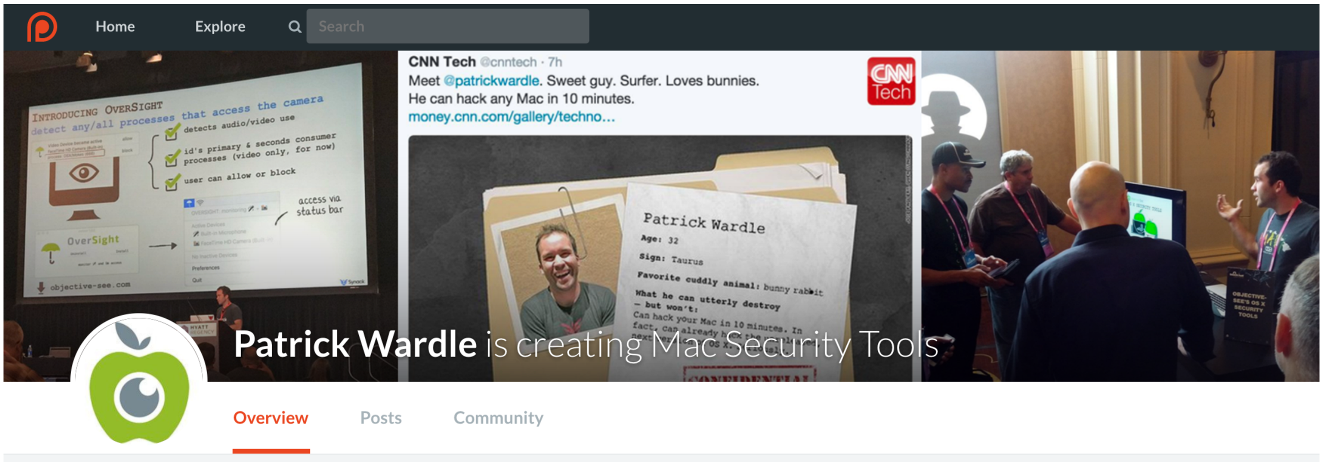
# Ay MaMi

› Analyzing a New macOS DNS Hijacker: OSX/MaMi

01/11/2018

love these blog posts? support my tools & writing on patreon! Mahalo :)



CNN Tech @cnntech · 7h
Meet @patrickwardle. Sweet guy. Surfer. Loves bunnies.
He can hack any Mac in 10 minutes.
money.cnn.com/gallery/techno...

Patrick Wardle is creating Mac Security Tools

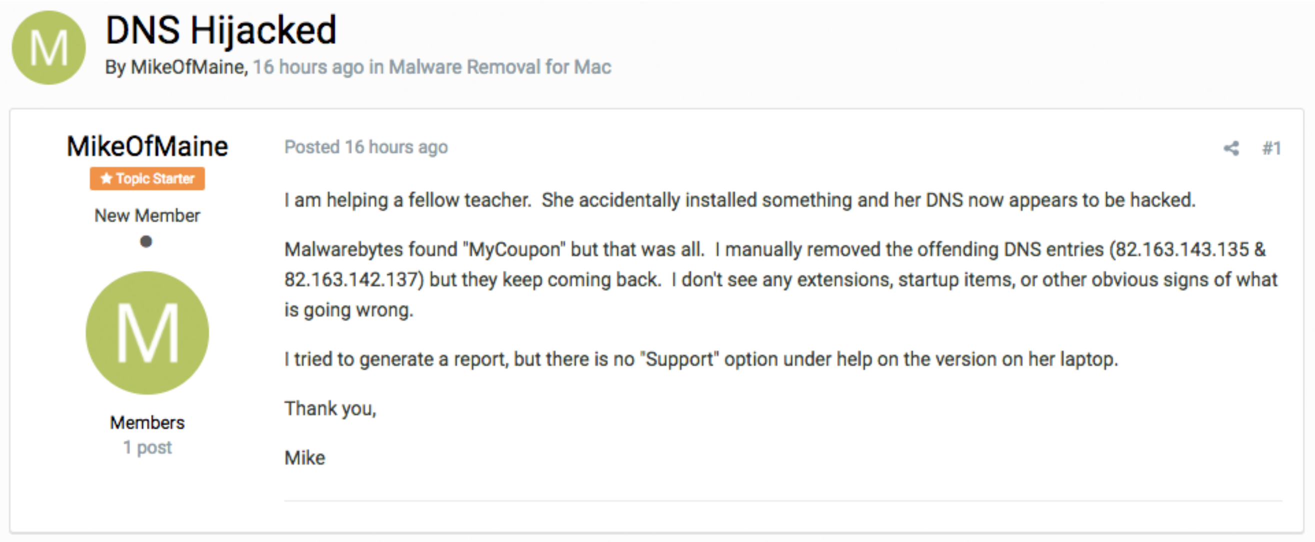
2018 is barely two weeks old, and already it looks like we've got new piece of macOS malware! Hooray :)

Want to play along? I've shared both the malware's binary executable ('MaMi'), which can be downloaded here (password: infect3d).

Please don't infect yourself!

## Background

Earlier today (01/11), someone on MalwareBytes' forum created a post titled "DNS Hijacked":



DNS Hijacked

By MikeOfMaine, 16 hours ago in Malware Removal for Mac

MikeOfMaine

Posted 16 hours ago

I am helping a fellow teacher. She accidentally installed something and her DNS now appears to be hacked.

Malwarebytes found "MyCoupon" but that was all. I manually removed the offending DNS entries (82.163.143.135 & 82.163.142.137) but they keep coming back. I don't see any extensions, startup items, or other obvious signs of what is going wrong.

I tried to generate a report, but there is no "Support" option under help on the version on her laptop.

Thank you,

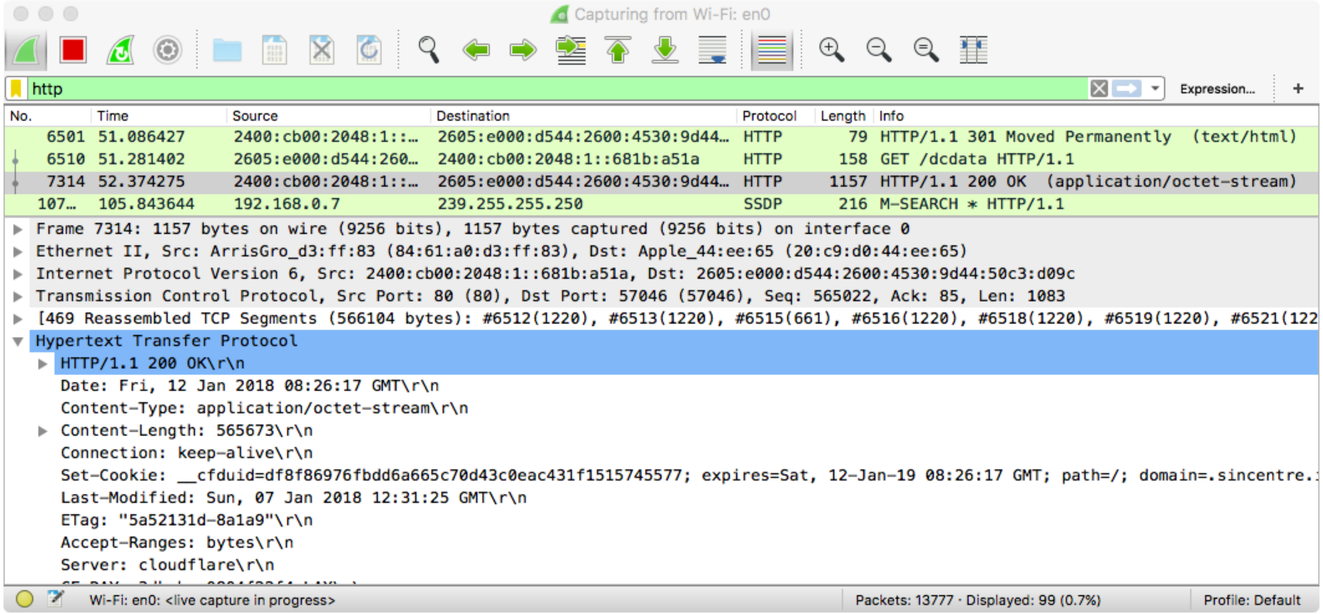
Mike

As I hadn't seen an answer to MikeOfMaine', and as far as I'm aware there haven't been any recent macOS malware that hijacks DNS settings - so I was intrigued! So without further adieu, let's dive in to analyzing (what I'm calling) OSX/MaMi!

