The DGA of Qadars v3

Archived: 2026-04-06 15:44:08 UTC

In March, the following sample caught my attention because it relies on a Domain Generation Algorithm (DGA) to communicate with its C&C-servers:

md5
: 0dcbb31cbc5279293cb5ebf4cd9eff4e

sha256
: 43199706ffdfab175fc27dfa8d739e842541e0c285f8f4c0b937e82345ae6362

source
: malwr

For example, on April 12th the first 10 generated domains are:

- nctqrgta7o52.net
- khqzwlib0pun.org
- lun0lmr4xe7k.com
- qvc9inkxy3o9.com
- z8daz8ti3wxy.net
- k1ar8levktun.net
- 1ij05qzgta70.org
- 6bwdyvw5if45.com
- jwde7stqb0da.org
- 05azc52rs1yb.org

What is it?

The Virustotal scan invoked by the malwr analysis gives a "*clean*" rating throughout the board, except for *"Qihoo-360"* which at least triggers a generic *"HEUR/QVM07.1.0000.Malware.Gen"* detection.

The significant strings of the malware are all encrypted with a 10 byte XOR key (`FC 57 91 BC 75 9A 12 CC A4 26`). You can see the full list of plaintext strings here, among them are:

- `klpszVersion`
- `gBitness`
- `kdwTimestamp dData fLength flpData@`
- `hmainType.gsubType`

These strings are symptomatic of the banking trojan **Qadars**, as can be seen in a report by Security Intelligence.

Qadars' binary contains a hard-coded version string. For my sample, the version string is **3.0.0.0**:

The Security Intelligence report is on Qadars 2.0.0.0 and does not mention a domain generation algorithm, so the feature was likely introduced with Qadars version 3.

