Break Out Of The Tinynuke Malware

New Tinynuke variant with a DGA in the wild

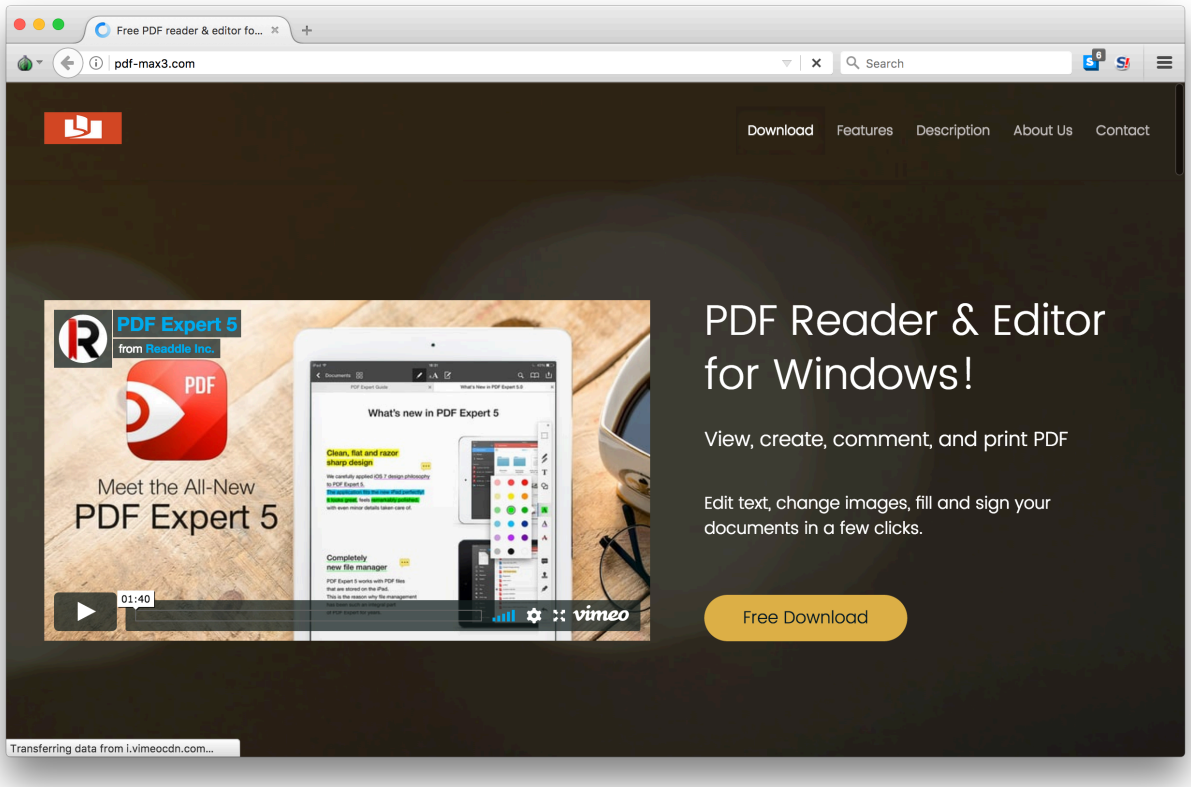
Tinynuke, or Nukebot malware, is a trojan able to perform man in the browser attacks against modern web browsers and equipped with the most common features needed by a bank trojan (e.g. Webinjects, Socks proxy, VNC, Remote command execution). This malware was in the spotlight in 2017 after the complete bot source code was leaked in March by someone claiming to be the author of the malware.

The leaked source contained a fully working bot, builder and botnet control panel and, as we have seen in past, leaks of working malware usually lead to that malware being used increasingly and new variants and adaptations of it start to emerge.

We at Bitsight constantly monitor the internet for new threats that may affect the security of organizations worldwide. A few months ago we first noticed an unknown DGA showing up in our network traffic analysis system that we could not identify. After some research we managed to identify the malware behind it, and found a malware operation targeting users in the United Kingdom and Canada using a new variant of Tinynuke with some improvements, such as the use of a DGA (Domain Generation Algorithm) for it’s command and control channel or asymmetric cryptography for C2 authentication.

The malware is being distributed through fake websites (fake product website or fake blogs) promoted through social networking and advertising, that contain links to malicious software installers.

The following is an example of one of these websites being used to distribute a trojanized pdf reader software installer.