## Analysis

Though currently I am unaware of the malware's infection vector, it is hosted on various sites such as http://regardens.info:

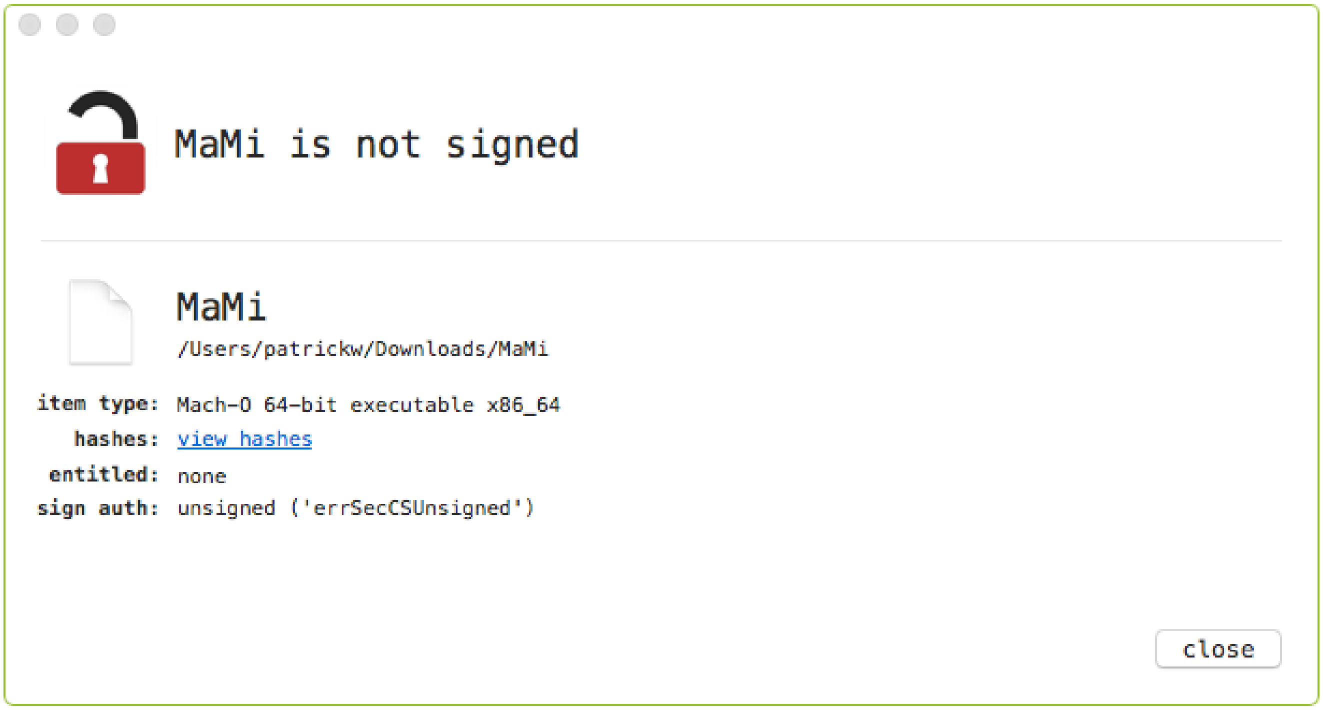
```
curl -L http://regardens.info/ > MaMi
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   178    0   178    0     0    381      0 --:--:-- --:--:-- --:--:--   381
100  552k  100  552k    0     0   314k      0  0:00:01  0:00:01 --:--:--  581k

MacBookPro:Downloads patrickw$ file MaMi
MaMi: Mach-O 64-bit executable x86_64
```



```
http
```

```
No.    Time        Source                   Destination                      Protocol  Length  Info
  6501 51.086427   2400:cb00:2048:1::…      2605:e000:d544:2600:4530:9d44…   HTTP        79   HTTP/1.1 301 Moved Permanently  (text/html)
  6510 51.281402   2605:e000:d544:260…      2400:cb00:2048:1::681b:a51a      HTTP       158   GET /dcdata HTTP/1.1
  7314 52.374275   2400:cb00:2048:1::…      2605:e000:d544:2600:4530:9d44…   HTTP      1157   HTTP/1.1 200 OK  (application/octet-stream)
 107… 105.843644   192.168.0.7              239.255.255.250                  SSDP       216   M-SEARCH * HTTP/1.1
▶ Frame 7314: 1157 bytes on wire (9256 bits), 1157 bytes captured (9256 bits) on interface 0
▶ Ethernet II, Src: ArrisGro_d3:ff:83 (84:61:a0:d3:ff:83), Dst: Apple_44:ee:65 (20:c9:d0:44:ee:65)
▶ Internet Protocol Version 6, Src: 2400:cb00:2048:1::681b:a51a, Dst: 2605:e000:d544:2600:4530:9d44:50c3:d09c
▶ Transmission Control Protocol, Src Port: 80 (80), Dst Port: 57046 (57046), Seq: 565022, Ack: 85, Len: 1083
▶ [469 Reassembled TCP Segments (566104 bytes): #6512(1220), #6513(1220), #6515(661), #6516(1220), #6518(1220), #6519(1220), #6521(122
▼ Hypertext Transfer Protocol
  ▶ HTTP/1.1 200 OK\r\n
    Date: Fri, 12 Jan 2018 08:26:17 GMT\r\n
    Content-Type: application/octet-stream\r\n
  ▶ Content-Length: 565673\r\n
    Connection: keep-alive\r\n
    Set-Cookie: __cfduid=df8f86976fbdd6a665c70d43c0eac431f1515745577; expires=Sat, 12-Jan-19 08:26:17 GMT; path=/; domain=.sincentre.
    Last-Modified: Sun, 07 Jan 2018 12:31:25 GMT\r\n
    ETag: "5a52131d-8a1a9"\r\n
    Accept-Ranges: bytes\r\n
    Server: cloudflare\r\n
Wi-Fi: en0: <live capture in progress>        Packets: 13777 · Displayed: 99 (0.7%)        Profile: Default
```

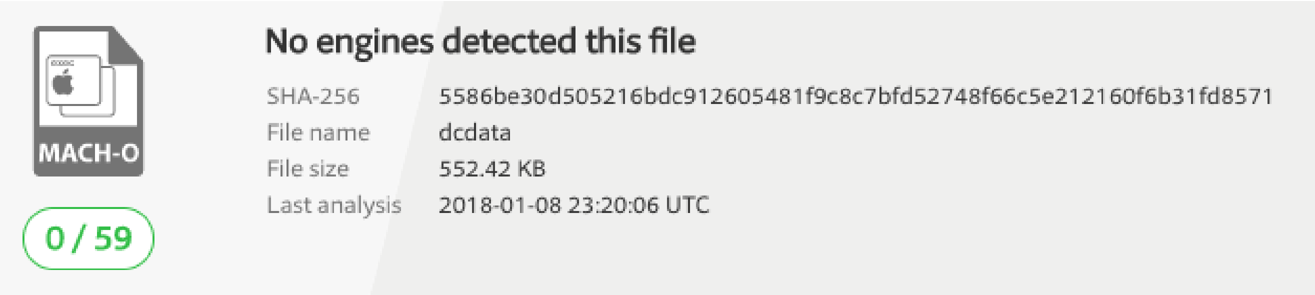
As shown by WhatsYourSign, nothing too special about the file; it's an unsigned Mach-O 64-bit executable:


MaMi is not signed

MaMi
/Users/patrickw/Downloads/MaMi

item type: Mach-O 64-bit executable x86_64
hashes: view hashes
entitled: none
sign auth: unsigned ('errSecCSUnsigned')

As is often the case with new malware, it's currently marked as 'clean' by all 59 engines on VirusTotal (this will hopefully change shortly as AV products start adding detections):


No engines detected this file

SHA-256 5586be30d505216bdc912605481f9c8c7bfd52748f66c5e212160f6b31fd8571
File name dcdata
File size 552.42 KB
Last analysis 2018-01-08 23:20:06 UTC

And speaking of 'new' if we load the malware's binary in a disassembler, we find an app version of 1.1.0, which (due to such low version number), may seem to indicate the malware likely hasn't been around for too long.

```
000000010003818f  db  "AppVersion: %@\nAppBuild: %@", 0
00000001000381ab  db  "1.1.0", 0
00000001000381b1  db  "0", 0
```

Before we dig further into the disassembly, let's dump the Objective-C class names and methods, as often this can allow us to quickly gain insight the malware's (likely) capabilities or at least guide our analysis.

I use J. Levin's invaluable jtool utility to dump such info:

```
$ ./jtool -d objc -v MaMi

@interface AppDelegate
...
/* 2 - 0x100001e0b */ - setupCert;
...
/* 7 - 0x1000027bc */ - setupDNS;
...
/* 9 - 0x100002a97 */ - takeScreenshotAt:;
...
/* 22 - 0x1000049d8 */ - mouseClick:;
...
/* 24 - 0x100004ac5 */ - runAppleScript:;


@interface SBMaMiSettings :
...
/* 2 - 0x10000518b */ - initMaMiSettings;
...
/* 9 - 0x100005385 */ - programArguments;
...
/* 11 - 0x1000053a7 */ - runAtLoad;
...
/* 25 - 0x10000548f */ - launchOnlyOnce;


@interface SBNetwork :
...
/* 0 - 0x10000d2e5 */ + downloadFile:atPath:;
/* 1 - 0x10000d4a8 */ + sendAsyncRequestWithUrls:andMethod:andBody:;


@interface SBFileSystem : ?
/* 0 - 0x10002407e */ + writeString:toPath:;
...
/* 8 - 0x1000247fb */ + runCmd:andPipeToCmd:withParams:andParams2:;
/* 9 - 0x100024b07 */ + runCmd:withParams:;
/* 10 - 0x100024b23 */ + runCmd:withParams:andUser:;


@interface SBCryptoSystem : ?
/* 0 - 0x100026731 */ + isAdmin;  // Protocol 129824ad7
/* 1 - 0x100026745 */ + elevatePrivilegesWithParams:;  // Protocol 1298247ca
/* 2 - 0x1000267aa */ + relaunchWithPrivilegesAndParams:;
```

Some very interesting methods! Of course we'll continue our analysis to confirm, but seems this malware is indeed a 'dns hijacker' (method: setupDNS), with a host of other abilities such as:

- taking screenshots

- generating simulated mouse events

- perhaps persists as a launch item (programArguments, runAtLoad)
- downloading & uploading files
- executing commands
- ...and more!

Jumping back to the disassemlby, within the application's main entrypoint (-[AppDelegate applicationDidFinishLaunching:]), we see a massive encrypted string that is passes to a setDefaultConfiguration: method:

```
 [SBConfigManager
setDefaultConfiguration:@"uZmgulcipekSbayTO9ByamTUu_zVtsflazc2Nsuqgq0dXko
 OzKMJMNTULoLpd-QV9qQy6VRluzRXqWOGscgheRvikLkPRzs1pJbey2QdaUSXUZCX-
UNERrosul22NsW2vYpS7HQO4

VG5l8qic3rSH_fAhxsBXpEe557eHIr245LUYcEIpemnvSPTZ_lNp2XwyOJjzcJWirKbKwtc3Q61pDwTzKvE0..
```

