Vol.2018-IOT-42 No.4
2018/6/28

悪性 Botnet 包囲網における DGA 検知の試み

村上順也[†1]　山之上卓[†1]

概要：現在開発中の悪性 Botnet 包囲網で Domain Generate Algorithm (DGA)を利用する Bot の DGA 利用を検知する試みについてのべる．悪性 Botnet 包囲網は NAT ルータやルータとその配下の LAN の間に設置する Agent Bot と,Agent Bot によって獲得されたデータを解析する Analyzing Bot によって構成されている．Agent Bot も Analyzing Bot も Wiki ページに書かれた script によって制御されている．Analyzing Bot は統計計算パッケージ R を備えており,それを操作するスクリプトに R 言語で解析処理を書くことができる．Agent Bot で,担当する LAN 内のホストの DNS へのアクセス状況を獲得し,それを Analyzing Bot の R で解析することで,DGA 利用の特定ができるのではないかと仮定し,その可能性の検証を行う．

キーワード：Botnet, Bot, Wiki, DGA, IDS, 分散協調,

An Attempt to Detect DGA by the Malicious Botnet Capturing Network

TAKASHI YAMANOUE[†1]

Abstract: A malicious botnet capturing network (beneficial botnet), which tries to cope with malicious botnets with Domain Generation Algorithm(DGA), is discussed. In order to cope with such botnets' technology, we are developing a beneficial botnet as an anti-bot measure, using our previous beneficial bot. The beneficial botnet is a group of beneficial bots which are Agent bots and an Analyzing bot. A malicious botnet with DGA is hard to detect by a single Intrusion Detection System (IDS). Our beneficial botnet has the ability to detect DGA, using collaboration of our beneficial bots. The beneficial bot could detect communication of the pseudo botnet which mimics malicious botnet communication.

Keywords: Botnet, Bot, Wiki, DGA, IDS, Distributed, collaboration

1. はじめに

多くのネットワーク管理者やセキュリティ担当者が悪性 Botnet に頭を抱えている．悪性 Botnet は多くの人々に様々な場所から迷惑メールをばらまいたり，その超並列特性を使って，多くのパケットを短時間に 1 台のサーバに送信することにより，DDoS 攻撃を行ったりする．Botnet の Bot はゾンビコンピュータの利用者の銀行口座を盗むことも行う．Botnet は粘り強い性質も持つ．情報セキュリティ担当者が Botnet の中のいくつかの Bot を見つけて対処しても，Botnet は悪事を続けることができる．悪性 Botnet は情報セキュリティ担当者の bot 対策を潜り抜けるため，日々進化している．

2000 年代中頃に現れた Agobot/Phatbot[6]などの Botnet には bot 対策を潜り抜けるため Peer To Peer (P2P)ネットワークの技術を使っている．2000 年代後半に現れた conficker[2]などの Botnet はドメイン生成アルゴリズム(Domain Generation Algorithm, DGA)の技術を使っている．

Gameover ZeuS は有名な botnet である．これは警察組織の国際的な連携により，2014 年に壊滅したが，FBI の公表[4]によると，損失は 1 億ドルに上ると見積もられている．

組織の出入り口に設置された 1 台の侵入検知システム(Intrusion Detection System, IDS) で組織内のボットの P2P 通信を検知することは難しい．Gameover Zeus のように動的に Command and Control(C2) サーバのドメインを探索できるよう作られた DGA を取り入れたマルウェアも増加している．

このような Botnet の技術に対処するため，我々が従来から開発を続けていた良性 Bot[10][11][12][13][14] を使って，悪性 Botnet 包囲網(良性 Botnet) を開発している．良性 Botnet は LAN の Nat の内側に設置する Agent Bot と，Agent Bot によって選択収集されたデータを解析する Analyzing Bot により構成された，良性 Bot のグループである．この包囲網の評価を行うため，悪性 botnet の振る舞いをまねる，偽 Gameover Zeus の開発も行っている．

本論文では DNS への問い合わせのデータを Agent bot により収集し，Analyzing bot でそのデータを解析することにより，DGA を使っている可能性の高いボットの検知を行なおうとしていることと，偽 Gameover Zeus を使った DGA 検知機能の検証しようとしていることについて述べる．