Disassembly of the DGA

The next listing shows the disassembly of the domain generation algorithm. Skip to the next Section for a run-down of the properties of the DGA.

```
 .text:004095F0 ; BOOL __cdecl dga(void *pDomain, size_t sld_len)
.text:004095F0 dga             proc near               ; CODE XREF: sub_409FD0+20p
.text:004095F0
.text:004095F0 charset         = byte ptr -38h
.text:004095F0 tlds            = dword ptr -10h
.text:004095F0 pDomain         = dword ptr  8
.text:004095F0 sld_len         = dword ptr  0Ch
.text:004095F0
.text:004095F0                 push    ebp
.text:004095F1                 mov     ebp, esp
.text:004095F3                 sub     esp, 38h
.text:004095F6                 push    ebx
.text:004095F7                 push    esi
.text:004095F8                 mov     [ebp+tlds], offset a_com ; ".com"
.text:004095FF                 mov     [ebp+tlds+4], offset a_org ; ".org"
.text:00409606                 mov     [ebp+tlds+8], offset a_net ; ".net"
.text:0040960D                 push    edi
.text:0040960E                 mov     edi, edi
.text:00409610
.text:00409610 loc_409610:                             ; CODE XREF: dga+166j
.text:00409610                 mov     ebx, [ebp+pDomain]
.text:00409613                 mov     ecx, 9
.text:00409618                 mov     esi, offset charset ; "abcdefghijklmnopqrstuvwxyz0123456789"
.text:0040961D                 lea     edi, [ebp+charset]
.text:00409620                 rep movsd
.text:00409622                 movsb
.text:00409623                 test    ebx, ebx
.text:00409625                 jz      loc_409740
.text:0040962B                 mov     edi, [ebp+sld_len]
.text:0040962E                 cmp     edi, 5
.text:00409631                 jbe     loc_409740
.text:00409637                 cmp     domain_nr, 0
.text:0040963E                 jnz     short loc_40966D
.text:00409640                 push    0
```

```
.text:00409642                call    ds:_time64
.text:00409648                add     esp, 4
.text:0040964B                push    0
.text:0040964D                mov     esi, eax
.text:0040964F                sub     eax, 345600
.text:00409654                push    604800
.text:00409659                sbb     edx, 0
.text:0040965C                push    edx
.text:0040965D                push    eax
.text:0040965E                call    _allrem
.text:00409663                sub     esi, eax
.text:00409665                and     esi, 7FFFFFFFh
.text:0040966B                jmp     short loc_409673
.text:0040966D ; ---------------------------------------------------------------------------
.text:0040966D
.text:0040966D loc_40966D:                            ; CODE XREF: dga+4Ej
.text:0040966D                mov     esi, r
.text:00409673
.text:00409673 loc_409673:                            ; CODE XREF: dga+7Bj
.text:00409673                push    edi             ; size_t
.text:00409674                push    0               ; int
.text:00409676                push    ebx             ; void *
.text:00409677                call    memset
.text:0040967C                lea     eax, [ebp+charset]
.text:0040967F                add     esp, 0Ch
.text:00409682                lea     edx, [eax+1]
.text:00409685
.text:00409685 loc_409685:                            ; CODE XREF: dga+9Aj
.text:00409685                mov     cl, [eax]
.text:00409687                inc     eax
.text:00409688                test    cl, cl
.text:0040968A                jnz     short loc_409685
.text:0040968C                sub     eax, edx
.text:0040968E                xor     ecx, ecx
.text:00409690                add     edi, 0FFFFFFFBh
.text:00409693                mov     ebx, eax
.text:00409695                jz      short loc_4096CB
.text:00409697                jmp     short loc_4096A0
.text:00409697 ; ---------------------------------------------------------------------------
.text:00409699                align 10h
.text:004096A0
.text:004096A0 loc_4096A0:                            ; CODE XREF: dga+A7j
.text:004096A0                                        ; dga+D9j
.text:004096A0                imul    esi, 3E39B193h
.text:004096A6                mov     edx, 89F5h
.text:004096AB                sub     edx, esi
.text:004096AD                and     edx, 7FFFFFFFh
```

```
.text:004096B3                 mov     esi, edx
.text:004096B5                 xor     edx, edx
.text:004096B7                 mov     eax, esi
.text:004096B9                 div     ebx
.text:004096BB                 inc     ecx
.text:004096BC                 mov     al, [ebp+edx+charset]
.text:004096C0                 mov     edx, [ebp+pDomain]
.text:004096C3                 mov     [ecx+edx-1], al
.text:004096C7                 cmp     ecx, edi
.text:004096C9                 jb      short loc_4096A0
.text:004096CB
.text:004096CB loc_4096CB:                             ; CODE XREF: dga+A5j
.text:004096CB                 imul    esi, 3E39B193h
.text:004096D1                 mov     ecx, 89F5h
.text:004096D6                 sub     ecx, esi
.text:004096D8                 and     ecx, 7FFFFFFFh
.text:004096DE                 mov     eax, 55555556h
.text:004096E3                 imul    ecx
.text:004096E5                 mov     eax, edx
.text:004096E7                 shr     eax, 1Fh
.text:004096EA                 add     eax, edx
.text:004096EC                 lea     eax, [eax+eax*2]
.text:004096EF                 mov     r, ecx
.text:004096F5                 sub     ecx, eax
.text:004096F7                 mov     ecx, [ebp+ecx*4+tlds]
.text:004096FB                 mov     eax, ecx
.text:004096FD                 lea     ecx, [ecx+0]
.text:00409700
.text:00409700 loc_409700:                             ; CODE XREF: dga+115j
.text:00409700                 mov     dl, [ecx]
.text:00409702                 inc     ecx
.text:00409703                 test    dl, dl
.text:00409705                 jnz     short loc_409700
.text:00409707                 mov     edi, [ebp+pDomain]
.text:0040970A                 sub     ecx, eax
.text:0040970C                 mov     edx, ecx
.text:0040970E                 dec     edi
.text:0040970F                 nop
.text:00409710
.text:00409710 loc_409710:                             ; CODE XREF: dga+126j
.text:00409710                 mov     cl, [edi+1]
.text:00409713                 inc     edi
.text:00409714                 test    cl, cl
.text:00409716                 jnz     short loc_409710
.text:00409718                 mov     ecx, edx
.text:0040971A                 shr     ecx, 2
.text:0040971D                 mov     esi, eax
```

```
.text:0040971F                mov     eax, domain_nr
.text:00409724                rep movsd
.text:00409726                mov     ecx, edx
.text:00409728                and     ecx, 3
.text:0040972B                rep movsb
.text:0040972D                inc     eax
.text:0040972E                xor     edx, edx
.text:00409730                mov     ecx, 0C8h
.text:00409735                div     ecx
.text:00409737                mov     ebx, [ebp+pDomain]
.text:0040973A                mov     domain_nr, edx
.text:00409740
.text:00409740 loc_409740:                            ; CODE XREF: dga+35j
.text:00409740                                        ; dga+41j
.text:00409740                push    ebx
.text:00409741                call    gethostbyname   ; ws2_32.gethostbyname
.text:00409747                neg     eax
.text:00409749                sbb     eax, eax
.text:0040974B                neg     eax
.text:0040974D                jnz     short loc_40976B
.text:0040974F                cmp     domain_nr, 0
.text:00409756                jnz     loc_409610
.text:0040975C                mov     edx, [ebp+sld_len]
.text:0040975F                push    edx             ; size_t
.text:00409760                push    0               ; int
.text:00409762                push    ebx             ; void *
.text:00409763                call    memset
.text:00409768                add     esp, 0Ch
.text:0040976B
.text:0040976B loc_40976B:                            ; CODE XREF: dga+15Dj
.text:0040976B                xor     eax, eax
.text:0040976D                cmp     domain_nr, eax
.text:00409773                pop     edi
.text:00409774                pop     esi
.text:00409775                setnz   al
.text:00409778                pop     ebx
.text:00409779                mov     esp, ebp
.text:0040977B                pop     ebp
.text:0040977C                retn
.text:0040977C dga            endp
.text:0040977C
.text:0040977C ; ---------------------------------------------------------------------------
```

