


# Bedep's DGA: Trading Foreign Exchange for Malware Domains

---

 [web.archive.org/web/20150524032716/http://asert.arbornetworks.com/bedeps-dga-trading-foreign-exchange-for-malware-domains/](http://web.archive.org/web/20150524032716/http://asert.arbornetworks.com/bedeps-dga-trading-foreign-exchange-for-malware-domains/)

Dennis Schwarz

April 21, 2015

As initially researched by Trend Micro [1] [2], Zscaler [1] [2], Cyphort, and Malware don't need Coffee, the Bedep malware family focuses on ad / click fraud and the downloading of additional malware. ASERT's first sample dates from September 22, 2014, which is in line with when Trend Micro started seeing it in their telemetry. In early 2015, the family got some more attention when it was being observed as the malware payload for some instances of the Angler exploit kit, leveraging the Adobe Flash Player exploit (CVE-2015-0311) which at the time was a 0day. It was also observed that this newer version was using a domain generation algorithm (DGA) to generate its command and control (C2) domain names.

This post provides some additional notes on the DGA including a proof of concept Python implementation, a look at the two most recent sets of DGA generated domains, and concludes with some sinkhole data.

## Samples

The following Bedep samples were used for this research:

- MD5 e5e72baff4fab6ea6a1fcac467dc4351
- MD5 1b84a502034f7422e40944b1a3d71f29

The former was originally sourced from KernelMode.

## Algorithm

I've posted a proof of concept (read: works for me) Python implementation of the DGA to ASERT's Github.

At the time of writing, I'm aware of two DGA configs. Each config contains three constants and a table of magical dwords used throughout the algorithm. The screenshot below highlights the table from the first sample:

```

IDA View-A Pseudocode-B Pseudocode-A Hex V
1 int __stdcall transform2_sub_1000FF79(unsigned int
2 {
3     int v5; // edx@1
4     int v6; // eax@2
5     int v7; // edi@2
6     unsigned int v8; // ecx@2
7     unsigned int v9; // eax@3
8     int v10; // ebx@6
9     int *v11; // ebx@10
10    int v12; // ecx@10
11    int v13; // edi@10
12    unsigned int v14; // eax@10
13    int v15; // ecx@10
14    unsigned int v16; // edx@11
15    int v17; // eax@12
16    int v18; // eax@13
17    int v20; // [sp+10h] [bp-564h]@6
18    int v21[335]; // [sp+14h] [bp-560h]@6
19    int v22[6]; // [sp+550h] [bp-24h]@11
20    int v23; // [sp+568h] [bp-Ch]@9
21    unsigned int v24; // [sp+56Ch] [bp-8h]@1
22    unsigned int v25; // [sp+580h] [bp+Ch]@10
23    int v26; // [sp+584h] [bp+10h]@10
24
25    v5 = 0;
26    v24 = 0;
27    do
28    {
29        v6 = *(6*transform2_Table_off_1001B230 + v5);
30        v7 = v6 + 8;
31        v8 = 0x281 * *(v6 + 4) ^ 0xD666E1F3;
32        do
33        {
34            v9 = (0x663D81 * *v7 ^ 0xD666E1F3) - v5;
35            v7 += 8;

```

Bedep's DGA starts by downloading an XML file from:

<http://www.earthtools.org/timezone/0/0>

This legitimate web service provides the time zone and local time at latitude zero and longitude zero. The <utctime> timestamp is parsed out and converted to milliseconds since year zero (0000-00-00). Then, 1-3 days are subtracted from it (depending on tick count timing—this feels like an anti-analysis technique) and it is converted to days since year zero. This value will be used in the next step.