†1 福山大学
Fukuyama University

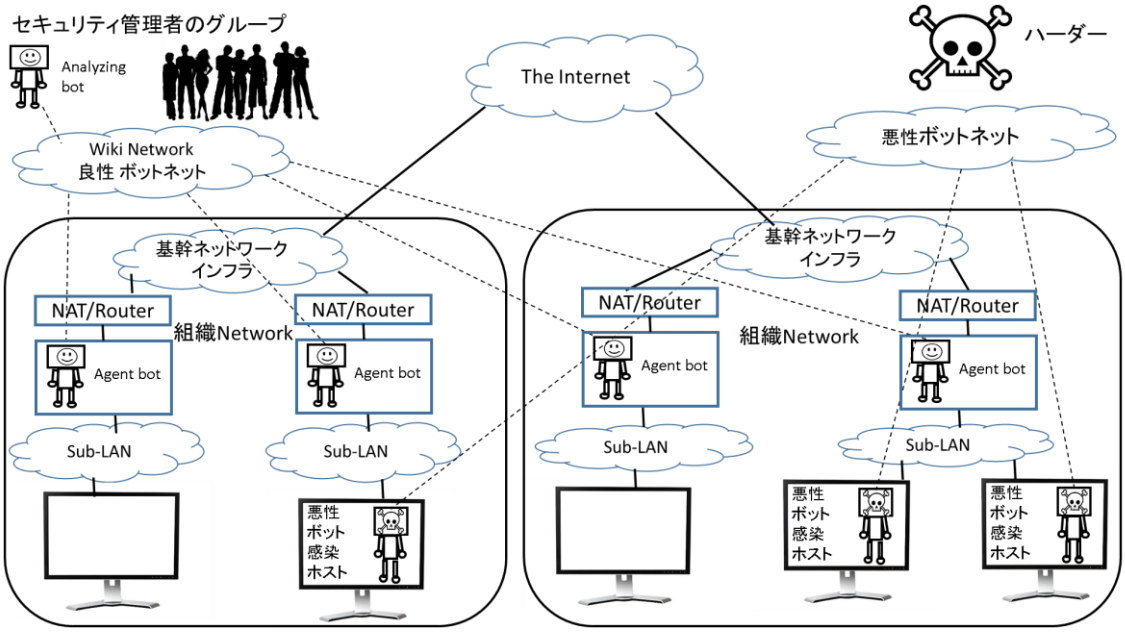

図 1　悪性 Botnet 包囲網(beneficial botnet)の概要

Figure 1　Outline of the beneficial botnet

2. 悪性 Botnet 包囲網

組織のネットワークの出入り口に IDS や IPS を設置して利用することが良く行われている．C2 サーバとすべての bot の間の通信がそれによって検知可能であるので，組織の出入り口に設置された IDS や IPS は中央集中的な C2 サーバを持つ悪性 Botnet には有効である．

しかしながら，DGA や P2P を使った中央集中的な C2 サーバを持たない悪性 Botnet の Bot を特定することは，このような IDS や IPS では困難である．我々は，Gameover ZeuS のような P2P や DGA 機能を持つ botnet に対処する悪性 botnet 包囲網(beneficial botnet)を設計している．

2.1 悪性 botnet 包囲網の概要

この論文で述べる悪性 botnet 包囲網(beneficial botnet) は，従来の IDS や IPS の欠点を克服しようとするものである．図 1 に悪性 botnet 包囲網の概要を示す．Agent bot は Sub-LAN と NAT または Router の間に設置される．Agent bot は Sub-LAN と Sub-LAN の外部との通信データを収集する．Analyzing bot は Agent bot が収集したデータを集めて解析し，sub-LAN 内に潜む bot を検出する．すべての Agent bot と Analyzing bot はインターネット上の Wiki のページに書かれた script によって制御される．

2.2 悪性 botnet 包囲網の bot

Beneficial botnet で利用する beneficial bot は wiki ソフトウェアの wiki ページ上に書かれたスクリプトのインタープリタである.図 2 に beneficial bot の振る舞いを示す．これは以下を繰り返す．

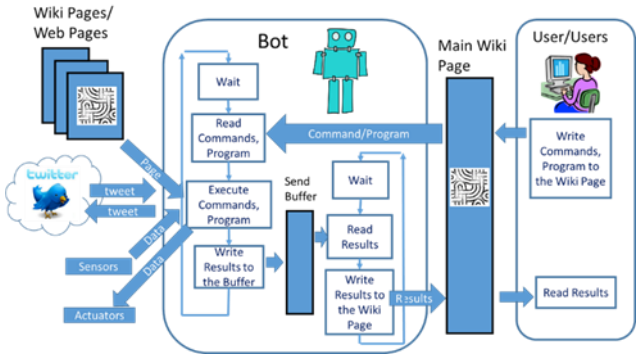

図 2　Bot の振る舞い

Figure 2　Behavior of a bot

1) 一定時間停止する．
2) コマンドとプログラムで構成されたスクリプトを，事前に bot に指示されていた wiki ページ(main wiki)から読み込む．
3) コマンドとプログラムを実行する．プログラムは main wiki の他に，他の wiki ページや web ページを読み込むことができる．もし，bot が agent bot であるなら，コマンドによって，bot が動いているコンピュータのネットワークインターフェース間の通信データを収集することができる．Agent bot は，インターフェース間の通信をコマンドによって制御することもできる．もし，bot が analyzing bot であるなら，R 統計計算システム[5]を用いて，収集されたデータを解析することができる．
4) Bot の実行結果は送信バッファに書きこまれる．送信バッファに書きこまれたデータは，スクリプトが書かれた wiki ページに書きこまれる．