Applying some classified decryption methods I learned as an intern working in NSA's Cryptanalysis and Exploitation Services (CES) group - it was trivial to decrypt this configuration data. I'm totally kidding - not about the internship - but about how to decrypt. Just step over that method in a debugger (lldb) and the data is sitting decrypted in memory:

```
# lldb MaMi
(lldb) target create "MaMi"
Current executable set to 'MaMi' (x86_64).

...

(lldb) po $rax
{
    defaults =     {
        affiliate = "";
        build = 0;
        "compilation_id" = 0;
        "confirmation_end_time" = 0;
        "confirmation_start_time" = 0;
        "download_complete_time" = 0;
        "download_location" = "";
        "download_retry_count" = 0;
        "download_start_time" = 0;
        "download_url" = "";
        "exception_id" = 0;
        "execute_location" = "";
        "execution_end_time" = 0;
        "execution_start_time" = 0;
        "exit_code" = 0;
        "external_id" = 0;
        "file_crc" = 0;
        "hardware_id" = 0;
        "hosts_active" = "";
        "installer_id" = 0;
        "is_admin" = false;
        "old_secondary_dns" = "";
        "os_build" = 0;
        "os_id" = 0;
        "product_id" = 0;
        "product_name" = "";
        "publisher_id" = 0;
        "register_date" = 0;
        "register_dsrc" = 0;
        "report_id" = 0;
        "run_args" = "";
        "screen_x" = 0;
        "screen_y" = 0;
        "secondary_dns" = "";
        "service_pack" = 0;
        "session_id" = 0;
        status = 0;
        "step_id" = 0;
        tag = "";
        tracker = "";
        "user_time" = 0;
        "validate_end_time" = 0;
        "validate_start_time" = 0;
        version = 0;
    };
    dnsChanger =     {
        affiliate = "";
        "blacklist_dns" =         (
        );
        encrypt = true;
        "external_id" = 0;
        "product_name" = dnsChanger;
        "publisher_id" = 0;
        raw = true;
        reports =         {
            "dnsChanger_activity" =             {
                async = false;
                body = "r={dnsChanger->reports->dnsChanger_activity->template}&rc=
{dnsChanger}";
                "connection_timeout" = 5;
                domains =                 (
                    "honouncil.info",
                    "gorensin.info"
                );
                "http_headers" =                 (
                                        {
                        name = "Content-Type";
                        value = "application/x-www-form-urlencoded";
                    },
                                        {
                        name = "User-Agent";
                        value = "";
                    }
                );
                "query_string" = "r={dnsChanger->reports->dnsChanger_activity-
>template}&rc={dnsChanger}";
                "request_method" = 1;
                "request_timeout" = 5;
                "retry_count" = 2;
                "send_port" = 80;
                "send_protocol" = http;
                template =                 {
                    affiliate = "%affiliate%";
                    build = "%build%";
                    "compilation_id" = "%compilation_id%";
                    dns =                     {
                        "hosts_active" = "%hosts_active%";
                        "hosts_config" = "[templates->secondary_dns]";
                    };
                    encrypt = true;
                    "exception_id" = "%exception_id%";
                    expand = true;
                    "external_id" = "%external_id%";
                    "hardware_id" = "%hardware_id%";
                    "is_admin" = "%is_admin%";
                    "old_dns" =                     {
                        "hosts_active" = "%hosts_active%";
                        "hosts_config" = "[templates->old_secondary_dns]";
                    };
                    "os_build" = "%os_build%";
                    "os_id" = "%os_id%";
                    "product_name" = "%product_name%";
                    "publisher_id" = "%publisher_id%";
                    "register_date" = "%register_date%";
                    "register_dsrc" = "%register_dsrc%";
                    "report_id" = "%report_id%";
                    "report_name" = "dnsChanger_activity";
                    "report_type" = 8;
                    "screen_x" = "%screen_x%";
                    "screen_y" = "%screen_y%";
                    "service_pack" = "%service_pack%";
                    "session_id" = "%session_id%";
                    status = "%status%";
                    tag = "%tag%";
                    tracker = "%tracker%";
                    "user_time" = "%user_time%";
                    version = "%version%";
                };
                "url_path" = "";
            };
            "time_report" =             {
                async = false;
                body = "r={dnsChanger->reports->time_report->template}&rc=
{dnsChanger}";
                "connection_timeout" = 5;
                domains =                 (
                    "squartera.info"
                );
                "http_headers" =                 (
                                        {
                        name = "Content-Type";
                        value = "application/x-www-form-urlencoded";
                    },
                                        {
                        name = "User-Agent";
                        value = "";
                    }
                );
                "query_string" = "";
                "request_method" = 2;
                "request_timeout" = 5;
                "retry_count" = 2;
                "send_port" = 80;
                "send_protocol" = http;
                template =                 {
                    affiliate = "%affiliate%";
                    build = "%build%";
                    "compilation_id" = "%compilation_id%";
                    dns =                     {
                        "hosts_active" = "%hosts_active%";
                        "hosts_config" = "[templates->secondary_dns]";
                    };
                    encrypt = true;
                    "exception_id" = "%exception_id%";
                    expand = true;
                    "external_id" = "%external_id%";
                    "hardware_id" = "%hardware_id%";
                    "is_admin" = "%is_admin%";
                    "os_build" = "%os_build%";
                    "os_id" = "%os_id%";
                    "product_name" = "%product_name%";
                    "publisher_id" = "%publisher_id%";
                    "report_id" = "%report_id%";
                    "report_name" = "time_request";
                    "screen_x" = "%screen_x%";
                    "screen_y" = "%screen_y%";
                    "service_pack" = "%service_pack%";
                    "session_id" = "%session_id%";
                    status = "%status%";
                    tag = "%tag%";
                    tracker = "%tracker%";
                    "user_time" = "%user_time%";
                    "verification_id" = "%verification_id%";
                    version = "%version%";
                };
                "url_path" = "";
            };
        };

        "setup_dns" =         (
            "82.163.143.135",
            "82.163.142.137"
        );

        "shared_storage" = "/Users/%USER_NAME%/Library/Application Support";
        "storage_timeout" = 120;
        tag = "";
        "timeout_dns" =         {
            "high_timeout" = 1;
            "low_timeout" = "0.3";
            "medium_timeout" = "0.5";
        };
        tracker = "";
    };
    "installer_id" = 1359747970602718687;
    "report_templates" =     {
        "report_config" =         {
            async = false;
            body = "";
            "connection_timeout" = 5;
            domains =             (
                "domain1.com",
                "domain2.com"
            );
            "http_headers" =             (
                                {
                    name = "Content-Type";
                    value = "application/x-www-form-urlencoded";
                },
                                {
                    name = "User-Agent";
            value = "";
        }
    );
    "query_string" = "";
    "request_method" = 2;
    "request_timeout" = 5;
    "retry_count" = 2;
    "send_port" = 80;
    "send_protocol" = http;
};
"report_config2" =         {
    async = true;
    body = "";
    "connection_timeout" = 5;
    domains =             (
        "domain1.com",
        "domain2.com"
    );
    "http_headers" =             (
                        {
            name = "Content-Type";
            value = "application/x-www-form-urlencoded";
        },
                        {
            name = "User-Agent";
            value = "";
        }
    );
    "query_string" = "";
    "request_method" = 2;
    "request_timeout" = 5;
    "retry_count" = 2;
    "send_port" = 80;
    "send_protocol" = http;
    "url_path" = "";
};
"report_template1" =         {
    affiliate = "%affiliate%";
    build = "%build%";
    "compilation_id" = "%compilation_id%";
    dns =             {
        "hosts_active" = "%hosts_active%";
        "hosts_config" = "[templates->secondary_dns]";
    };
    "exception_id" = "%exception_id%";
    "external_id" = "%external_id%";
    "hardware_id" = "%hardware_id%";
    "is_admin" = "%is_admin%";
    "os_build" = "%os_build%";
    "os_id" = "%os_id%";
    "product_name" = "%product_name%";
    "publisher_id" = "%publisher_id%";
    "report_id" = "%report_id%";
    "screen_x" = "%screen_x%";
    "screen_y" = "%screen_y%";
```

```
            "service_pack" = "%service_pack%";
            "session_id" = "%session_id%";
            status = "%status%";
            tag = "%tag%";
            tracker = "%tracker%";
            "user_time" = "%user_time%";
            version = "%version%";
        };
        "report_template2" =         {
            affiliate = "%affiliate%";
            build = "%build%";
            "compilation_id" = "%compilation_id%";
            dns =             {
                "hosts_active" = "%hosts_active%";
                "hosts_config" = "[templates->secondary_dns]";
            };
            "exception_id" = "%exception_id%";
            "external_id" = "%external_id%";
            "hardware_id" = "%hardware_id%";
            "is_admin" = "%is_admin%";
            "os_build" = "%os_build%";
            "os_id" = "%os_id%";
            "product_name" = "%product_name%";
            "publisher_id" = "%publisher_id%";
            "register_date" = "%register_date%";
            "register_dsrc" = "%register_dsrc%";
            "report_id" = "%report_id%";
            "screen_x" = "%screen_x%";
            "screen_y" = "%screen_y%";
            "service_pack" = "%service_pack%";
            "session_id" = "%session_id%";
            status = "%status%";
            tag = "%tag%";
            tracker = "%tracker%";
            "user_time" = "%user_time%";
            version = "%version%";
        };
    };
    templates =     {
        "old_secondary_dns" =         {
            "fill_template" = "%old_secondary_dns%";
            "fill_type" = string;
        };
        "secondary_dns" =         {
            "fill_template" = "%secondary_dns%";
            "fill_type" = string;
        };
    };
    version = 1;
}
```

Ok, that's a lot of configuration data! The most interesting part is probably the 'setup_dns' array:

```
 "setup_dns" = (
 "82.163.143.135",
 "82.163.142.137"
 );
```

...we'll see those DNS addresses used shortly!

In lldb we can set a breakpoints on methods of interest such as setupCert and setupDNS methods;

```
# lldb MaMi

(lldb) b -[AppDelegate setupCert]
Breakpoint 1: where = dcdata`-[AppDelegate setupCert], address = 0x0000000100001e0b

(lldb) b -[AppDelegate setupDNS]
Breakpoint 2: where = dcdata`-[AppDelegate setupDNS], address = 0x00000001000027bc
```

