config / session** → Notepad++ configuration and session metadata (critical for file paths and timestamps)
- **backup/** → Notepad++ backup directory preserving historical file states
- **YOU HAVE BEEN HACKED.txt** → Data extortion / ransom note left by the attacker
- **LootAndPurge.java** → Malicious Java source code compiled locally for data collection and staging

These artifacts collectively enabled reconstruction of the attacker's activity timeline and intent.

Objectives

- Identify legitimate development artifacts used by the victim
- Detect attacker-introduced malicious source code
- Determine what data was staged for exfiltration
- Extract accurate timestamps and attacker infrastructure details

Scenario Overview

Simon, a developer working at **Forela**, reported a ransom note that appeared on his desktop. The note claimed that his system had been compromised and that sensitive development data had been exfiltrated.

This Sherlock focuses on **developer environment abuse**, **living-off-the-land compilation**, and **forensic analysis of editor metadata** to reconstruct attacker activity.

Question 1

What is the full path of the script used by Simon for AWS operations?

Answer

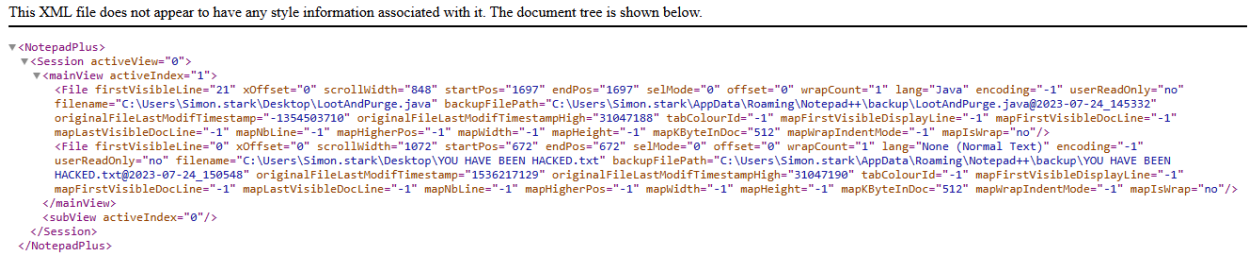
`C:\Users\Simon.stark\Documents\Dev_Ops\AWS_objects migration.pl`

Evidence

- Located in **Notepad++ configuration history**
- Repeatedly accessed by the user
- Indicates routine AWS object migration work using Perl

Question 2

What is the full path of the program's source file used by the attacker?



Answer

`C:\Users\Simon.stark\Desktop\LootAndPurge.java`

Evidence

- Identified in **Notepad++ session file**
- Java source compiled locally on the system
- Desktop location suggests attacker convenience and rapid execution

The attacker avoided dropping external tooling and instead abused the victim's existing developer environment.

Question 3

What is the name of the final archive file containing the exfiltrated data?

```
java
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.zip.ZipEntry;
import java.util.zip.ZipOutputStream;

public class Sensitive_data_extort {
    public static void main(String[] args) {
        String username = System.getProperty("user.name");
        String desktopDirectory = "C:\\Users\\" + username + "\\Desktop\\";
        List<String> extensions = Arrays.asList("zip", "docx", "ppt", "xls", "md", "txt", "pdf");
        List<File> collectedFiles = new ArrayList<>();

        collectFiles(new File(desktopDirectory), extensions, collectedFiles);

        String zipFilePath = desktopDirectory + "Forela-Dev-Data.zip";
        String password = "sdklY57BLghvyh5FJ#fion_7";

        createZipArchive(collectedFiles, zipFilePath, password);

        System.out.println("Zip archive created successfully at: " + zipFilePath);
    }

    private static void collectFiles(File directory, List<String> extensions, List<File> collectedFiles) {
        File[] files = directory.listFiles();
        if (files != null) {
            for (File file : files) {
                if (file.isDirectory()) {
                    collectFiles(file, extensions, collectedFiles);
                } else {
                    String fileExtension = getFileExtension(file.getName());
                    if (extensions.contains(fileExtension)) {
                        collectedFiles.add(file);
                    }
                }
            }
```

Answer

`Forela-Dev-Data.zip`

Evidence

- Referenced directly inside the malicious Java source code
- Used to stage sensitive development data prior to exfiltration

---

Question 4

What is the UTC timestamp when the attacker last modified the program source file?

