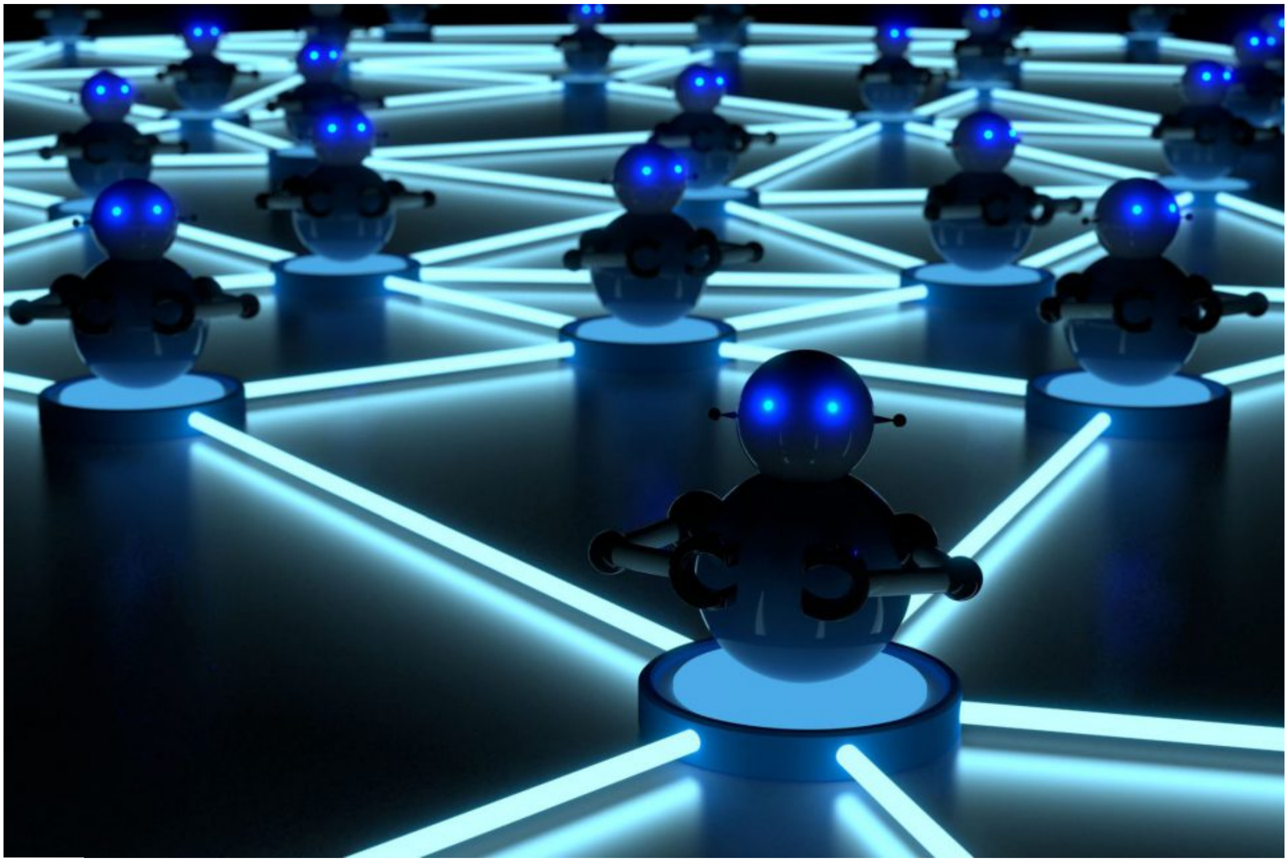
New Mozi malware family quietly amasses IoT bots

blog.centurylink.com/new-mozi-malware-family-quietly-amasses-iot-bots/

April 13, 2020



Black Lotus Labs Posted On April 13, 2020

0

Summary

The explosion of Internet of Things (IoT) devices has long served as a breeding ground for malware distribution. The inability for users to patch many IoT devices has only compounded this problem, as bad actors continue to evolve tactics to leverage botnets for DDoS attacks and other malicious behavior. Black Lotus Labs tracks malware families that present new or distinct threats to the global community, and recently began tracking a new malware family called Mozi.