Properties of the DGA

The DGA of Qadars produces up to 200 different domains which are tested with `gethostbyname`. If all 200 domains fail to resolve, then Qadar sleeps 20 seconds (not part of the previous disassembly) and starts over with the first domain.

The DGA uses a linear congruential generator as pseudo random number generator. The multiplier and increment are uncommon:

$$r \leftarrow (35317 - 1043968403 \cdot r) \mod 2147483647$$

The random number generator is seeded with the current date:

```
 .text:00409640                push    0
.text:00409642                call    ds:_time64
.text:00409648                add     esp, 4
.text:0040964B                push    0
.text:0040964D                mov     esi, eax
.text:0040964F                sub     eax, 345600
.text:00409654                push    604800
.text:00409659                sbb     edx, 0
.text:0040965C                push    edx
.text:0040965D                push    eax
.text:0040965E                call    _allrem
.text:00409663                sub     esi, eax
.text:00409665                and     esi, 7FFFFFFFh
```

The above disassembly boils down to

$$r = u - ((u - 4 \cdot 24 \cdot 3600) \mod 7 \cdot 24 \cdot 3600)$$

where u is the current unix timestamp. The assignment can be rewritten as:

$$r = \left\lfloor \frac{u}{7 \cdot 24 \cdot 3600} \right\rfloor \cdot 7 \cdot 24 \cdot 3600 + 4 \cdot 24 \cdot 3600$$

The calculation results in a different value every week on Thursday at midnight, as 1 January 1970 was a Thursday. All Qadars v3 samples will generate the same domains because there are no magic numbers involved in seeding the random number generator.

The DGA uses three hard-coded top level domains: *.com*, *.org* and *.net*. The second level domains consist of 12 random characters, picked from the lower case letters and the digits.

In summary, these are the properties of Qadars' DGA:

property	**value**
type	TDD (time-dependent deterministic)
generation scheme	Linear Congruential Generator

property	value
seed	current date
domain change frequency	7 days
domains per day	200
sequence	sequential
wait time between domains	20 seconds after 200 domains, none otherwise
top level domains	.com, .org, .net
second level characters	lower case letters and digits
second level domain length	12

Reimplementation of the DGA

The next code listing shows a reimplementation of Qadars' DGA in Python. You also find this reimplementation — along with other DGAs — in my [my Github repository](#).

```
 import argparse
import time
from datetime import datetime
import time
import string

def rand(r, seed):
    return  (seed - 1043968403*r) & 0x7FFFFFFF

def dga(date, seed):
    charset = string.ascii_lowercase + string.digits
    tlds = [".net", ".org", ".top"]
    unix = int(time.mktime(date.timetuple()))
    b = 7*24*3600
    c = 4*24*3600
    r = ((unix//b)*b + c)
    for i in range(200):
        domain = ""
        for _ in range(12):
            r = rand(r, seed)
            domain += charset[r % len(charset)]
        r = rand(r, seed)
        tld = tlds[r % 3]
        domain += tld
        print(domain)
```

```
if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument("-d", "--date",
             help="date for which to generate domains")
    parser.add_argument("-s", "--seed",
             help="seed as hexstring", choices={"89f5", "4449"},
             default="e08a")
    args = parser.parse_args()

    if args.date:
        d = datetime.strptime(args.date, "%Y-%m-%d")
    else:
        d = datetime.now()
    dga(d, int(args.seed,16))
```

Source: https://www.johannesbader.ch/2016/04/the-dga-of-qadars/