図 3に script を書いた wiki ページの例を示す. result: の行より前の部分が script のコマンドとプログラムであり, result: より後の部分が実行結果である.
コマンドは command: で始まる行に書かれていて, プログラムは program: から始まる行に書かれている. この Script は, 1 から 10 までの和を計算し, その途中経過も含めて, result: より後に書きこんでいる.

```
objectPage http://www.[redacted]
device yamaRasPiDp9_1 or yamaRasPiDp9_2 start after no w
command: set readInterval=60000
command: set execInterval=0
command: clear sendBuffer;
command: program ex1
program: s=0;
program: for i=0 to 10
program:   s=s+i
program:   ex("service","putSendBuffer "+s)
program: next i
command: end ex1
command: run ex1
command: ex("service","sendResults.")
result:
0
1
3
6
10
15
21
28
36
45
55
currentDevice="yamaRasPiDp9_1",Date=2018/5/15/ 12:54:17
```

図 3　Bot が実行する Script の例
Figure 3　Example of the Script which is interpreted by a Bot

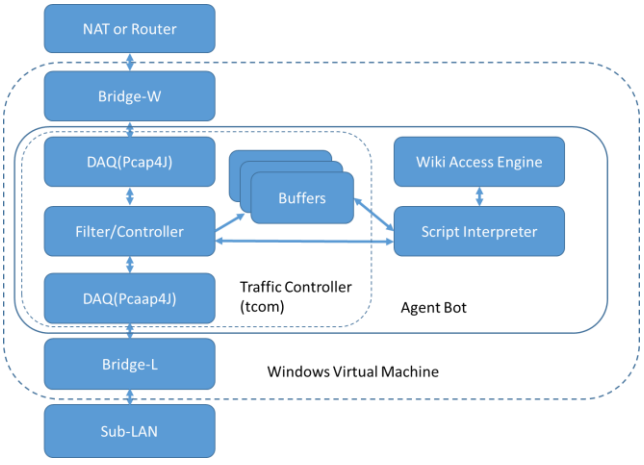

図 4　Agent Bot の構成
Figure 4　Structure of the Agent Bot

コマンドとして, “set pageName <page-name>” と “include <url>” も使うことができる. “set pageName <page-name>” が bot のインタープリタで解釈されたとき, bot は次に読み込む main wiki ページを, 元の main wiki ページと同じ wiki の, <page-name> で示されたページで置き換える. <page-name>の一部としてその時の時間や日付を使うこともできる. これにより, 日時に応じた別のページに実行結果を書きこむことが可能になる. “include <url>” が bot のインタープリタで解釈されたとき, bot はその部分に <url> で示された wiki に書かれている script を埋め込む. これにより, オブジェクト指向プログラミングにおいて複数のオブジェクトが 1 つの class で書かれたプログラムを実行するのと同様に, 複数の bot が同じ script を実行することが可能になる. Include される script を含んだページを class ページと呼ぶことにする. それに対して, main wiki のことを object ページとも呼ぶことにする.

Bot は http を使って wiki ページと通信しているため, Bot が NAT で守られた LAN の中にいた場合でも, main wiki のページをインターネットや Bot から見える場所にある Wiki のサーバに置くことにより, Bot の管理者は Bot を NAT の外部から制御することが可能になる.

2.3　Agent Bot

Agent bot は”Traffic controller” (tcon) を備えた良性 bot である.

Tcon は 2 つの network interfaces とデータ取得ライブラリ(DAQ)と Filter/Controller と複数の Buffer を持っている. DAQ として Pcap4j[7] を使っている.

この Bot の script interpreter は tcon の操作も行う. Bot は 2 つの interface の間の通信データを収集し, main wiki の scriptによってそのデータを main wiki のページに書きこむ. 2 つの interfaces 間の通信の制御することもできる. Network interface の一つを NAT または router に接続し, もう一つを sub-LAN に接続することにより, sub-LAN に接続されたホストと sub-LAN の外との間の通信のすべてを収集し, 制御することができる.図 4 に Agent bot の構造を示す.

tcon は以下の buffer を持つ. 通信データはその種類によって分類され, これらの buffer に格納される.