Mozi is evolved from the source code of several known malware families – Gafgyt, Mirai and IoT Reaper – that have been brought together to form a peer-to-peer (P2P) botnet capable of DDoS attacks, data exfiltration and command or payload execution. The malware targets IoT devices, predominantly routers and DVRs that are either unpatched or have weak telnet passwords. After a notable traffic increase in December was mistakenly attributed to other malware families by researchers, Black Lotus Labs reviewed entries in our reputation system for that timeframe, which revealed a different story. This traffic was not simply increased activity by a known family, but a new family altogether.

Black Lotus Labs Findings

In December 2019, Black Lotus Labs observed an increase of entries of compromised hosts in our reputation system labeled as IoT Reaper. Since this malware family has not changed in some time, the increase was unexpected, and led to further investigation of the increase. Upon review of these entries we began to see a pattern develop, each host had an http server listening on a random port that served a file which included “Mozi” in the name. File names such as “Mozi.m” and “Mozi.a” were seen throughout all of the identified hosts.

Research revealed that these hosts were part of a growing P2P botnet and were making the Mozi files available for distribution to newly infected hosts. While the increase in data began in December, our data shows that the use of the Mozi filename began earlier, in September. Further analysis revealed that the malware had been identified under many of its contributing families, including Mirai, Gafgyt, and IoT Reaper. The original spike consisted of 69 hosts, but a broader look revealed 4,216 hosts. Shortly after we began our research based on samples being detected as multiple families, Netlab360 shared their findings on this malware[1] confirming our understanding.

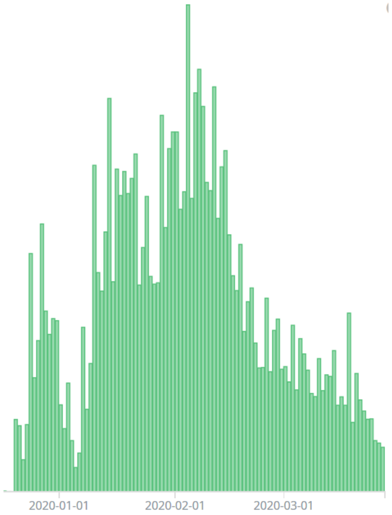
The Mozi botnet is comprised of nodes that utilize a distributed hash table (DHT) for communication, similar to the code used by IoT Reaper and Hajime. These nodes also host the Mozi.m and Mozi.a malware binary files, passed during the compromise of new hosts, on a randomly chosen port. The standard DHT protocol is commonly used to store node contact information for torrent and other P2P clients. Using DHT allows the malware to bypass the use of standard malware command and control servers while hiding behind the large amount of typical DHT traffic. This makes it harder to track and impact the control infrastructure. As a P2P botnet, Mozi implements its own custom extended DHT described later.

To enumerate the botnet, Black Lotus Labs implemented a machine learning model trained on the observed unique DHT traffic implementation utilized by Mozi. This allows us to distinguish between Mozi nodes and benign hosts, and identify the IPs suspected of participating in the botnet. When we identify a new suspected Mozi node, our software attempts to confirm the suspicion by sending messages proprietary to the malware's p2p protocol, and looks for correctly formatted responses. When the correct response is seen, the host is validated as a member of the Mozi botnet.