****

Given values:
originalFileLastModifTimestamp = -1354503710
originalFileLastModifTimestampHigh = 31047188

Key Concept

These two values together represent a **Windows FILETIME timestamp**.

- FILETIME is a **64-bit value**
- Counts **100-nanosecond intervals since 1601-01-01 (UTC)**
- Stored as:
    - **Low DWORD (signed)**
    - **High DWORD (unsigned)**

In this challenge:

- LOW = `originalFileLastModifTimestamp`
- HIGH = `originalFileLastModifTimestampHigh`

Step-by-Step Solution

Step 1: Convert LOW value to unsigned 32-bit

The LOW value is signed and negative:

LOW = -1354503710

Convert to unsigned 32-bit integer:

LOW = 2^32 - 1354503710
LOW = 2940463586

Step 2: Combine HIGH and LOW into 64-bit FILETIME

Formula:

FILETIME = (HIGH << 32) + LOW

Calculation:

FILETIME = (31047188 × 2^32) + 2940463586
FILETIME = 133346660033227234

Step 3: Convert FILETIME to Unix Epoch

Formula:

UnixTime = (FILETIME / 10,000,000) − 11644473600

Calculation:

UnixTime = (133346660033227234 / 10,000,000) − 11644473600
UnixTime = 1690192403

(Fractional seconds are ignored; HTB expects seconds precision.)

Step 4: Convert Unix Epoch to UTC

1690192403

Converts to:

2023-07-24 09:53:23 UTC

****

Answer

`2023-07-24 09:53:23`

Evidence & Methodology

- Extracted from **Notepad++ session metadata**
- Timestamp stored as **Windows FILETIME (High + Low DWORD)**
- Converted using:
    - Unsigned LOW conversion
    - FILETIME → Unix Epoch
    - Unix Epoch → UTC

This timestamp represents the attacker's final modification before compilation.