- Packet-history

この buffer は 2 つの interface 間で転送されるパケットの情報を格納する.この buffer は同じ source IP address と destination IP address のペア同士を格納するための sub buffer を持っている. Packet の payload の sha1 hash とその packet が転送された時間と packet の情報もペアと一緒に格納される. Sha1 の値は 2 つの異なる sub-LAN 間で行われる P2P 通信を検知するために使われる.

Sub buffer があふれた時, 古い情報が削除される.

- MAC-list

この buffer は sub-LAN 内でやりとりされるすべての frame の MAC address を格納する. MAC address に対応付けられたすべての IP address も Mac address に関連付け

て格納される. MAC list はこの sub-LAN に接続されているホストや, LAN 内の IP address を知るのに使うことができる. 他よりも多くの IP アドレスが結び付けられている MAC address を見つけることにより, default gateway を特定することもできる.

- Domain-list

この buffer は sub-LAN 内の DNS query の返答をその query が行われた時間と一緒に格納する. Domain-list は sub-LAN のどのホストが LAN 外のどのホストと通信したか知るために使うことができる. この list は DGA の利用を検知するためにも利用することができる.

- Dhcp-list

この buffer は DHCP 通信の結果をそれが行われた時間と共に格納する. この list は DHCP spoofing や不正 DHCP サーバの検知を行うのに使うことができる. Default gateway を特定するのにも役立つ.

- Arp-list

この buffer は Arp 通信をそれが行われた時間と共に格納する. Arp spoofing の検知に使うことができる.

Agent bot はこれらの buffer から情報を得たり, 操作したりするコマンドを実行できる. 2 つのネットワーク間の通信を制御するコマンドも持っていて, このコマンドを使うことで悪性 bot の通信を見つけた時, その通信を遮断することもできる.

2.4 Analyzing Bot

Analyzing bot は各 agent bot で収集された情報を集め, それを解析する.

すべての良性 bot の言語プロセッサは, comma separated value (CSV) parser と表操作/表計算関数を備えている.

Analyzing bot はこれに加えて R 統計計算システム[5]も備えている. Analyzing bot の script の中に R のプログラムを埋め込むことができる. 良性 bot が元々備えている言語プロセッサと R 統計計算システムの間で値を交換する関数も使うことができる.

3. 実験

我々は beneficial botnet を使って実験的な bot 検知基盤を実装し、偽 Gameover Zeus の DGA の機能をこの基盤の中で動作させて, beneficial botnet が DGA を検知できるかどうかの確認作業を行っている.

3.1 偽 Gameover ZeuS

Gameover ZeuS は C2 server と Web サーバである C2 Proxy と P2P ネットワークのノードである Harvester bot によって構成されている. Harvester bot の一部は proxy bot として, C2 Proxy と P2P ネットワークの間の通信を担う(図 5).