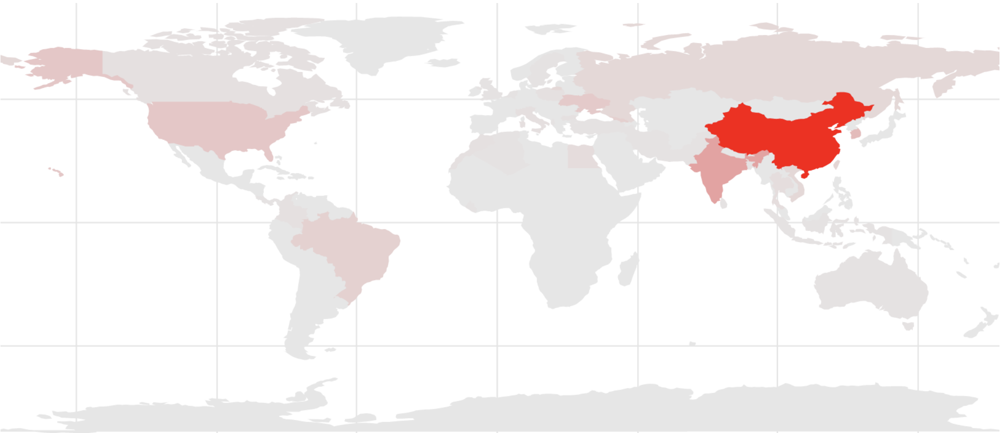
The Mozi botnet is growing as the malware evolves

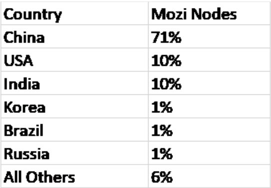
To get a view on the potential size of this new botnet we continue to review the data within our reputation system, and the IPs of the hosts confirmed as Mozi botnet nodes.

Looking at unique Mozi nodes hosting the malware during the initial spike witnessed, there were 323 unique nodes, by Dec. 27 unique nodes had grown to 1,205, then hitting a high point on Feb. 4 with 2,191 nodes. Our review showed a steady increase in the number of compromised hosts over a span of three months – then a decline beginning in late February. Overall, we have observed over 15,858 unique Mozi nodes over the last four months.



The geographic distribution of Mozi botnet nodes since September, 2019:





Country	Mozi Nodes
China	71%
USA	10%
India	10%
Korea	1%
Brazil	1%
Russia	1%
All Others	6%

Throughout the life of the Mozi botnet, the bulk of the nodes have been located in Asia, with China hosting 71% of infected hosts. Outside of Asia, the largest footprint has been the US, with 10% of all nodes. A review of the data for just the month of March shows China increasing to 81% of the botnet nodes and the United Stated dropping to less than 1%.

Our first observations of the malware in December found only three distinct samples. As time as has passed, we've now seen 34 unique samples, suggesting the operators are continuing to manipulate the malware.

A deeper look into the Mozi malware

The Mozi samples analyzed are ELF binaries with versions targeted for MIPS and ARM processor architectures. Once executed, the binary forks many versions of itself renamed as 'ssh' or 'dropbear'. The forked processes are responsible for setting up the DHT communications and closing ports to prevent infection by other malware. The forked processes can also set up HTTP on a randomly chosen port to host the Mozi binary.

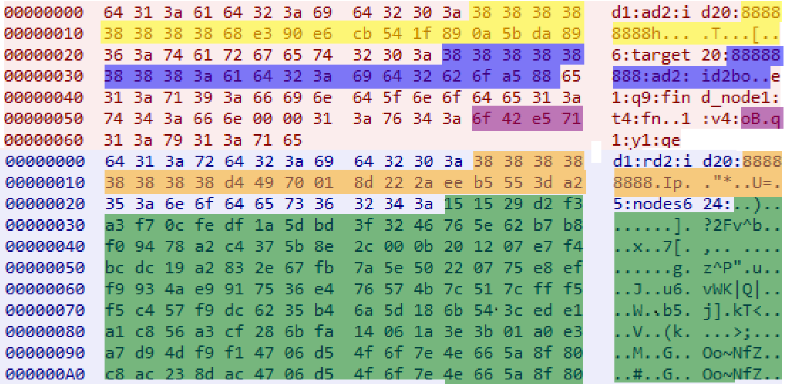
Mozi uses a modified DHT protocol for communication. The bot initially DHT pings several nodes hardcoded in the binary to bootstrap the initial connection to the DHT network (listed below). For the nodes that respond, the bot then sends a DHT 'find_node' command to locate other bots on the Mozi botnet. It often takes several find_node attempts to locate an active Mozi peer.