Another interesting example is a fake cryptocurrency security advice blog that was set-up to make available a set of fake installers of known tools that can be used to make a PC more secure. This fake blog was reported to be promoted through facebook advertising by a user who found it suspicious. He wrote about it on his blog (see references). The following image shows this website:

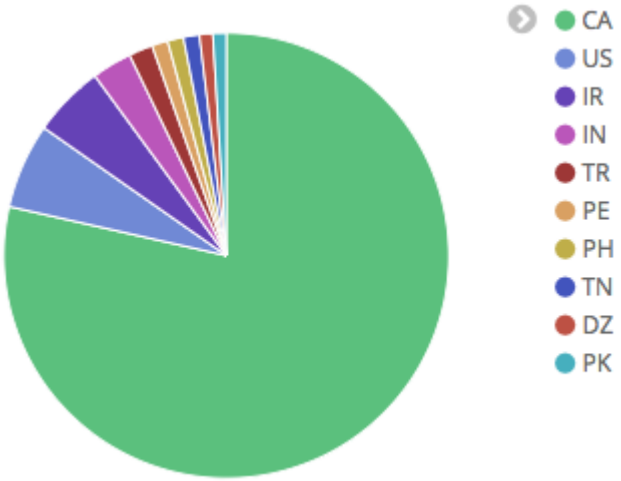
The software installers contain both the trojan and a legitimate software installer and are, in most cases, digitally signed by a company called “AGM 1980 Limited” with a valid certificate signed by Comodo CA.

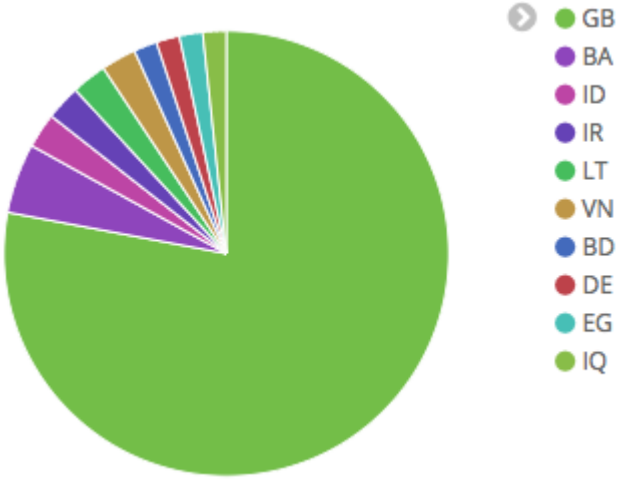
Once the user executes the fake installer:

- Requests administrative privileges to continue with the installation;
- Performs several checks to verify if it is running on a sandboxed environment (e.g. number of CPU’s, Installed RAM, Virtualization software);
- If the installer is running on a sandbox it proceeds with the legitimate tool installer and does not install the malware;
- If the installer is not on a sandbox it installs the malware and then proceeds with the legitimate installer;

After the trojan is installed and running, the user’s system is compromised and it’s data at risk.

In order to gather some data about the geographies targeted by the malware we have sinkholed it’s DGA domains and observed that there are currently at least 2 different botnets, each using a different TLD on the DGA and each targeting users in a specific country. In particular, one botnet seems to be targeting users in Canada and other users in the United Kingdom.





Webinjects and targets

Looking at the webinject configuration we found that besides a few test injects there were also specific injects for two targets. Amazon and Lloyds bank, as shown in the following excerpt (the full webinjects are listed in the IOC's section):

```
{
   "host": "*amazon*",
   "path": "/*",
   "content":
   [
      {
         "code": "<head>",
         "before": "",
         "after": "<div id=\"_brows.cap\" [...]SNIP[...]
      }
   ]
},
{
   "host": "*.lloydsbank.co.uk",
   "path": "/personal/*",
   "content":
   [
      {
         "code": "<head>",
         "before": "",
         "after": "<script>var home_link [...]SNIP[...]
      }
   ]
}
```


These injects include additional javascript from domains that were registered using the same email address that was used to register the hardcoded domains and the active DGA domains, indicating that these are possibly being developed by the same person or group.

It is also worth noting that besides these two webinjects, the normal bot behaviour is to collect HTTP POST request data on IE, Firefox and Chrome for all hosts not listed in the fg_blacklist object in the config. The following blacklist was found active:


```
"fg_blacklist":["*ocsp*.*","*symc*.com*","*clients*.google.com*","*telemetry.mozilla.org*","*facebook.co
m/ajax/*","*safebrowsing.google.com*","*services.mozilla.com*","*youtube.*","*google.*","*linkedin.*","*
facebook*","*twitch*","*googleapis*","*netflix*","*.gvt*","*reddit*","*.comet.*","*outlook.live.com*","*
fls-na.amazon.com*","*messenger.live.com*","*kixeye.com*","*mg.mail.yahoo.com*","*microsoft.com*","*outl
ook.office.com*","*clicktale.net*","*optimatic.com*","*twitter.com*","*gstatic.com*","*adnxs.com*","*ttv
nw.net*","*kijiji.ca*","*bilibili.com*","*avast.com*","*fortinet.com*","*i-techsolutions.ca*"]
```


The malware is mostly the same as the TinyNuke bot that was leaked. However, there are some new features, of which the following are the most relevant:

DGA

Perhaps the most interesting modification is the introduction of a DGA. This DGA increases the resiliency of the botnet against takedown requests and is triggered when the hardcoded C2 is unreachable.