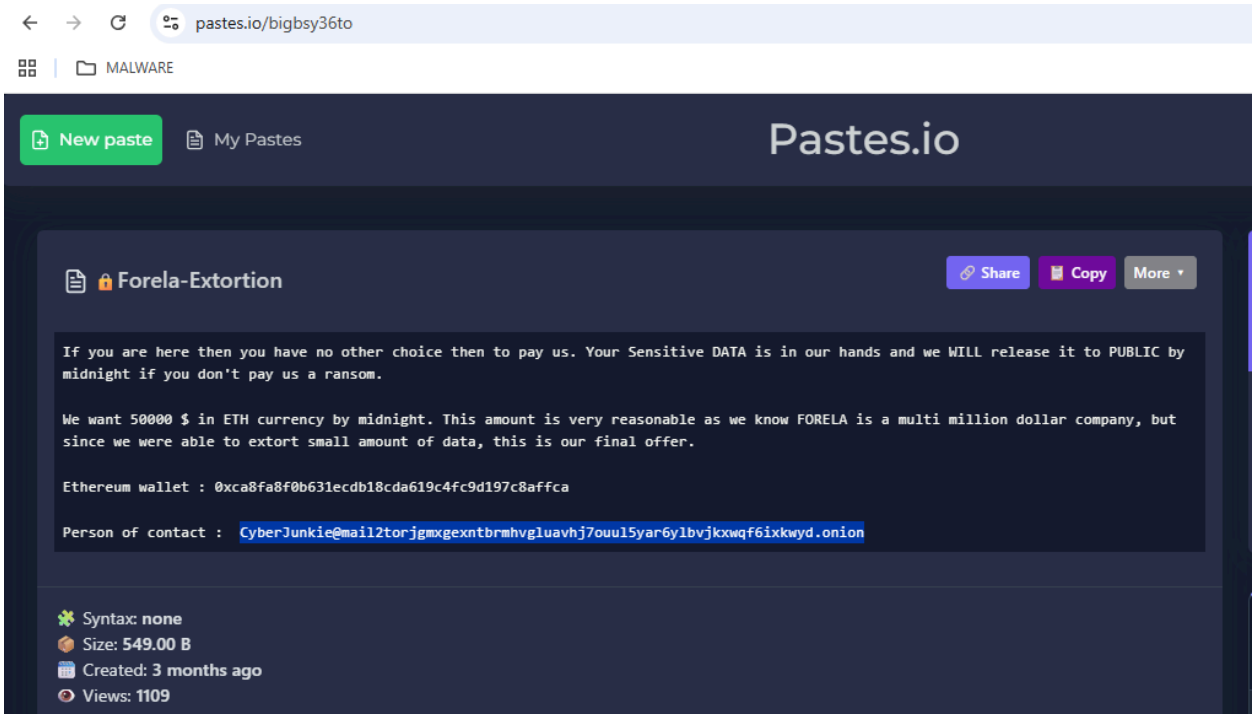
Question 5

What crypto wallet address did the attacker demand payment to?

A note was present & once you open the url present , they will ask the password that is part of java file & after that you will find the wallet & attacker mail id.

```
Hello

This note is placed in your desktop and copied to other locations too. You have been hacked and your data has been deleted from your
system. We made copies of your sensitive data and uploaded to our servers. The rule is simple

                                                YOU PAY US
                                                  AND
                        WE DO NOT RELEASE YOUR COMPANY SECRETS TO PUBLIC AND RETURN YOUR DATA SAFELY TO YOU


Failiure to oblige will result in immediate data leak to the public.



For detailed information and process , Visit any of the below links

i)  https://pastebin.ai/bigbsy36to
ii) https://pastebin.com/xmTkajd5
iii) https://pastecode.io/s/0rqtutec
```

```
java
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.zip.ZipEntry;
import java.util.zip.ZipOutputStream;

public class Sensitive_data_extort {
    public static void main(String[] args) {
        String username = System.getProperty("user.name");
        String desktopDirectory = "C:\\Users\\" + username + "\\Desktop\\";
        List<String> extensions = Arrays.asList("zip", "docx", "ppt", "xls", "md", "txt", "pdf");
        List<File> collectedFiles = new ArrayList<>();

        collectFiles(new File(desktopDirectory), extensions, collectedFiles);

        String zipFilePath = desktopDirectory + "Forela-Dev-Data.zip";
        String password = "sdklY57BLghvyh5FJ#fion_7";

        createZipArchive(collectedFiles, zipFilePath, password);

        System.out.println("Zip archive created successfully at: " + zipFilePath);
    }

    private static void collectFiles(File directory, List<String> extensions, List<File> collectedFiles) {
        File[] files = directory.listFiles();
        if (files != null) {
            for (File file : files) {
                if (file.isDirectory()) {
                    collectFiles(file, extensions, collectedFiles);
                } else {
                    String fileExtension = getFileExtension(file.getName());
                    if (extensions.contains(fileExtension)) {
                        collectedFiles.add(file);
                    }
                }
            }
```



Answer

`0xca8fa8f0b631ecdb18cda619c4fc9d197c8affca`

Evidence

- Found inside the ransom note:
  `YOU HAVE BEEN HACKED.txt`

- Address format corresponds to an **Ethereum-compatible wallet**

Question 6

What email address was provided for attacker communication?

Answer

`CyberJunkie@mail2torjgmxgexntbrmhvgluavhj7ouul5yar6ylbvjkxwqf6ixkwyd.onion`

Evidence

- Included in the extortion note
- `.onion` domain indicates **Tor-based anonymous communication**

MITRE ATT&CK Mapping

Tactic	Technique
Persistence	**T1136.001 – Create Local Account**
Defense Evasion	Living-off-the-Land Compilation
Collection	Custom Java Data Harvester
Exfiltration Prep	Local ZIP Archive Creation
Impact	Data Extortion

Key Takeaways

- Attackers abused **developer trust and tooling**
- No external malware was introduced
- **Notepad++ metadata** proved critical for timeline reconstruction
- FILETIME artifacts are a common **Sherlock precision trap**

Learning Outcome

This Sherlock highlights:

- How developer environments can be weaponized
- The forensic value of editor configuration files
- The importance of accurate timestamp interpretation in DFIR