Example find_node request:

```
d1:ad2:id20:88888888h\xe3\x90\xe6\xcbT\x1f\x89\x0a[\xda\x896:target20:88888888:ad2:id2bo\xa5\x88e1:q9:find_node1:t4:fn\x00\x00\1:v4oB\xe5
```

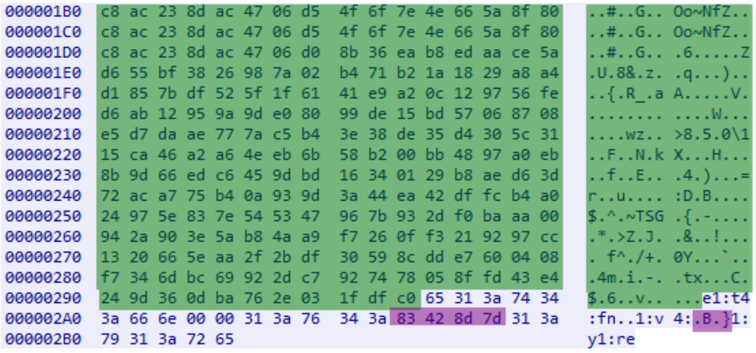
If this request is sent to an active Mozi peer, the peer will often return the config of the contacted peer as the value to the 'nodes' key, alternatively it will return other nodes in the 'nodes' key. The config is XOR encrypted with a hardcoded key.

The encoded 'find_node' request looks like:



```
00000000  64 31 3a 61 64 32 3a 69  64 32 30 3a 38 38 38 38   d1:ad2:i d20:8888
00000010  38 38 38 38 68 e3 90 e6  cb 54 1f 89 0a 5b da 89   8888h... .T...[..
00000020  36 3a 74 61 72 67 65 74  32 30 3a 38 38 38 38 38   6:target 20:88888
00000030  38 38 38 3a 61 64 32 3a  69 64 32 62 6f a5 88 65   888:ad2: id2bo..e
00000040  31 3a 71 39 3a 66 69 6e  64 5f 6e 6f 64 65 31 3a   1:q9:fin d_node1:
00000050  74 34 3a 66 6e 00 00 31  3a 76 34 3a 6f 42 e5 71   t4:fn..1 :v4:oB.q
00000060  31 3a 79 31 3a 71 65                               1:y1:qe
00000000  64 31 3a 72 64 32 3a 69  64 32 30 3a 38 38 38 38   d1:rd2:i d20:8888
00000010  38 38 38 38 d4 49 70 01  8d 22 2a ee b5 55 3d a2   8888.Ip. ."*..U=.
00000020  35 3a 6e 6f 64 65 73 36  32 34 3a 15 15 29 d2 f3   5:nodes6 24:..)..
00000030  a3 f7 0c fe df 1a 5d bd  3f 32 46 76 5e 62 b7 b8   ......]. ?2Fv^b..
00000040  f0 94 78 a2 c4 37 5b 8e  2c 00 0b 20 12 07 e7 f4   ..x..7[. ,.. ....
00000050  bc dc 19 a2 83 2e 67 fb  7a 5e 50 22 07 75 e8 ef   ......g. z^P".u..
00000060  f9 93 4a e9 91 75 36 e4  76 57 4b 7c 51 7c ff f5   ..J..u6. vWK|Q|..
00000070  f5 c4 57 f9 dc 62 35 b4  6a 5d 18 6b 54 3c ed e1   ..W..b5. j].kT<..
00000080  a1 c8 56 a3 cf 28 6b fa  14 06 1a 3e 3b 01 a0 e3   ..V..(k. ...>;...
00000090  a7 d9 4d f9 f1 47 06 d5  4f 6f 7e 4e 66 5a 8f 80   ..M..G.. Oo~NfZ..
000000A0  c8 ac 23 8d ac 47 06 d5  4f 6f 7e 4e 66 5a 8f 80   ..#..G.. Oo~NfZ..
```

