Gazavat / Expiro DMSniff connection and DGA analysis

By Jason Reaves

Published: 2023-08-30 · Archived: 2026-04-05 12:42:30 UTC

By: Jason Reaves and Joshua Platt

Gazavat, also known at least partially as Expiro, is a multi-functional backdoor that has code overlaps with the POS malware DMSniff[1]. Functionality includes:

- Loading other executables
- Load hash cracking plugin
- Load DMSniff plugin
- Perform webinjection and webfakes
- Form grabbing
- Command execution
- Download file from infected system
- Convert infection into proxy
- DDOS
- Spreading and EXE infecting

Recovered Gazavat manual:

Press enter or click to view image in full size

Technical Overview

Gazavat, along with a few other malware variants over the years, have all been lumped together as a file infector called Expiro by AV companies. This is due to code reuse from the Carberp malware leak[2] being utilized by multiple malware families. Gazavat itself, which is believed to be what AV companies initially referred to as Expiro, is much more complicated than just a simple file infector. First, let's see the connection to DMSniff, which is how this malware ended up catching my attention: the bot id for Gazavat is passed in the user agent, which is the same method used by DMSniff.

DMSniff:

```
User-Agent: Mozilla/4.0 (compatible; MSIE 11.0; DSNF_2768=NT6.1.76016.1.7601-C386B17D.ENU.26F427F6-7
```

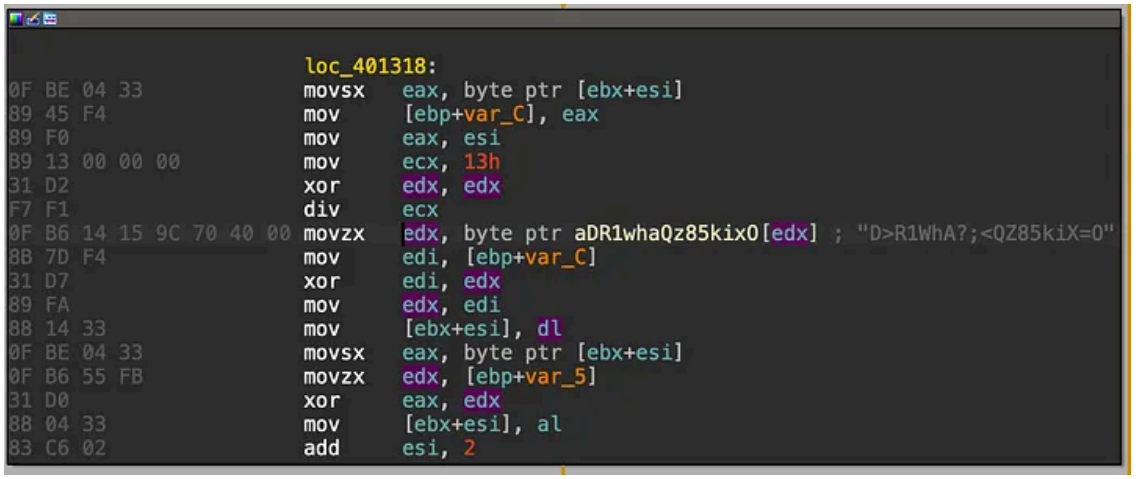
Gazavat[3]:

```
Mozilla/4.0 (compatible; MSIE 33; NT5.1.2600-74952D50.ENU.362235D7-ED9E5B-5D967F-1438147D; .NET CLR (
```

The user agents are strikingly similar in their construction — the similarities do not stop there, though. Taking a look in the binary of Gazavat shows the exact same string encoding routine as DMSniff.

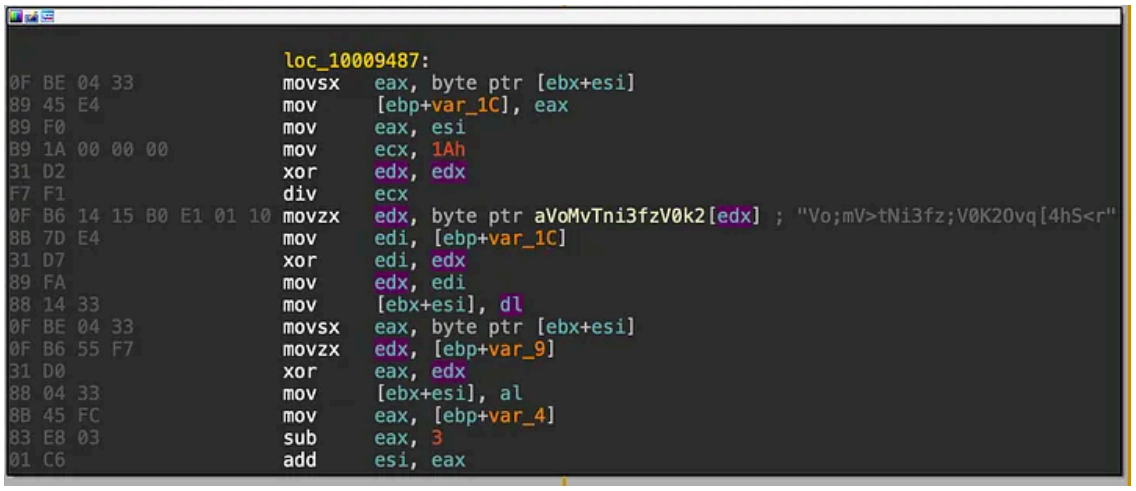
DMSniff sample(7d69e2c4e75c76c201d40dbc04b9f13b2f47bf9667ce3b937dd4b1d31b11a8af) string decryption routine:

Press enter or click to view image in full size



Gazavat sample(a3f886db3d2691794e9ec27dca65dcc5d96e6095ec1de5275967a6e6d156d1f7) string decryption routine:

Press enter or click to view image in full size



By taking the YARA rule from previous research and broadening it out a bit we are able to find hundreds of samples recently submitted to VirusTotal and begin decoding the strings. Afterwards the various types of functionality available can be discovered.

Cred Harvesting

```
00sqlite3_open|03sqlite3_free|01sqlite3_close|02sqlite3_exec|
\\mozsqlite3.dll
\\mozglue.dll
\\msvcr100.dll
\\mozcrt19.dll
FZILLA|ftp://
\\FileZilla\\sitemanager.xml
INETCOMM Server Passwords
Software\\Microsoft\\Internet Explorer\\IntelliForms\\Storage2
```

Proxy UPNP

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmls
<?xml version="1.0"?><s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle=
M-SEARCH * HTTP/1.1\r\nHOST: 239.255.255.250:1900\r\nMAN: "ssdp:discover"\r\nMX: 1\r\nST: urn:schema
POST %s HTTP/1.1\r\nHOST: %s:%u\r\nCONTENT-LENGTH: %u\r\nCONTENT-TYPE: text/xml; charset="utf-8"\r\n
<NewRemoteHost></NewRemoteHost><NewExternalPort>%i</NewExternalPort><NewProtocol>%s</NewProtocol><Ne
schemas-upnp-org:control-1-0
```

AV Scanning

```
|wscsvc|WinDefend|MsMpSvc|NisSrv|
\\Microsoft Security Client\\
\\Windows Defender\\
```

Dropping Browser Extensions

```
software\\Mozilla\\Mozilla FireFox
\r\nExtension%u=%s\\%s
[ExtensionDirs]
.ini
chrome\\content.jar
components
chrome
{ec9032c7-c20a-464f-7b0e-13a3a9e97385}
install.rdf
chrome.manifest
chrome\\content
components\\red.js
%s\\%s\\extensions
%s\\*
%s\\Mozilla\\Firefox\\Profiles
content.js
manifest.json
background.js
##HOST_ID##
##DOMAIN##
##XERSION##
Extensions\\
dlddmedljhmbgdhapibnagaanenmajcm
"f (document.location.href==\'chrome://tasks/\') parent.self.location=\'chrome://sessions/\';\n
if (document.location.href==\'chrome://extensions-frame/\') { window.setInterval(CheckExt,10); };\n
function CheckExt(){ var el=document.getElementById(\'%s\'); if (el != null) el.parentNode.removeChi

<?xml version="1.0"?>\n\n<RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"\n     xmlns:em="ht
>90.*</em:maxVersion>\n      </Description>\n    </em:targetApplication>\n\n    <em:name>.</em:name>
```

```
content\tsample\tjar:chrome/content.jar!/content/\n#content\tsample\tchrome/content/\n\noverlay chro
```

DGA related

```
%s%c%c%c%c-%c%c%c%c%c.com
%s%c%c%c%c%c-%c%c%c%c.ru
```

Webinjects and Webfakes (via browser extension)

```
beforeEnd
return Ci.nsIContentPolicy.ACCEPT},shouldProcess:function(c,e,a,d,b,f){return Ci.nsIContentPolicy.AC
RL!=l.toreplace){delete d.redirecting}else{return Ci.nsIContentPolicy.ACCEPT}}}try{var n=j.spec.repl
ategoryManager);a.addCategoryEntry("content-policy",this.classDescription,this.contractID,true,true)
:"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789(/)",encode:function(c){var a="";var
const Ci=Components.interfaces;const Cc=Components.classes;const Cr=Components.results;const Cu=Comp
fexport[a]}catch(b){}},FindRedirectSign:function(b,a){if(!this.RedirectList){return false}for(var c=
<2048)){a+=String.fromCharCode((d>>6)|192);a+=String.fromCharCode((d&63)|128)}else{a+=String.fromCha