Its implementation is very simple, it consists of the md5 of a number concatenated with the current day. The following is an example implementation of the DGA in python:

```
"""
Bitsight Technologies 2018
"""

import json
import hashlib
import argparse
from datetime import datetime


class NukebotDGA(object):
    def dga(self, datestr, tld):
        domains = []
        for i in range(1, 33):
            m = hashlib.md5()
            m.update(str(i).encode('utf-8') + datestr.encode('utf-
8'))
            domains.append(m.hexdigest() + '.' + tld)
        return domains

    def get_domains(self, tld, dt):
        return {
            "domains": [domain for domain in
self.dga(dt.strftime("%d.%m.%Y"), tld)],
            "family": "Nukebot",
            "date": dt.strftime("%Y-%m-%d"),
            "notes": {
                "tld": tld
            }
        }


if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="Nukebot DGA
domains generator")
    parser.add_argument("tld", help="tld")
    parser.add_argument("-d", "--date", help="date [DD-MM-YYYY]
(defaults to current day)")
    args = parser.parse_args()

    pdt = datetime.strptime(args.date, "%d-%m-%Y") if args.date
else datetime.now()
    dga = NukebotDGA().get_domains(args.tld, pdt)
    print json.dumps(dga)
```

C2 authentication using public RSA key

The bot uses asymmetric cryptography to authenticate the C2, as a measure to prevent botnet takeover. To do this, the malware:

- Generates a random 8 byte long string and calculates it’s md5 value;
- Encrypts this value using the C2 RSA Public key that is hard coded on the bot;
- Checks if the md5 of the send value is present in the response to the ping command;
- If it is, the malware stores this C2 as the active C2 in a global variable and communicates with it directly on further interactions;

The following public key was found in use:

```
-----BEGIN PUBLIC KEY-----MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgK
CAQEAxxUk/C3M413qwlO04xFJEbzBzFz7Sy+mv1bSq4uD2L3dCZLeDKgsdRm83N8y
0/kjqa1mv28l0SwgtmOA6Z020GM3CCfgHhv/1gIVWahHC8KKnfEmg4G1dUYR2C22l
tPsF/DPrruWk/kExUQtzAu3hWPz2Qe7ZhEAbcpvfYfXU6iXB6pN+iiRjg8wzqPGlw
oYNfOFy3HqePAW/IKKtYzEhCKD9PaIhAQQlJwcYVSybopHdL30lzMxKmop6I7kwxV
ieukaDLDQiU68nExc9FygU1TXPCkNN+BvgdNwjPrRNMX29GHwc2aZK9wnMXuO59WD
AY3M4invvdukhEcgEtIU2QIDAQAB-----END PUBLIC KEY-----
```

C2 Proxy layer

The bot has been modified to support what appears to be a proxy layer between the C2 and the infected bots. The HTTP requests now have the following format:

```
/proxy.php?b=<BOTID>&h=<HOSTNAME>
```

Sandbox/malware analysis detection

The new version includes a few self protection features not present in the leaked source. These are the same that exist in the fake software installers used to drop the malware:

- Check for sandbox related usernames by looking for the strings:

```
‘sandbox’ ,’virus’, ‘malware’, ‘nod’, ‘kas’, ‘av’, ‘kis’ ‘esset’
```

- Check physical memory on the device is higher than 1Gb;
- Check if the system has more than one CPU;
- Check if the Sleep command is being bypassed by calling Sleep and checking if the process actually sleeps;
- Check for VMWare tools registry keys;
- Check for VirtualBox guest additions files;
- Check if the user is connected through remote desktop;

This is an interesting evolution on the TinyNuke malware family and we will continue to monitor this threat as it progresses.

- https://securingtomorrow.mcafee.com/business/tinynuke-may-ticking-time-bomb/
- https://securelist.com/the-nukebot-banking-trojan-from-rough-drafts-to-real-threats/78957/
- https://krebsonsecurity.com/2017/04/self-proclaimed-nuclear-bot-author-weighs-u-s-job-offer/
- https://securityintelligence.com/the-nukebot-trojan-a-bruised-ego-and-a-surprising-source-code-leak/
- https://rootvideochannel.blogspot.pt/2017/12/suspicious-website-cryptocurrencysecuri.html

e2a3bf38387c751bcb971f0234a7a89f74f2b7c807bf6503b4b58fcfbaafa1d6