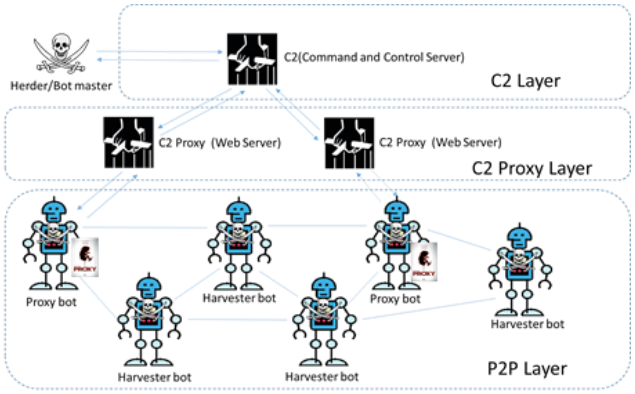

図 5 Gameover ZeuS の構成

Figure 5 Structure of the Gameover ZeuS

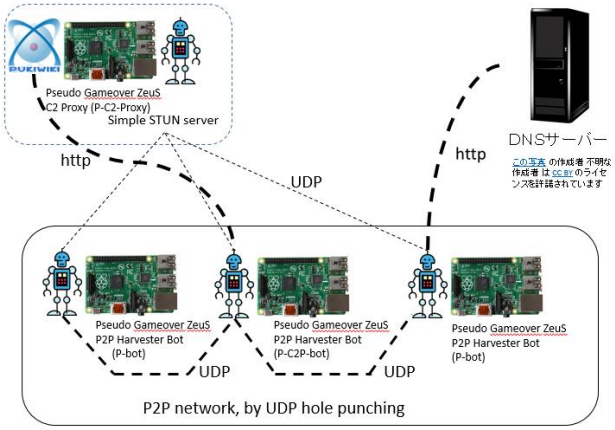

図 6 偽 Gameover ZeuS

Figure 6 Pseudo Gameover ZeuS

偽 Gameover ZeuS は, C2 Proxy に相当するホスト(P-C2-Proxy)の Wiki ページに書かれたコマンドを, Proxy bot に相当する P2P Harvester bot (P-C2P-bot)が読み込み, それを, P2P ネットワークを通じて, Harvester bot に相当する, 偽 Gameover ZeuS の他の P2P Harvester bot (P-bot)に転送する. P-bot はコマンドを読んでもなにも実行しない. 偽 Gameover ZeuS の P2P ネットワークは, UDP ホールパンチングを使って, bot 間を UDP 通信で結んでいる. UDP ホールパンチングを行う為, P-C2-Proxy で簡易 STUN サーバを動かしている. DGA の機能も持っている(図 6).

偽 Gameover ZeuS は良性 bot に P2P 通信の機能を加えるなどして作成した.