     },\r\n            "incognito": true,\r\n            "install_time": "12991426726872000",\r\n
// Copyright (c) 2011 The Chromium Authors. All rights reserved.\r\n// Use of this source code is go
rl=="chrome://cache/"){chrome.tabs.update(a,{url:"chrome://predictors/"})}if(b.url=="chrome://net-in
 "manifest_version": 2,\r\n              "name": "Google Chrome",\r\n              "permissions":
ion|subm|submit|||substring|userAgent|navigator|this|form|value|blank|true|addEventListener|greeting
(b.4=="S"){5}d+=a+":"+b[a].4+":"+((b[a].6=="")?"<z>:":b[a].6)+":";3((b[a].4=="R")||(b[a].4=="Q")){d+
\r\n         "dlddmedljhmbgdhapibnagaanenmajcm": {\r\n            "active_permissions": {\r\n

tener("error",function(f){onErrorAbortImage(f)},true);a.addEventListener("abort",function(f){onError
ld(a);delete BUF[b]}function onErrorAbortImage(b){var a=b.target;a.parentNode.removeChild(a)}functio
ction Base64_encode(d){var c="ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789(/)";var
{\r\n  "name": "Google Chrome",\r\n  "version": "1.0",\r\n  "background": { "scripts": ["background.
[5]!="a")){return}var d=/;(\\S*)/.exec(b);ParseInjects(d[1]);chrome.storage.local.set({INJ_BLOCK:d[1
a<MAX;a++){if(BUF[a]){continue}BUF[a]=b;InsertImg(document,a);return}}function BefSendHead(e){if(e.t
// Copyright (c) 2011 The Chromium Authors. All rights reserved.\r\n// Use of this source code is go
e);d++}else{if((e>191)&&(e<224)){c2=a.charCodeAt(d+1);b+=String.fromCharCode(((e&31)<<6)|(c2&63));d+
rn}var b=0;if(c.status==200){var a=c.responseText;if((a[42]==";")&&(a[0]=="G")&&(a[1]=="I")&&(a[2]==
("INJ_BLOCK",get_inj);chrome.extension.onMessage.addListener(onMsg);chrome.webNavigation.onCompleted
Code(l);if(f!=64){a=a+String.fromCharCode(j)}if(e!=64){a=a+String.fromCharCode(g)}}a=Base64v2_utf8_d
```

Card related

```
VISA
MasterCard
Unable to authorize.\n%s processing center is unable to authorize your card %s.\nMake corrections an
Unable to authorize
```

```
 )=====\r\n\r\n
\r\n\r\n=====[
```

The samples also come with an encoded list of C2 domains. In the event that the binary is built to perform webinjection attacks, a second list will also be on board that will be utilized by the browser extension as mentioned in previous work[3].

Get Jason Reaves's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

Decoded C2 list example:

```
vietwarok[.]in#fari-khan[.]in#oldlexus-sales[.]in#viewofpakiwar[.]in#prom-zonaars[.]ru#avchecpk[.]ru
```

Decoded list for the browser extension traffic:

```
systemtime[.]ru#systemsync[.]ru#altruist[.]pro#uni-link[.]in#fedlaw-gosdep[.]ru#save-galapagos-turtl
```

DGA

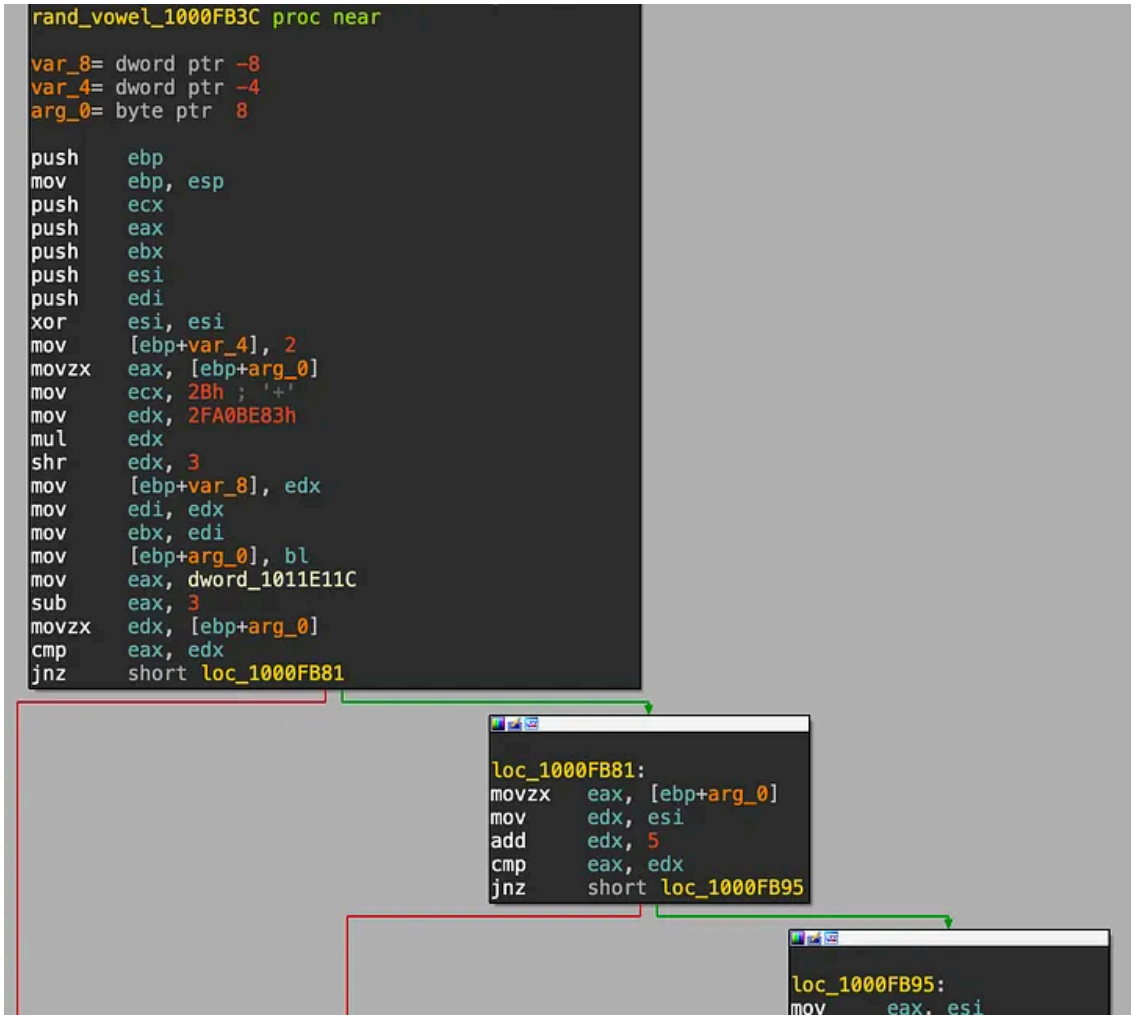
As was previously mentioned some strings are related to a DGA, the DGA is different from DMSniff — but it's also much more obfuscated in the binaries we reverse-engineered. It's worth mentioning that this malware has been around for a long time, so there are likely more variants than what have been found so far. The biggest difference in the algorithm is the usage of a hardcoded string which, in the samples analyzed, was all a single character and the structure of the domain in the strings:

```
%s%c%c%c%c-%c%c%c%c%c.com
%s%c%c%c%c%c-%c%c%c%c.ru
```

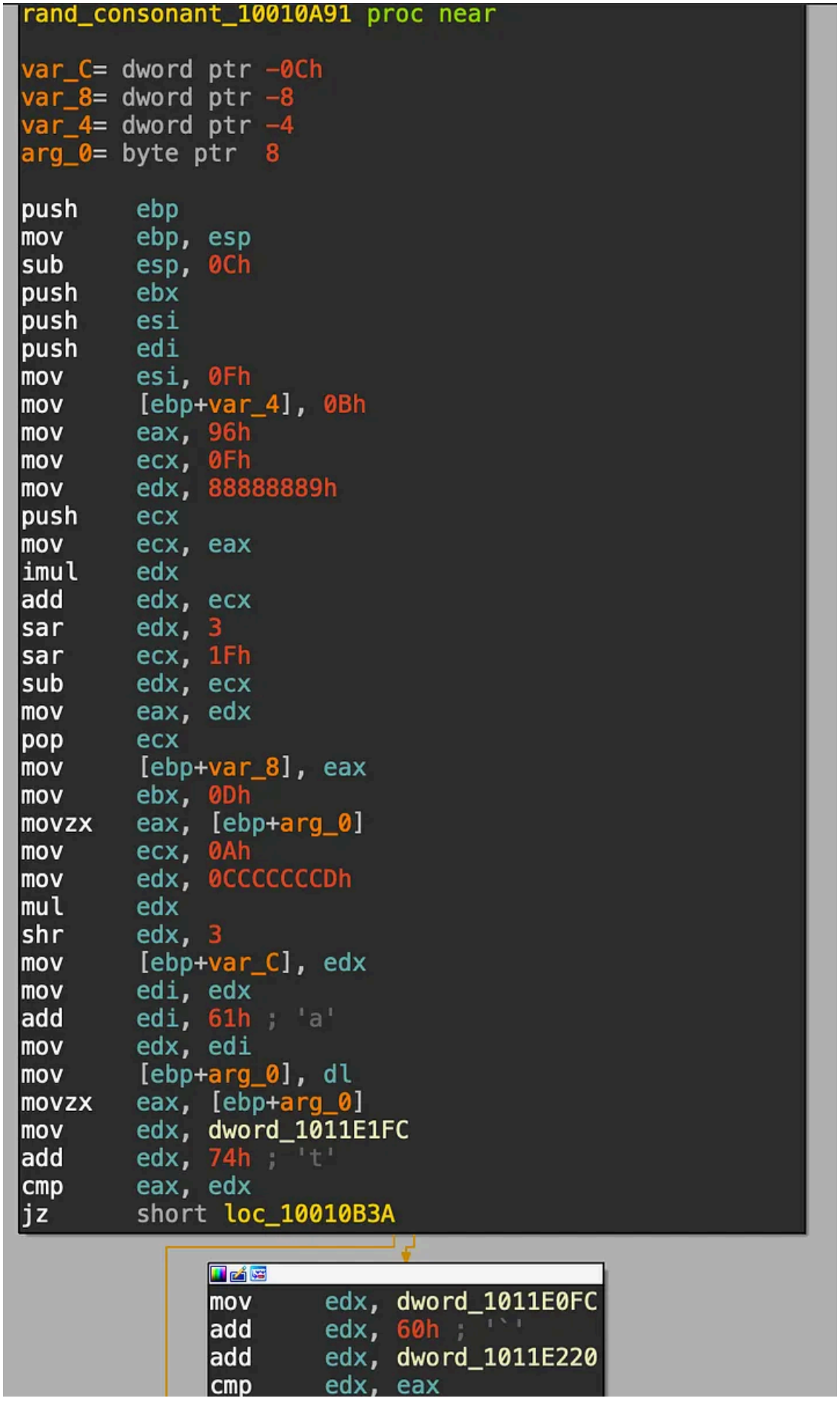
The algorithm creates 9 char values which will be a mix of vowels and consonants, along with the hardcoded piece that was all a single char value stored as a C-style string.

There are two functions used to pick the chars, which have remained static in the samples found — one for picking a vowel and one for picking a consonant:

Press enter or click to view image in full size



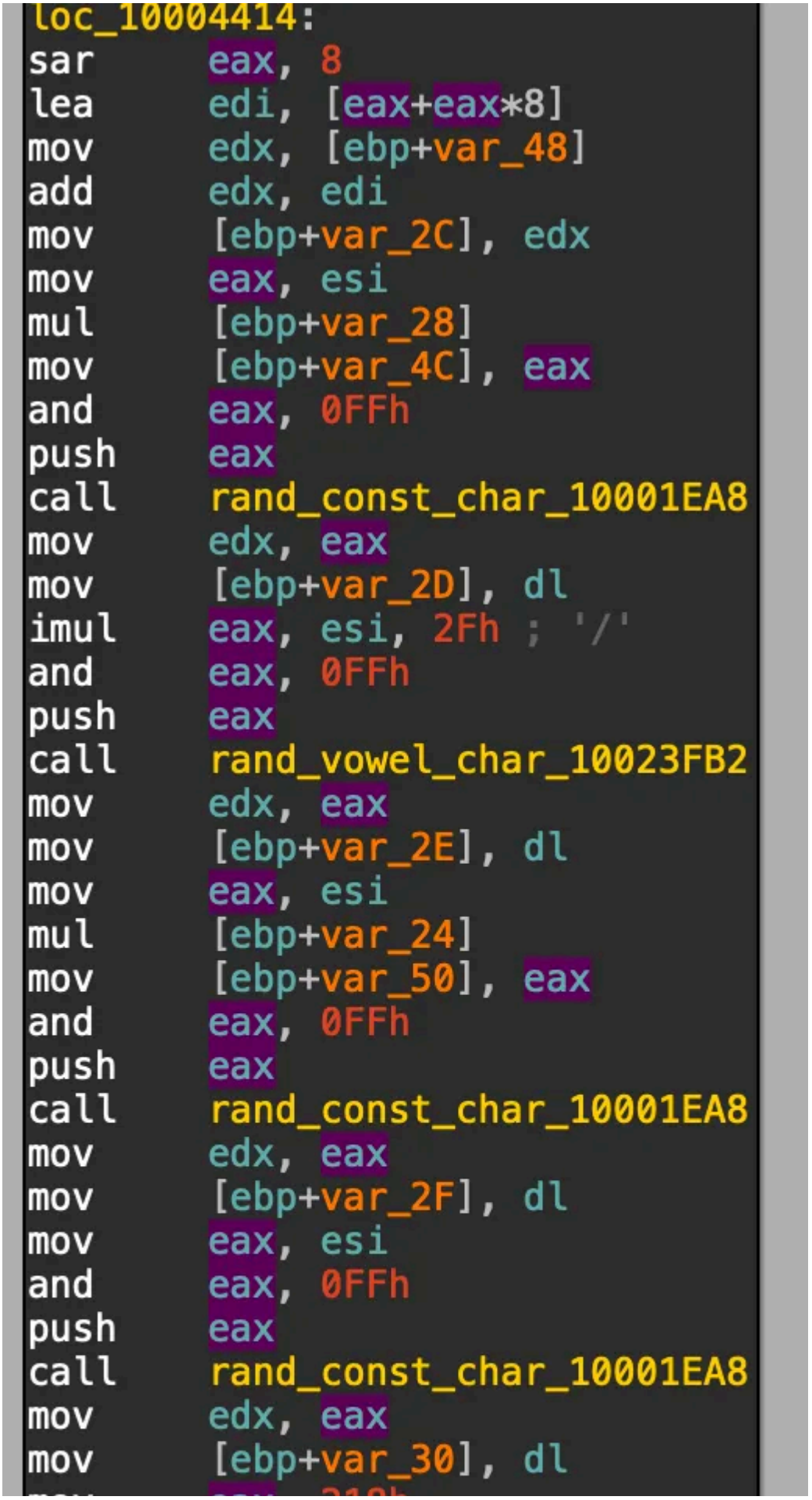
Press enter or click to view image in full size

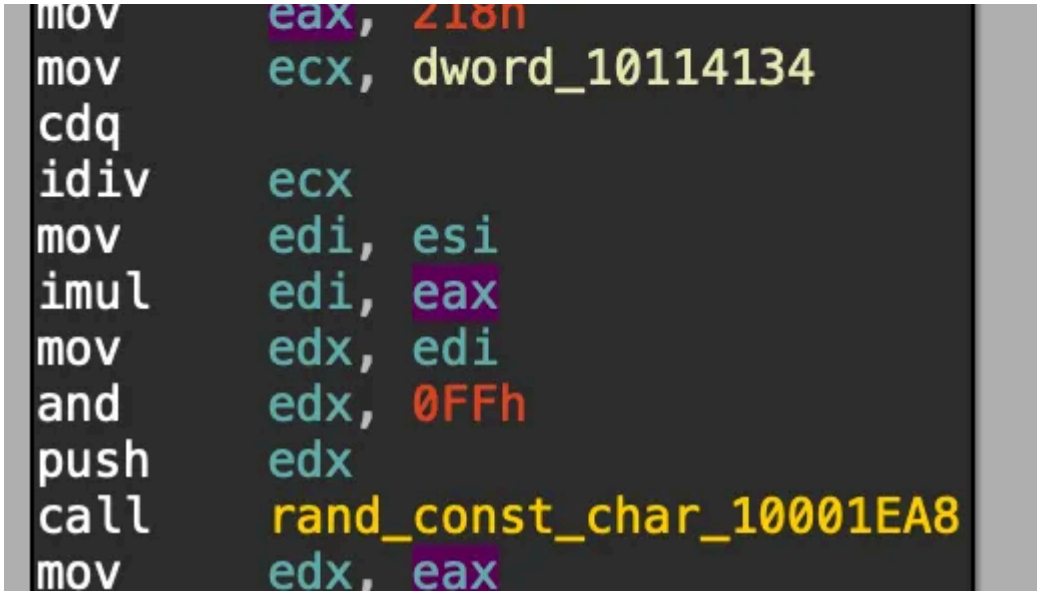

rand_consonant_10010A91 proc near
var_C= dword ptr -0Ch
var_8= dword ptr -8
var_4= dword ptr -4
arg_0= byte ptr 8
push ebp
mov ebp, esp
sub esp, 0Ch
push ebx
push esi
push edi
mov esi, 0Fh
mov [ebp+var_4], 0Bh
mov eax, 96h
mov ecx, 0Fh