Next, Bedep downloads an XML file from:

<http://www.ecb.europa.eu/stats/eurofxref/eurofxref-hist-90d.xml>

This legitimate file from the European Central Bank (ECB) contains the last 90 days of “Euro foreign exchange reference rates” and is updated daily. Each date is extracted from the <Cube time=”...”> tags then the days since year zero is calculated for “date minus one”. If the days since value is less than or equal to the value calculated in the first step AND if it falls on a Monday, then the foreign exchange reference rates for “date” are extracted and used. Here’s a visual showing this process:

```

three days ago: 0xb3b42 (736066)
downloaded currency xml, 71010 bytes
parsed 62 dates from currency xml
first date: 2015-04-15
finding correct days since:
trying: 0xb3b44 (736066) nope
trying: 0xb3b43 (736067) nope
trying: 0xb3b42 (736066) nope
trying: 0xb3b3f (736063) nope
trying: 0xb3b3e (736062) nope
trying: 0xb3b3d (736061) nope
trying: 0xb3b3c (736060) found
parsed 31 currencies from 2015-04-07 (currency date):
USD: 1.0047
JPY: 130.33
BGN: 1.9558
CZK: 27.455
DKK: 7.4714
GBP: 0.7286
HUF: 299.08
PLN: 4.0578
RON: 4.4165
SEK: 9.374
CHF: 1.0438
NOK: 0.73
HRK: 7.619
RUB: 59.6265
TRY: 2.0079
AUD: 1.4192
BRL: 3.3979
CAD: 1.3563
CNY: 6.7241
HKD: 0.4886
IDR: 14891.77
ILS: 4.267
INR: 67.598
KRW: 1183.28
MXN: 16.1919
MYR: 3.9527
NZD: 1.4423
PHP: 48.3
SGD: 1.4724
THB: 35.335
ZAR: 12.8345

```

After testing, my analysis reveals that Bedep updates using “last Tuesday’s” foreign exchange reference rates—where “last Tuesday” refers to “the preceding week’s Tuesday” until “this week’s Thursday”. After this, it means “this week’s Tuesday.”

From here, the algorithm becomes a bit opaque. Various values such as “days since,” the first parsed currency’s abbreviation, the low dword of the first parsed currency’s rate, the magical dword values from the extracted table (noted above), and various other constant and calculated values are transformed a number of times. I wasn’t able to deduce the “big picture” of these transforms, so I’m treating them as a blackbox where the output is the number of domains to generate and three values that that will be used to calculate a modular exponent starting seed. If anyone has more details on this blackbox, please reach out!

The number of domains to generate is 22 for the first config and 28 for the second for a total of 50 domains per set. To generate each domain, the starting seed and foreign exchange reference rates are transformed a number of times to calculate the domain length and the domain characters themselves:



This info is inline with what Zscaler observed.

Using the foreign exchange rates from 2015-04-14, here are the domains registered out of the set, so far:

- prlvlpdeiopx.com (5.196.181.244)
- tqadnvxgppn1.com (5.196.181.244)
- gllmrtvteldx.com
- uydsqobdcmcxpdxng.com
- owwiloxvthttt1.com
- gcrnbjlgchu.com

The first two were registered on 2015-04-19, then 2015-04-17, and the last two on 2015-04-15. All six used the same registrant noted above.

## **Sinkhole**

To get a better idea of how active and widespread the above campaign is, we setup a sinkhole. The sinkhole was vmznlwrgtcnasmfhz.com and from 2015-04-13 13:47 UTC to 2015-04-16 17:06 UTC (about 3 days) it received phone homes from about 82,127 unique source IPs. The top 10 TLDs of the resolved source IPs were:

1. net (31578)
2. com (11952)
3. de (3193)
4. mx (2611)
5. tr (2104)
6. it (1521)
7. pl (1500)
8. fr (1440)
9. br (1360)
10. au (1247)
11. ca (1107)
12. jp (1054)
13. es (769)

And, except for Russia, infections were all over the map:



## Conclusion

This post has taken a closer look at Bedep's DGA and the recent campaign around it. Compared to some of the other date based DGAs we've looked at in the past, this algorithm is quite a bit more complicated and involved—effectively relying on the foreign exchange markets to generate its C2 domains. Based on the domain registration and sinkhole activity, Bedep is a current and active threat and will likely remain so for the foreseeable future.