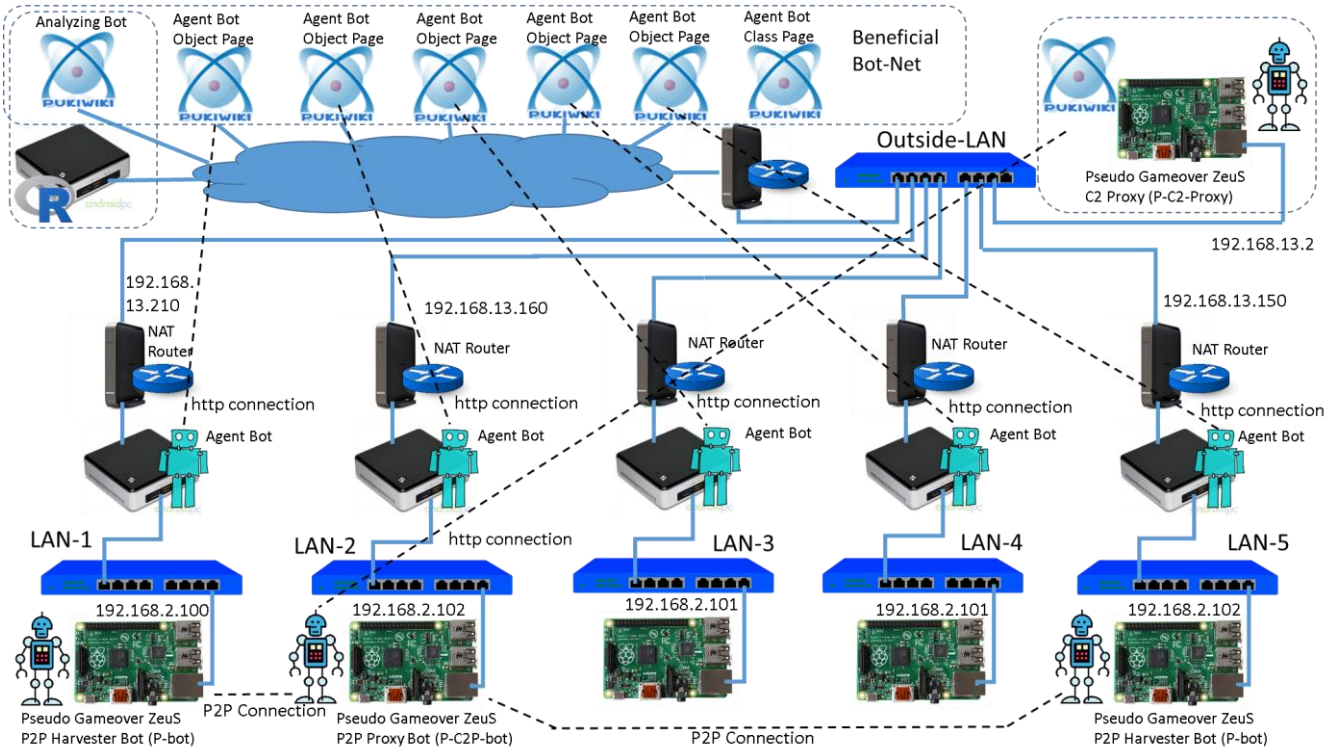

図 7　Bot 検知基盤に偽 Gameover ZeuS を配置した実験ネットワーク

Figure 7　Experimental network consists of the beneficial botnet and the Pseudo Gameover ZeuS

```
command: set readInterval=180000
command: set sendInterval=180000
command: set execInterval=0
command: tcon clear all.
command: set reportLength=600
command: clear sendBuffer.
#command: tcon get repeating unicast over 300000.
command: program ex1
program:  ex("tcon","set repeating number=20.")
program:  output10=ex("tcon","get repeating unicast over 60000.")
program:  output11=grep(output10,"prtcl=udp")
program:  output11=grepNot(output11,"dp=123,")
program:  output11=grepNot(output11,"sp=123,")
program:  output11=grepNot(output11,"dp=53,")
program:  output11=grepNot(output11,"sp=53,")
program:  ex("service","print In "+output11)
program:  ex("service","putSendBuffer "+output11)
program:  output20=ex("tcon", "get hosts.")
program:  ex("service","print In "+output20)
program:  ex("service","putSendBuffer "+output20)
program:  output30=ex("tcon", "get domain-list.")
program:  ex("service","print In "+output30)
program:  ex("service","putSendBuffer "+output30)
#command: tcon get dhcp-list.
#command: tcon get arp-list.
command: end ex1
command: clear sendBuffer
command: run ex1
command: sendResults.
```

図 8　すべての Agent bot に対する指示を行う Class page の Script

Figure 8　Script of the Class page to direct all of agent bots

偽 Gameover Zeus の DGA プログラムは,1 分ごとに 100 個のドメイン名を自動生成し,生成ごとに DNS に問い合わせに行き,C2 Proxy の IP アドレスを探索していく(図 6). DGA を使った通信では, DNS にランダムに見えるドメイン名を多数問い合わせする特徴がある.

3.2 実験ネットワーク

Beneficial botnet が DGA を使っている bot の検知を行うことができるか否かを知るために, 図 7 で示す実験のための bot 検知基盤に偽 Gameover ZeuS を配置したネットワークを構築し, 偽 Gameover ZeuS と beneficial botnet を動作させた.

3.3 Agent Bots の script と実行結果

図 8 に agent bot の script を示す. この script の中で, *program: output30=ex("tcon", "get domain-list.")* が tcon の domain-list を Object ページに書き込むことを表す.

Agent Bot が作成する domain-list の中の DNS への問い合わせは, DNS から正引きできないとき, 問い合わせるドメイン名の後ろに no_such_name がつけられて表示される.

```
cmd=get domain-list, date="2018/06/01 17:29:42 +0900", qhost="192.168.2.100", name=cs9.wac.phicdn.net, ip="117.18.237.29", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:20:05 +0900", qhost="192.168.2.100", name=googleapis.l.google.com, ip="216.58.197.10", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:23:26 +0900", qhost="192.168.2.102", name=0.debian.pool.ntp.org, ip="36.3.117.150", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:28:45 +0900", qhost="192.168.2.100", name=a-0019.a-msedge.net, ip="204.79.197.222", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:06:28 +0900", qhost="192.168.2.100", name=hk2-eap.settings.data.microsoft.com.akadns.net, ip="52.175.39.99", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:15:09 +0900", qhost="192.168.2.102", name=mirror1.malwaredomains.com, ip="74.63.222.170", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:42:07 +0900", qhost="192.168.2.100", name=array706-prod.do.dsp.mp.microsoft.com, ip="52.229.170.171", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:24:13 +0900", qhost="192.168.2.102", name=1.debian.pool.ntp.org, ip="160.16.75.242", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:09:23 +0900", qhost="192.168.2.102", name=www.google.com, ip="216.58.221.164", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:27:26 +0900", qhost="192.168.2.100", name=urlreputation-sg2p.smartscreen.microsoft.com, ip="40.65.178.165", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:21:06 +0900", qhost="192.168.2.101", name=onecollector.cloudapp.aria.akadns.net, ip="52.114.76.37", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:28:41 +0900", qhost="192.168.2.100", name=ocsp.globalsign.cloud, ip="104.18.25.243", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:20:01 +0900", qhost="192.168.2.100", name=dual-a-0001.a-msedge.net, ip="204.79.197.200", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:17:54 +0900", qhost="192.168.2.100", name=s-0001.s-msedge.net, ip="13.107.3.128", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:37:59 +0900", qhost="192.168.2.101", name=e10663.dscg.akamaiedge.net, ip="23.219.34.110", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:12:17 +0900", qhost="192.168.2.102", name=github.map.fastly.net, ip="151.101.0.133", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:19:36 +0900", qhost="192.168.2.100", name=iceotf-prod-fe-eastus.cloudapp.net, ip="40.114.54.223", server="192.168.2.1".
```