mov edx, 88888889h
push ecx
mov ecx, eax
imul edx
add edx, ecx
sar edx, 3
sar ecx, 1Fh
sub edx, ecx
mov eax, edx
pop ecx
mov [ebp+var_8], eax
mov ebx, 0Dh
movzx eax, [ebp+arg_0]
mov ecx, 0Ah
mov edx, 0CCCCCCCDh
mul edx
shr edx, 3
mov [ebp+var_C], edx
mov edi, edx
add edi, 61h ; 'a'
mov edx, edi
mov [ebp+arg_0], dl
movzx eax, [ebp+arg_0]
mov edx, dword_1011E1FC
add edx, 74h ; 't'
cmp eax, edx
jz short loc_10010B3A
mov edx, dword_1011E0FC
add edx, 60h ; '`'
add edx, dword_1011E220
cmp edx, eax

Python versions:

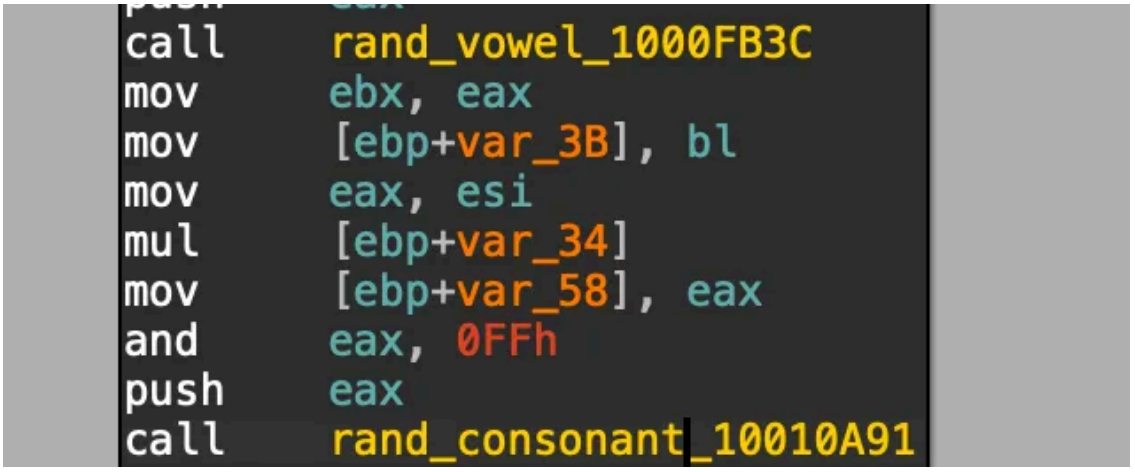
```
def rand_vowel_char(a):
 val = int((a & 0xff)/0x2b) & 0xff
 if val == 4:
  return(chr(111))
 if val == 5:
  return(chr(97))
 if val == 3:
  return(chr(105))
 if val == 2:
  return(chr(117))
 if val == 0:
  return(chr(101))
 result = chr(val)
 if val == 1:
  return(chr(121))
 return(result)