The bytes removed are repeating bytes of the XOR key and will be 0x00 bytes after decryption



```
000001B0  c8 ac 23 8d ac 47 06 d5  4f 6f 7e 4e 66 5a 8f 80   ..#..G.. Oo~NfZ..
000001C0  c8 ac 23 8d ac 47 06 d5  4f 6f 7e 4e 66 5a 8f 80   ..#..G.. Oo~NfZ..
000001D0  c8 ac 23 8d ac 47 06 d0  8b 36 ea b8 ed aa ce 5a   ..#..G.. .6.....Z
000001E0  d6 55 bf 38 26 98 7a 02  b4 71 b2 1a 18 29 a8 a4   .U.8&.z. .q...)..
000001F0  d1 85 7b df 52 5f 1f 61  41 e9 a2 0c 12 97 56 fe   ..{.R_.a A.....V.
00000200  d6 ab 12 95 9a 9d e0 80  99 de 15 bd 57 06 87 08   ........ ....W...
00000210  e5 d7 da ae 77 7a c5 b4  3e 38 de 35 d4 30 5c 31   ....wz.. >8.5.0\1
00000220  15 ca 46 a2 a6 4e eb 6b  58 b2 00 bb 48 97 a0 eb   ..F..N.k X...H...
00000230  8b 9d 66 ed c6 45 9d bd  16 34 01 29 b8 ae d6 3d   ..f..E.. .4.)...=
00000240  72 ac a7 75 b4 0a 93 9d  3a 44 ea 42 df fc b4 a0   r..u.... :D.B....
00000250  24 97 5e 83 7e 54 53 47  96 7b 93 2d f0 ba aa 00   $.^.~TSG .{.-....
00000260  94 2a 90 3e 5a b8 4a a9  f7 26 0f f3 21 92 97 cc   .*.>Z.J. .&..!...
00000270  13 20 66 5e aa 2f 2b df  30 59 8c dd e7 60 04 08   . f^./+. 0Y...`..
00000280  f7 34 6d bc 69 92 2d c7  92 74 78 05 8f fd 43 e4   .4m.i.-. .tx...C.
00000290  24 9d 36 0d ba 76 2e 03  1f df c0 65 31 3a 74 34   $.6..v.. ...e1:t4
000002A0  3a 66 6e 00 00 31 3a 76  34 3a 83 42 8d 7d 31 3a   :fn..1:v 4:.B.}1:
000002B0  79 31 3a 72 65                                     y1:re
```

Yellow – Bot ID

Blue – Target ID

Purple – Version flag

Orange – Responding nodes ID

Green – Encrypted config

Bytes without color coding – Bencode encoded bytes not specific to Mozi

Eventually Mozi peers will return their XOR encoded config. Once decrypted the config contains (as described by Netlab360) a bot role and a DHT prefix. The bot roles that we have seen in our data are 'bot', 'sk', 'ftp', and 'sns'. The DHT prefix seems to always be '88888888', except in the case of the 'sns' bot role in which it is just '888'.

Decoded config samples:

[ss]**sns**[/ss][cpu].m[/cpu][hp]888[/hp]

[ss]**ftp**[/ss][cpu].w[/cpu][hp]88888888[/hp]

[ss]**sk**[/ss][hp]88888888[/hp][count]http://ia.51[.]la/go1?id=20198527&pu=http%3a%2f%2fbaidu[.]com/[idp][/count]

[ss]**bot**[/ss][hp]88888888[/hp][count]http://ia.51[.]la/go1?id=19894027&pu=http%3a%2f%2fbaidu[.]com/[idp][/count]

Mitigation Recommendations

Mozi is a fitting example of a new botnet ramping up virtually unnoticed or misattributed. What was initially viewed as small spike of a single family was actually a single, much larger, growing botnet. As evidenced by the rapid increase in size of this botnet, there are still large numbers of vulnerable IoT devices on the Internet. Black Lotus Labs continues to monitor the Mozi botnet and will continue to work to enumerate the P2P nodes. Black Lotus Labs has reached out to the most active nodes' upstream providers in an attempt to disrupt and slow the botnet's growth.