図 9　通常時の DNS サーバへの問い合わせ

Figure 9　DNS Queries of a LAN normal situation

```
cmd=get domain-list, date="2018/06/01 17:30:22 +0900", qhost="192.168.2.100", name=ssl.gstatic.com, ip="216.58.199.227", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:34:09 +0900", qhost="192.168.2.103", name=czobxmnmgaynbdylhyxahecm.com---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:36:28 +0900", qhost="192.168.2.103", name=jnzdinsjwlnkjljbzmfwpbe.biz.elecom---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:37:28 +0900", qhost="192.168.2.103", name=nffavgaulkxgxcygkroxmw.biz---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:39:32 +0900", qhost="192.168.2.103", name=ybrhvnwysgjidzbadpjdxg.biz.elecom---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:34:35 +0900", qhost="192.168.2.101", name=urlreputation-os1p.smartscreen.microsoft.com, ip="104.215.21.84", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:41:26 +0900", qhost="192.168.2.103", name=pgvukezcmgytlwcvzatxxgpdded.org---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:38:30 +0900", qhost="192.168.2.103", name=lgipffvdiktouudwtfcrguxrx.biz---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:42:24 +0900", qhost="192.168.2.101", name=hk2-eap.settings.data.microsoft.com.akadns.net, ip="52.175.39.99", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:02:27 +0900", qhost="192.168.2.101", name=e1898.dspg.akamaiedge.net, ip="23.211.12.114", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:39:55 +0900", qhost="192.168.2.103", name=pughuyljnxofzutmtkbvadahm.org.elecom---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:35:10 +0900", qhost="192.168.2.103", name=uvkhfztleahmzpbaibpgz.biz---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:34:50 +0900", qhost="192.168.2.103", name=qkfrayvxfmqwvtfaudurkofvxgmnj.com---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:00:53 +0900", qhost="192.168.2.100", name=www.google.com, ip="216.58.221.164", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:03:28 +0900", qhost="192.168.2.103", name=easylist-downloads.adblockplus.org.elecom---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:34:09 +0900", qhost="192.168.2.103", name=kfvkjfmfpojzphrkbugmsqpsy.ru.elecom---no_such_name, ip="*", server="192.168.2.1".
cmd=get domain-list, date="2018/06/01 17:43:39 +0900", qhost="192.168.2.103", name=fiasxivdaycagzearouoeabvbr.ru---no_such_name, ip="*", server="192.168.2.1".
```

図 10　DGA を使用した際の DNS サーバへの問い合わせ

Figure 10　DNS Queries of a LAN with a bot with DGA

図 9 に，DGA の機能を持った bot がいない LAN の domain-list，図 10 に，DGA の機能を持った bot が DGA の通信を行っているときの domain-list を示す．図 10 は図 9 よりも no_such_name がはるかに上回る数で確認できる．

このことから,急激な no_such_name の増加が窺えたら DGA を使用している可能性が十分にあると推測できる．

3.4 DGA を検知するための Analyzing Bot の script

Gameover Zeus は悪性 bot が p2p レイヤー内で感染端末とやり取りできない場合や一週間仲間のリストを更新できない場合,DGA を使用する．悪性 Bot が DGA を使うとき，定期的に,短時間に多くの不審な FQDN の問い合わせを DNS に行う[1]．その問い合わせの大半は失敗するので，Domain list は DGA を検知するにも役立つ場合があり得る．

このデータを，Analyzing Bot の R を使って統計解析し，DGA を使っているホストをある程度推定する部分を現在作成中である．

R による統計解析は以下のように行う．

1. Agent Bot に対して，Domain list をその Object ページに書き込むように指示を行う．
2. Analyzing Bot は定期的に Agent bot の object ページから，LAN 内のホストの DNS への問い合わせのうち，その問い合わせを行った日時(Date)，問い合わせを行ったホスト(qhost)，問い合わせる FQDN(name)，問い合わせの結果得られたその FQDN の IP アドレス(ip)を入手する．ここで，もしドメイン名を得ることができなかった場合は，Agent Bot により FQDN の後ろに”---no_such_name”が付けられ, ip が”*”になる．
3. Analyzing bot は LAN 内のそれぞれのホストの一定期間内の DNS への問い合わせについて, ip=”*”となる問い合わせ数を数え,そのホストの，一定期間内の問い合わせ全体のうち，どのくらい，問い合わせが失敗した割合があるかということを解析し，DNS に通信を行ったホストの IP アドレス,解析した日付,LAN の識別子を Analyzing Bot の object ページに書き込む．