def rand_const_char(a):
 val = int((a & 0xff) / 0xa + 97) & 0xff
 if val in [121,111,101,117,97, 106, 105]:
  val += 1
 return chr(val)
```

Below are two examples of the DGA algorithm where it can see the general flow remains the same, but there are some obfuscation additions that change between samples:


loc_10004414:
sar eax, 8
lea edi, [eax+eax*8]
mov edx, [ebp+var_48]
add edx, edi
mov [ebp+var_2C], edx
mov eax, esi
mul [ebp+var_28]
mov [ebp+var_4C], eax
and eax, 0FFh
push eax
call rand_const_char_10001EA8
mov edx, eax
mov [ebp+var_2D], dl
imul eax, esi, 2Fh ; '/'
and eax, 0FFh
push eax
call rand_vowel_char_10023FB2
mov edx, eax
mov [ebp+var_2E], dl
mov eax, esi
mul [ebp+var_24]
mov [ebp+var_50], eax
and eax, 0FFh
push eax
call rand_const_char_10001EA8
mov edx, eax
mov [ebp+var_2F], dl
mov eax, esi
and eax, 0FFh
push eax
call rand_const_char_10001EA8
mov edx, eax
mov [ebp+var_30], dl


mov eax, 218h
mov ecx, dword_10114134
cdq
idiv ecx
mov edi, esi
imul edi, eax
mov edx, edi
and edx, 0FFh
push edx
call rand_const_char_10001EA8
mov edx, eax


loc_1001363D:
sar eax, 8
imul edi, eax, 11h
imul edx, dword_1011E1B0, 7
add edi, edx
mov [ebp+var_38], edi
imul eax, esi, 2Fh ; '/'
and eax, 0FFh
push eax
call rand_vow
pop ecx
mov edx, eax
mov [ebp+var
mov eax, 60h
mov ecx, 8
test eax, eax
jge short lo
...
var_3F db ?
var_3E db ?
var_3D db ?
var_3C db ?
var_3B db ?
var_3A db ?
var_39 db ?
var_38 dd ?
...
x5e
add eax, 7
loc_10013674:
sar eax, 3
mul esi
mov [ebp+var_54], eax
mov edx, eax
and edx, 0FFh
push edx
call rand_vowel_1000FB3C
mov edx, eax
mov [ebp+var_3A], dl
imul eax, esi, 49h ; 'I'
and eax, 0FFh
push eax



After getting through the obfuscation, a recreation of the algorithm in python can be seen below:

```
def dga(a):
 hc_char = chr(102)
 t1 = 9 * int((a+1)/256) + 31
 t2 = 7 * 3 + 17 * int((a + 1)/256)
 t3 = 11 * 3 + 23 * int((a+1) / 0x10000)
 v1 = rand_const_char(t1 * (a+1))
 v2 = rand_vowel_char(12 * (a+1))
 v3 = rand_const_char(t2 * (a+1))
 v4 = rand_vowel_char(113 * (a+1))
 v5 = rand_const_char(a+1)
 v6 = rand_vowel_char(47 * (a+1))
 v7 = rand_const_char(t3 * (a+1))
 v8 = rand_vowel_char(73 * (a+1))
 v9 = rand_const_char(67 * (a+1))
 tld = ''
 if(2 * (ord(v17) >> 1)) == ord(v17):
  tld = '.ru'
  dom = hc_char+v1+v2+v3+v4+v5+'-'+v6+v7+v8+v9+tld
 else:
  tld = '.com'
  dom = hc_char+v1+v2+v3+v4+'-'+v5+v6+v7+v8+v9+tld
 return(dom)
```

YARA

```
rule gaza_dga
{
strings:
$a1 = {0f b6 ?? ?? b9 2b 00 00 00 ba 83 be a0 2f f7 e2 c1 ?? 03}
$const1 = {0f b6 ?? ?? b9 0a 00 00 00 ba cd cc cc cc f7 e2 c1 ?? 03}
condition:
all of them
```

```
}

rule gazavat_broad_hunting
{
strings:
$a1 = {b9 ?? 00 00 00 [1-2] f7 ?? 0f b6}
$a2 = {31 d? 89 ?? 8? [1-8] 0f b? ?? ?? 0f b? ?? ?? 3? d?}
condition:
all of them
}
```

Samples

DMSniff

```
7d69e2c4e75c76c201d40dbc04b9f13b2f47bf9667ce3b937dd4b1d31b11a8af
```

Gazavat

```
a3f886db3d2691794e9ec27dca65dcc5d96e6095ec1de5275967a6e6d156d1f7
08c656125a3c1abdb74ede3712aecca1a5e4a48984cae78aa60cb833f7231295
050ad1608dca7a938203d185ecfb1ecc69b3e8501129327264d5bc4b67eacff3
0a6cb9adcc23cb33b87689d2c328b742952e8e50c547380cfefe087c055af652
0df600d642caf0969134605d010942b4394843054f37c063621c60508a93c9c0
10f47f9f2fc3d3e46fbbaed21a6298b0d3882faa6a2de5fbb99094c6513ec392
17e81becddbac84bb2fa9412ae11d6b066945ca85ef1a77c51e688bcf42f59b8
23ef8ce504d5430d543509adeccb3f218aed56f5444aeecdc7d115b96e4b2373
36ab0415049c5d8c9eb5721ad0c1d941976ff905a824609007e6c4c086e9aa6e
38223ad3f30e8cdb0602e3f818e80ca936e15e4bd3bc427aff2ab1f91bb2fe46
6da2602e7a95012a258b205c7f44bee5a964a938876509a09b7b01ce92fad764
81a977f7b415480f01a2d44340be4cc35fe8868e7fa699a305b2dcc312c33dd8
84636c0500d344a0c40252381521bf8adf9e2829326aec06892a36f10079a6f5
85cb5eadfbc883448bfde48713f9b1dea9e731b6537361ea1f3d807123af982b
85cc3d2f8b6a40ccdf446ad77fcf3681403ac5dc633b1baa1297581118f5160d
d0405857330b188e808002c6ba457a858ab1a6d6bdef71831be4195db04d5c1d
d200e0227bbea44646dcc41bcbb7d3bba5e7fa9cdc63dbeaaa99389d3e54c945

1d7c5347aa687da9da8c329811392faafd76aa3ddbd77b7774470d7f8ba094d9
0d7aba6c6c88372928daf3b43323a70324515d1791785d10e1798e105185144c
```

Mutex

```
gazavat-svc
kkq-vx
```

Browser Extensions

```
mdgkfajodaliacghnafobjnclblcfmlm
dlddmedljhmbgdhapibnagaanenmajcm
\1.0_0\background.js
\1.0_0\content.js
\1.0_0\manifest.json
{ec9032c7-c20a-464f-7b0e-13a3a9e97385}\chrome.manifest
{ec9032c7-c20a-464f-7b0e-13a3a9e97385}\chrome\content.jar
{ec9032c7-c20a-464f-7b0e-13a3a9e97385}\components\red.js
{ec9032c7-c20a-464f-7b0e-13a3a9e97385}\install.rdf
```

References

1: https://flashpoint.io/blog/dmsniff-pos-malware-actively-leveraged-target-medium-sized-businesses/

2: https://github.com/m0n0ph1/malware-1/blob/master/Carberp%20Botnet/source%20-%20absource/pro/all%20source/Worm/Black_JW/kkqvx.h

3: https://stopmalvertising.com/malware-reports/abuse-teams-targeted-by-expiro-analysis.html

4: https://malware447.rssing.com/chan-6195220/all_p2.html

Source: https://medium.com/walmartglobaltech/gazavat-expiro-dmsniff-connection-and-dga-analysis-8b965cc0221d