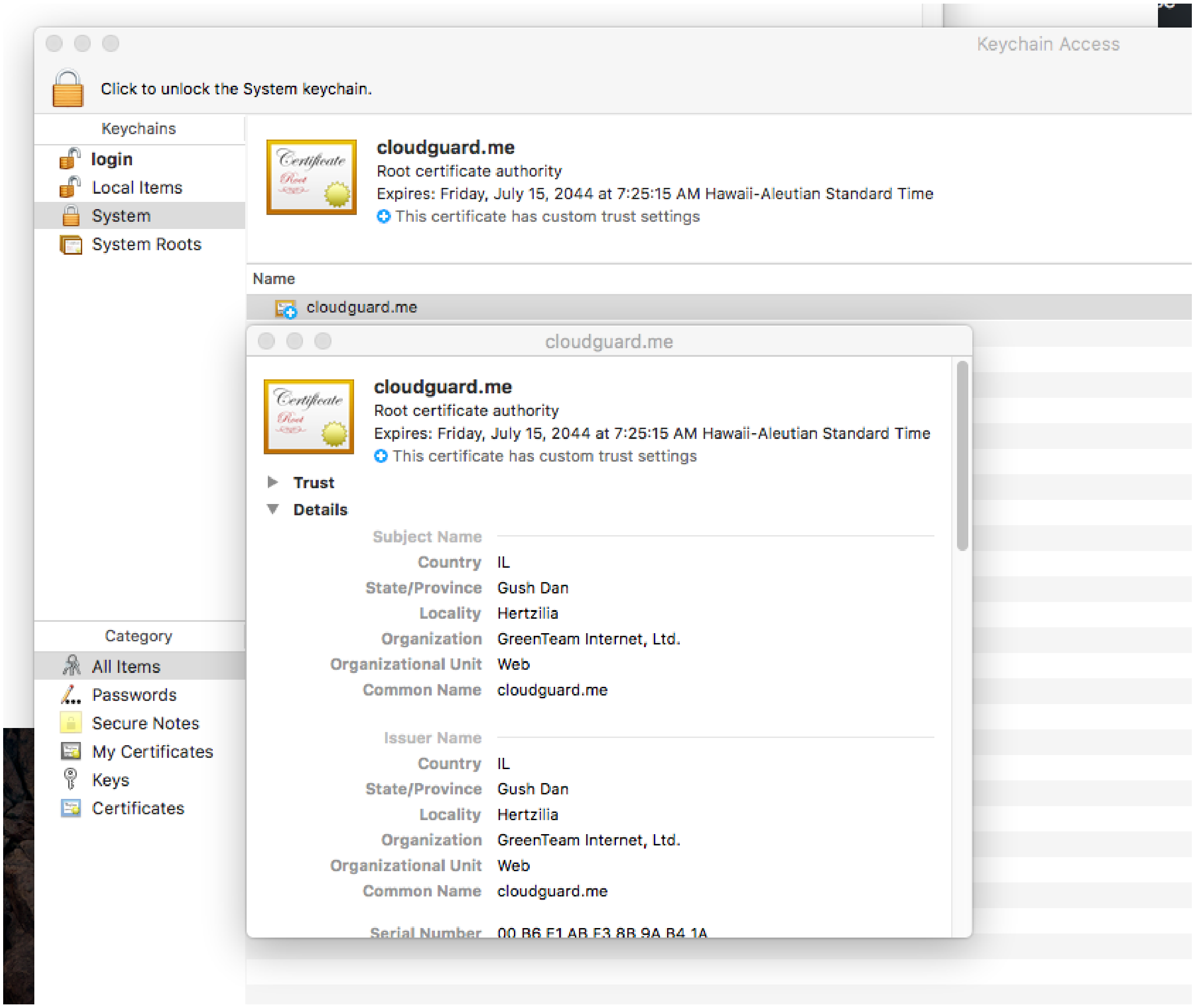
Once these breakpoints are hit, we can step thru the each instruction, or as I had fired up ProcInfo, the open-source process monitor I recently wrote (on github: ProcInfo) just let the malware run to see what it does. I'm voting for the latter as it's almost midnight.

```
# ./procInfo
starting process monitor
process monitor enabled...

pid: 1294
path: /usr/bin/security
args: (
    "/usr/bin/security",
    "add-trusted-cert",
    "-d",
    "-r",
    trustRoot,
    "-k",
    "/Library/Keychains/System.keychain",
    "/Users/user/Desktop/dcdata.bin"
)
```

First we see the malware invoking the security tool to install a new certificate (dcdata.bin) it's downloaded from the internet. Let's take a peak at this cert:

```
$ openssl x509 -inform der -in dcdata.bin -out dcdata.pem
$ openssl x509 -in dcdata.pem -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: b6:e1:ab:f3:8b:9a:b4:1a
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=IL, ST=Gush Dan, L=Hertzilia, O=GreenTeam Internet, Ltd.,
                OU=Web, CN=cloudguard.me
        Validity
            Not Before: Jul 23 17:25:15 2014 GMT
            Not After : Jul 15 17:25:15 2044 GMT
        Subject: C=IL, ST=Gush Dan, L=Hertzilia, O=GreenTeam Internet, Ltd.,
                 OU=Web, CN=cloudguard.me
        ...

$ openssl x509 -in dcdata.pem -fingerprint -noout
SHA1 Fingerprint=26:D9:E6:07:FF:F0:C5:8C:78:44:B4:7F:F8:B6:E0:79:E5:A2:22:0E
```

We can also view the (now installed) certificate via the 'Keychain Access' app. It's in the System keychain as a root certificate authority....MitM anybody?!



**cloudguard.me**
Root certificate authority
Expires: Friday, July 15, 2044 at 7:25:15 AM Hawaii-Aleutian Standard Time
This certificate has custom trust settings

Subject Name
Country IL
State/Province Gush Dan
Locality Hertzilia
Organization GreenTeam Internet, Ltd.
Organizational Unit Web
Common Name cloudguard.me

Issuer Name
Country IL
State/Province Gush Dan
Locality Hertzilia
Organization GreenTeam Internet, Ltd.
Organizational Unit Web
Common Name cloudguard.me

Serial Number 00 B6 F1 AB F3 8B 9A B4 1A

Back to process monitoring:

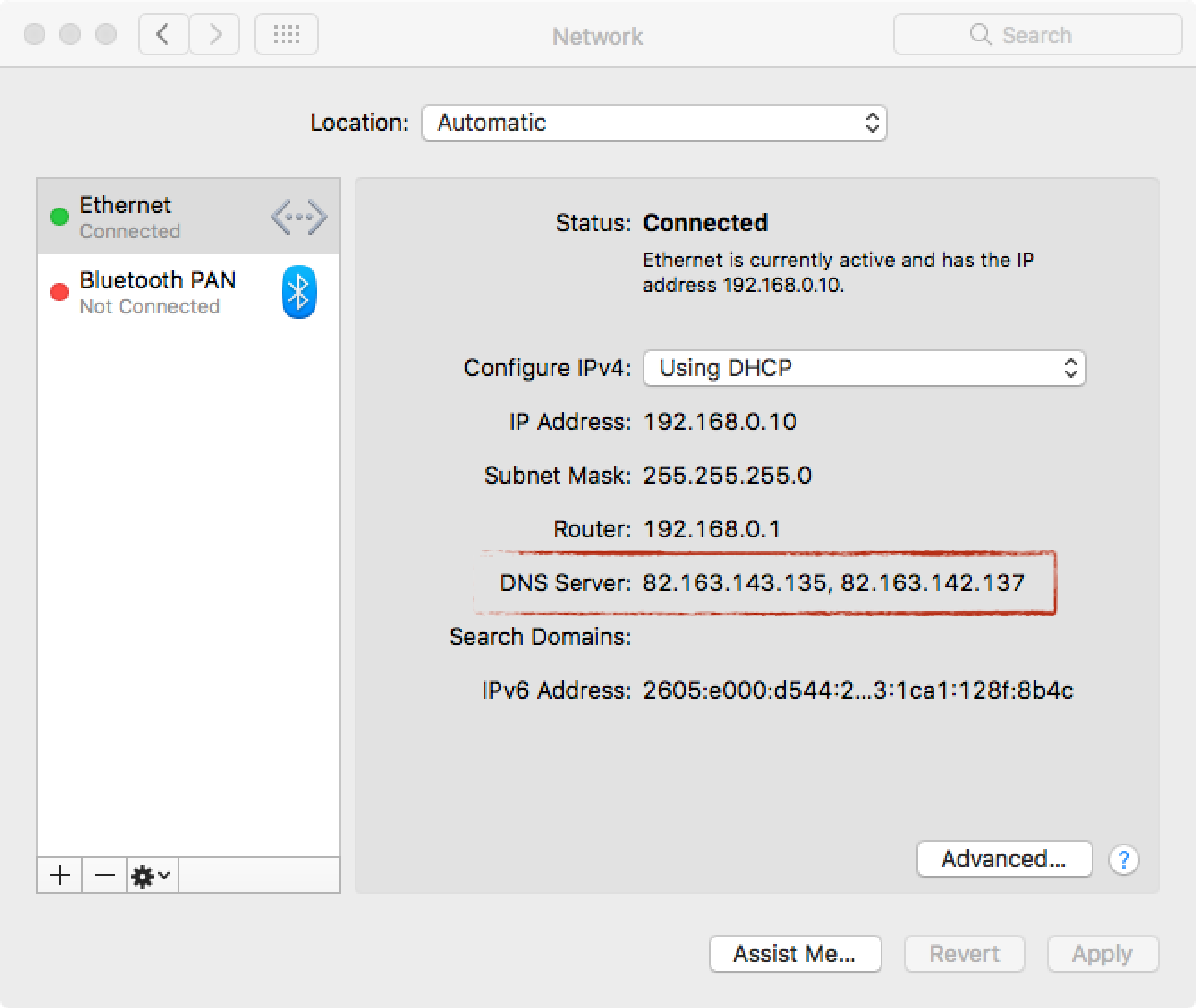
```
# ./procInfo

process start:
pid: 1177
path: /bin/cp
args: (
    "/bin/cp",
    "/Library/Preferences/SystemConfiguration/preferences.plist",
    "/Library/Preferences/SystemConfiguration/preferences.plist.old"
)
```