All the vulnerabilities targeted by the botnet are well known and are prevented by either proper patching or proper password management. Companies and individuals can follow these security best practices to help prevent these types of compromises in the future:

- Implement effective passwords on all IoT devices when possible.
- Restrict IoT devices access to the wide-open internet.
- Implement proper patching methodologies.
- Review this list of affected devices and their vulnerabilities.

Netlab360 shared a table of affected devices and their vulnerabilities:

Affected Device	Vulnerability
Eir D1000 Router	Eir D1000 Wireless Router RCI
Vacron NVR devices	Vacron NVR RCE
Devices using the Realtek SDK	CVE-2014-8361
Netgear R7000 and R6400	Netgear cig-bin Command Injection
DGN1000 Netgear routers	Netgear setup.cgi unauthenticated RCE
MVPower DVR	JAWS Webserver unauthenticated shell command execution
Huawei Router HG532	CVE-2017-17215
D-Link Devices	HNAP SoapAction-Header Command Execution
GPON Routers	CVE-2018-10561, CVE-2018-10562
D-Link Devices	UPnP SOAP TelnetD Command Execution
CCTV DVR	CCTV/DVR Remote Code Execution

IoCs

Malware Hashes

006965027c1f636295b5011a46905121

1bd4f62fdad18b0c140dce9ad750f6de

2560a86361257837b78d7ba289a031fb

2d2ffa0422db66640561c46b8e428267

2f8d6c0c6a449f3c074cfc0d6c8dbfc6

300f850c0186077550830fa35edddc4e

39434e0d800b62a72e8dfa202e2da9cd

3a103ab0da4d13ccc9ed2d612de71441

4dde761681684d7edad4e5e1ffdb940b

5b9b2a796c88da82d75553c48488b63f

635d926cace851bef7df910d8cb5f647

649e482199c9eb826fa0fca7016c325d

68bf06fb2a8cef72a61b01dcd10fd10d

6aa92a03083a19783ddf4e4913c230d3

781228e0a889c0624a5f1d8e9f5b0b30

849b165f28ae8b1cebe0c7430f44aff3

868180d3f78ae330c8ab4e6c20045930

8bfbda4203cfb4bb7aaeafe7afe9748a

8d207e2b6d13ebd5fc4430ef3670558f

8e81f08432ba7d64c67032a2a5580a48

92defd440acfd41595ce20c9107c3262

93be88ab0908a9359d7e5472ace22fe5

9a111588a7db15b796421bd13a949cd4

9c6539c9f5b3e831d5bcb1357d51d049

b08d4099b14e37ceda1923681a2f70f2

c16feda9ad177c8f7e6a07f57d84851f

c46327a65a1f9bf9c367fbda95f1bd22

c89a06d5b3a55a45b7d508e6b9152aa8

d107c5dc752cd262cd4d6c461c8583c4

d2b8a429bcf9e0eca54939e2cd4408dc

d71d01d469414e992ded9038ea761564

d96bbc2b1e5cc6b085bf04a8e487632e

dd4b6f3216709e193ed9f06c37bcc389

eda730498b3d0a97066807a2d98909f3

Top talking botnet nodes from most recent data

67.58.78.157

78.187.40.53

177.128.34.146

122.165.131.7

80.92.189.5

78.187.20.244

85.105.104.74

94.156.57.84

112.196.16.26

185.101.27.48

82.209.9.181

187.85.255.194

103.123.46.51

213.174.31.77

103.97.244.22

103.41.56.62

187.85.248.209

103.59.134.156

88.247.16.223

89.160.95.67

External References

This information is provided “as is” without any warranty or condition of any kind, either express or implied. Use of this information is at the end user’s own risk.