Interesting! It's mucking with the SystemConfiguration/preferences.plist file. What's in there? If you guessed DNS settings - you're right!

And remember the two DNS addresses from the decrypted config data? 82.163.143.135 and 82.163.142.137, they've been added to the plist file:

```
$ grep -B 4 -A 2 82. /Library/Preferences/SystemConfiguration/preferences.plist
DNS

        ServerAddresses

                82.163.143.135
                82.163.142.137
```

If you're more inclined to use the UI, you can see these changes via the System Preference app (Network pane):



Location: Automatic

Status: **Connected**

Ethernet is currently active and has the IP address 192.168.0.10.

Configure IPv4: Using DHCP

IP Address: 192.168.0.10

Subnet Mask: 255.255.255.0

Router: 192.168.0.1

DNS Server: 82.163.143.135, 82.163.142.137

Search Domains:

IPv6 Address: 2605:e000:d544:2...3:1ca1:128f:8b4c

So, the DNS settings on the infected host have been hijacked as well.

What about the other interesting methods? (e.g. takeScreenshotAt, mouseClick, runAppleScript). We in my brief reversing/analysis/debugging session I didn't see them being executed. Moreover, though the malware has an embedded launch item plist it didn't attempt to persist (though as it's altered system settings, it really doesn't need to hang around - in fact it does self-delete). When I coerced the malware to execute the method that modifed the launch item plist, initMaMiSettings, the value it configured in the ProgramArguments key - which tells the OS what to persistently execute - was simply: ls -la && sleep 28 && ls:

```
# lldb MaMi
(lldb) po $rax
{
    AbandonProcessGroup = "<key>AbandonProcessGroup</key><true/>";
    FooterStage = "</dict></plist>";
    HeaderStage = "<?xml version=\"1.0\" encoding=\"UTF-8\"?><!DOCTYPE plist PUBLIC
\"-
    //Apple//DTD PLIST 1.0//EN\" \"http://www.apple.com/DTDs/PropertyList-1.0.dtd\">
    <plist version=\"1.0\"><dict>";
    KeepAlive = "<key>KeepAlive</key><true/>";
    LabelStage = "<key>Label</key><string>%Label%</string>";
    ProgramArguments = "<key>ProgramArguments</key><array><string>/bin/sh</string>
    <string>-c</string><string>%ProgramArguments%</string></array>";
    RunAtLoad = "<key>RunAtLoad</key><true/>";
    ...
}

(lldb) po %$rsi
ls -la && sleep 28 && ls
```

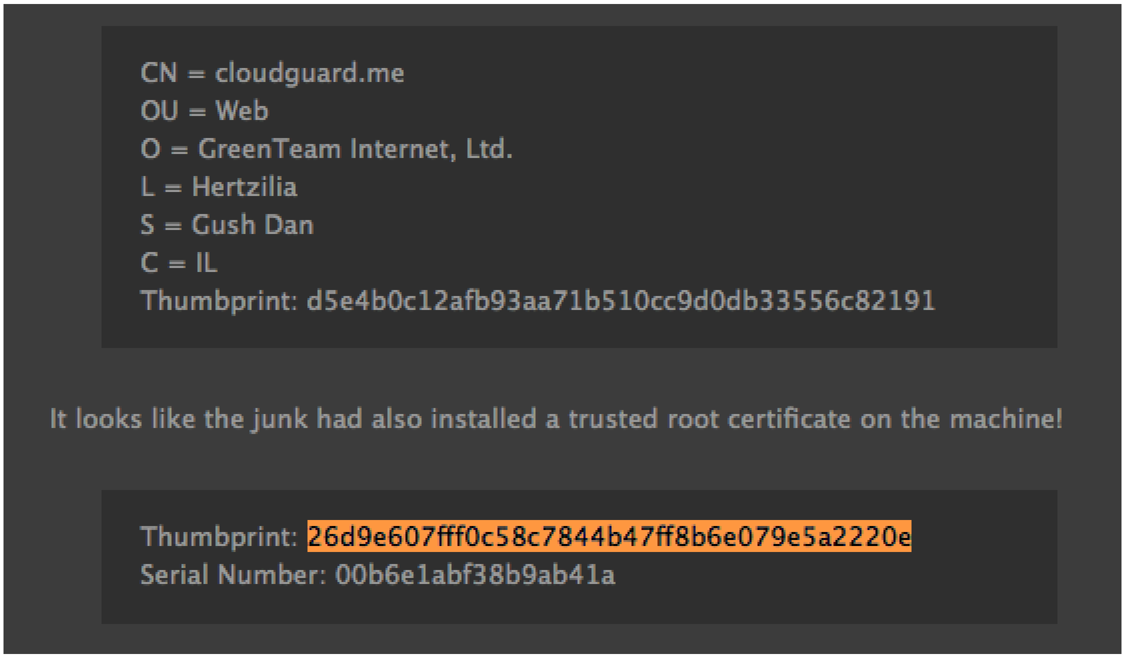
Perhaps in order for the methods to be executed or for the malware to be persisted, requires some attack-supplied input, or other preconditions that just weren't met in my VM. I'll keep digging!

### (Windows) Relatives

After chatting with @noarfromspace, about this malware, he dug up an interesting article from 2015. Titled, "The mystery of 82.163.143.172 and 82.163.142.174", the article dicusses a piece of Windows malware named DNSUnlocker that also hijacked DNS settings on Windows systems. This DNSUnlocker malware seems closely related to OSX/MaMi for a few reasons:

- DNS servers:
  DNSUnlocker, hijacks Windows victim's DNS servers to: **82.163.143.**172 and **82.163.142.**174
  OSX/MaMi, hijacks Mac victim's DNS servers to: **82.163.143.**135 and **82.163.142.**137

- Certificate:
  The certifcate installed by both malware specimens is the same:



```
CN = cloudguard.me
OU = Web
O = GreenTeam Internet, Ltd.
L = Hertzilia
S = Gush Dan
C = IL
Thumbprint: d5e4b0c12afb93aa71b510cc9d0db33556c82191
```

It looks like the junk had also installed a trusted root certificate on the machine!

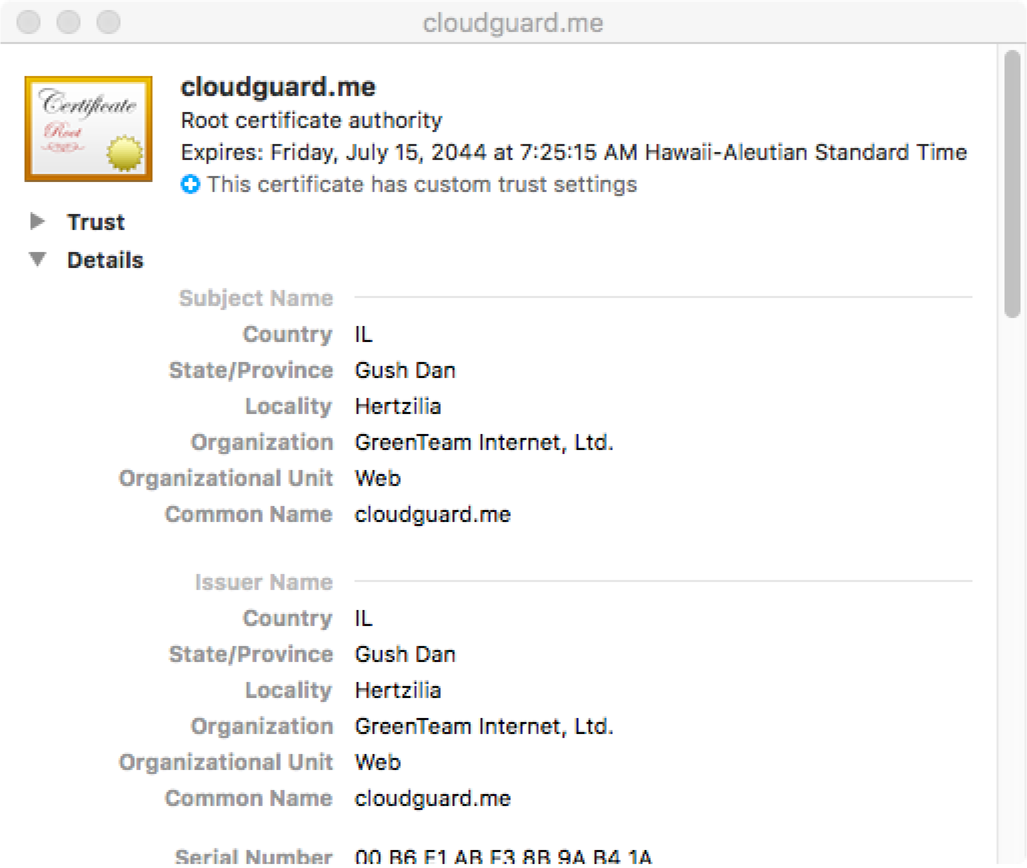
```
Thumbprint: 26d9e607fff0c58c7844b47ff8b6e079e5a2220e
Serial Number: 00b6e1abf38b9ab41a
```

Clearly DNSUnlocker, while older (circa 2015) and Windows only, is closely related to OSX/MaMi. If I had to guess, I'd say it's likely OSX/MaMi is a (fully re-written?) macOS version of DNSUnlocker, with a lot of extra macOS-specific evilness.

Conclusions
Ok, that's a wrap. OSX/MaMi isn't particular advanced - but does alter infected systems in rather nasty and persistent ways. By installing a new root certifcate and hijacking the DNS servers, the attackers can perform a variety of nefarious actions such as man-in-the-middle'ing traffic (perhaps to steal credentials, or inject ads).

Let's end with some Q&A!
Q: How do I get infected?
A: At this time, this is unknown. However, it's likely the attacker are using (rather lame) methods such as malicious email, web-based fake security alerts/popups, or social-engineering type attacks to target mac users

Q: How do I know if I'm infected?
A: Check your DNS settings, looking to see if they've been set to 82.163.143.135 and 82.163.142.137. You can check via the terminal (e.g. networksetup -getdnsservers Wi-Fi), or via the System Preferences app (Network pane). Also check for malicious cloudguard.me certifcate, which if installed, will appear in the System Keychain:



**cloudguard.me**

Root certificate authority

Expires: Friday, July 15, 2044 at 7:25:15 AM Hawaii-Aleutian Standard Time

This certificate has custom trust settings

▶ Trust

▼ Details

Subject Name

Country IL
State/Province Gush Dan
Locality Hertzilia
Organization GreenTeam Internet, Ltd.
Organizational Unit Web
Common Name cloudguard.me

Issuer Name

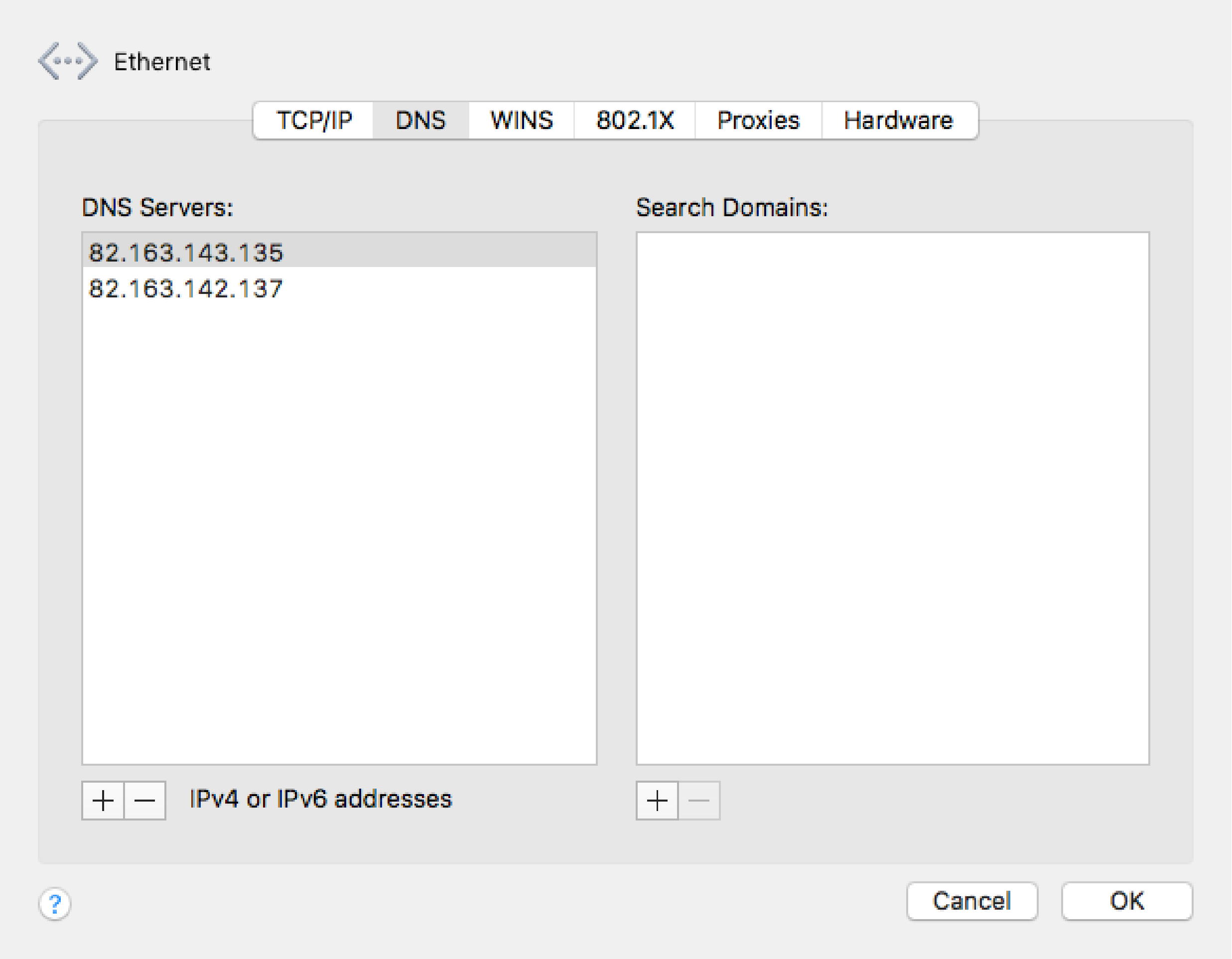
Country IL
State/Province Gush Dan
Locality Hertzilia
Organization GreenTeam Internet, Ltd.
Organizational Unit Web
Common Name cloudguard.me

Serial Number 00 B6 F1 AB F3 8B 9A B4 1A

Q: How do I disinfect myself?

A: Often malware can install other malware, or allow an remote attacker to do what ever they want. Thus if you were/are infected it's suggested you fully re-install macOS. However, you can probably get away with simply resetting the DNS servers and deleting the malicious certifcate.

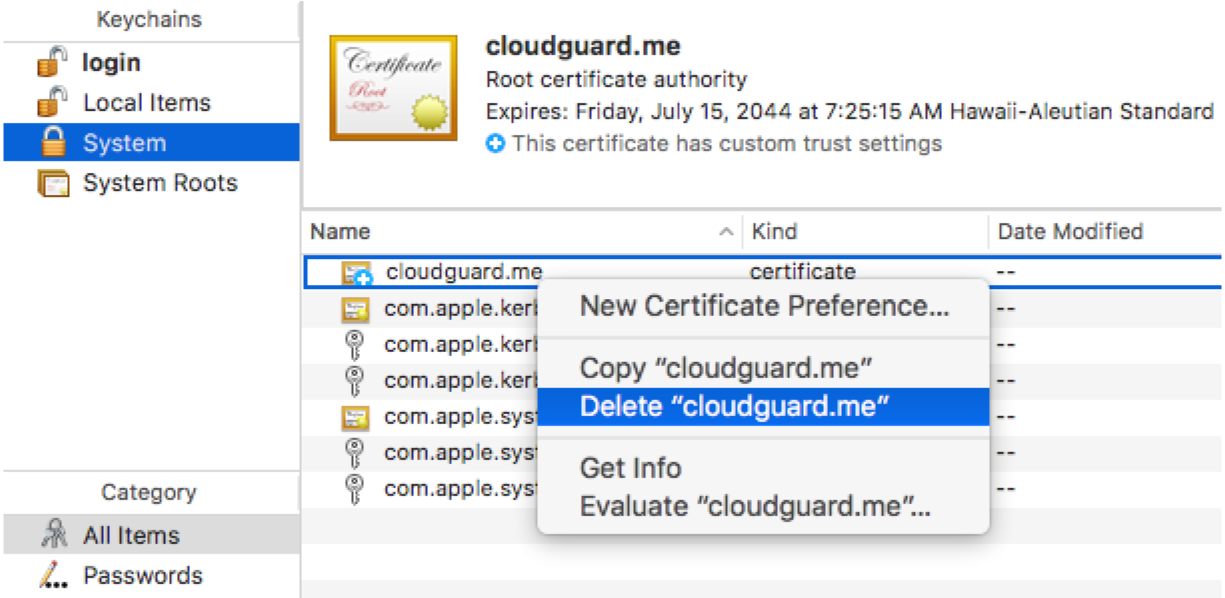
- Remove DNS Servers:
  Open the System Preferences Application, click the 'Network' Icon, then the 'Advanced' button, and finally the 'DNS' button. If infected, you'll see the malicous DNS servers (82.163.143.135 and 82.163.142.137):



DNS Servers:
82.163.143.135
82.163.142.137
+ − IPv4 or IPv6 addresses

Selected each server, then click the '-' button to delete.

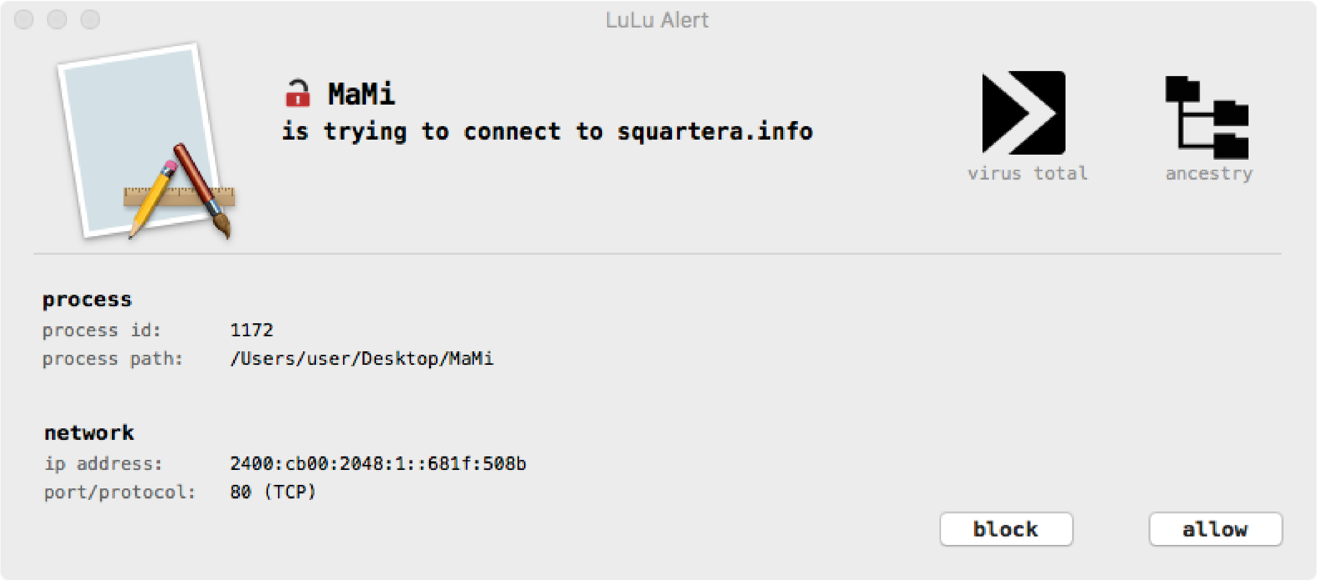
- Remove Certificate:
  Open the Keychain Access Application, click on 'System' in the Keychains (top left). If infected you'll see the malicious certificate (cloudguard.me). Right click on the certifitcate and select 'Delete' to remove it:



**cloudguard.me**
Root certificate authority
Expires: Friday, July 15, 2044 at 7:25:15 AM Hawaii-Aleutian Standard
This certificate has custom trust settings

| Name | Kind | Date Modified |
|---|---|---|
| cloudguard.me | certificate | -- |
| com.apple.ker | | -- |
| com.apple.ker | | -- |
| com.apple.ker | | -- |
| com.apple.sys | | -- |
| com.apple.sys | | -- |
| com.apple.sys | | -- |

Q: Will my AV product protect me?
A: Evenutally. But for now, it does not appear that any will. I'd recomment a 3rd-party tool such as firewall that can detect & block outgoing traffic. I'm currently working on a free open-source firewall named 'LuLu' that will detect OSX/MaMi's network traffic:



LuLu Alert

**MaMi**
**is trying to connect to squartera.info**

```
process
process id:      1172
process path:    /Users/user/Desktop/MaMi

network
ip address:      2400:cb00:2048:1::681f:508b
port/protocol:   80 (TCP)
```

Q: Did I discover this malware?
A: No, a good friend brought it to my attention. I just happen to blog about things such as macOS malware!

Q: Why the name, OSX/MaMi
A: Since there are already several (IMHO unrelated) malware specimens that perform DNS hijackering (that are named 'DNSChanger', etc), I decided to call is OSX/MaMi due to a core class the malware named:

'SB**MaMi**Settings'

love these blog posts & tools? you can support them via patreon! Mahalo :)