3 の解析を行うプログラムはまだデバッグ中であるが、R 環境でプログラムを動かしてそれにテストデータを与える

ことにより，その動作を確認している．図 11 にデバッグ中の, DGA を検知する Analyzing Bot の Script の一部を示す．

R: で始まる行が R 統計計算システムのプログラムである．この中で，右端に::がある部分は，R の関数の定義が次の行に続いていることを表す．図 1 の中で定義されている R の関数の dgaList は，それぞれの Agent bot で収集された domain-list のデータについて，DNS への問い合わせを行ったホストの IP アドレスと，一定期間内にそのホストが DNS へ問い合わせを行った回数と，失敗した回数と，解析を行った時間のリストを作成する．

```
R: dgaList<-function(df){            ;;
R:       dgaXList <- as.numeric(NULL)   ;;
R:       dfy=split(df,df$QHOST)         ;;
R:       ly<-labels(dfy)                ;;
R:       dfx = split(df,df$NNX)         ;;
R:       dfx2<- dfx$'*'                 ;;
R:       dfx3<-split(dfx2,dfx2$QHOST)   ;;
R:       hn <- length(dfx3)             ;;
R:       lx <- labels(dfx3)             ;;
R:       for( i in 1:hn){               ;;
R:               dfxi<-dfx3[[lx[i]]]    ;;
R:               nnn<-nrow(dfxi)                    ;;
R:               nan<-nrow(dfy[[lx[i]]])            ;;
R:               z<-Sys.time()                      ;;
R:               xline<-paste("date=",z,            ;;
R:                  ", qhost=",lx[i],               ;;
R:                  ", number_of_no_such_name=",nnn,  ;;
R:                  ", all_request_number=",nan,    ;;
R:                  ", ratio=",nnn/nan)             ;;
R:               dgaXList <- append(dgaXList,xline) ;;
R:       }                                          ;;
R:       return (dgaXList)                          ;;
R:  }
```

図 11　デバッグ中の, DGA を検知する Analyzing Bot の Script の一部

Figure 11　A part of the Script of the Analyzing Bot

4.　4　関連研究

4.1 AAFID

Autonomous Agents for Intrusion Detection (AAFID)[9]は我々の良性 botnet と同様に分散 IDS の agent の集合である．このシステムは, agents と transceivers と monitors で構成されている．AAIF の agent と我々の agent bot は，どちらともコマンドによって制御され，通信データを収集する部分で類似している．Agent AAFID の agent は client host に install されているのに対して，我々の agent bot は LAN とそのルータ又は NAT ルータの間に設置される．我々の良性 botnet の管理者は agent bot を client host のそれぞれに install する必要はない．AAFID の monitor と transceiver は，双方とも agent などからデータを集めて，それを解析する部分で類似している．AAFID の monitor は wiki ページの script で制御されないのに対して，我々の agent bot や analyzing bot は wiki ページの script で制御される．Agent 間の通信方式については，AAFID については定義されていないのに対して，我々の botnet は wiki API を使っている．

4.2 Man in the Middle Attack

我々の agent bot は一種の man in the middle attack[3] を行っていると解釈することもできる．Agent bot によって，sub-LAN 内の多くの通信を制御可能である．我々は Agent bot が dark side に行かないように注意する必要がある．

4.3 DNS ログ解析

渡辺らは DNS サーバのクエリログの解析による DGA を用いたマルウェア検知のための予備調査を行った[15]．DGA より生成された FQDN を問い合わせる送信元は,定期的に短期間で大量の DNS サーバからの応答エラーがあるという特徴を検出した. DNS サーバのクエリログを用いて,その特徴を参考にネガティブキャッシュの変化をみて,マルウェアを検出するという手法である．我々の手法においては,それぞれの LAN 内のネットワークを監視して DNS のアクセス状況を Wiki ページに記録し,確認して検出する．

4.4 機械学習

津田は Domain Forest システムの機械学習によって bot の通信の様々な特徴を学習させ,学習によって得られた特徴を持つ botnet の通信を検出することにより, DGA を含めた botnet の通信の検出を行っている[16]．我々は過去のデータの蓄積を基にするのではなく、実際に DNS サーバへ通信が行われているパケットの問い合わせ失敗の割合を使って検知しているが，津田が使っているようなアルゴリズムを組み込むことでより正確な検出が可能になると思われる．

5　おわりに

現在試作を行っている悪性 botnet 包囲網(beneficial botnet)によって，マルウェアの DGA を使った通信を検出できる可能性があることを示した．我々の beneficial botnet は script を格納するための wiki ページとその script の interpreter から構成されている．評価実験を行う為，悪性 botnet の通信をまねる偽 Gameover ZeuS も作成した．

現時点で LAN-WAN 間通信が非常に遅いので実用的に使う為にはこの問題を改善しなければならない．セキュリティの強化とその検証も必要である．この他，利用しやす

くするために，デバッグの方法や環境についても改善する必要がある．

謝辞

本研究の一部は JSPS 科研費 16K00197 の助成を受けて実施しました．良性 botnet およびその開発時に利用した PukiWiki, Java, Pcap4J, Eclipse, Eclipse Egit, M2Eclipse, Apache, Apache http client, twitter4j, Raspberry Pi, Raspbian の開発者，実験の実施を手伝ってくれた学生諸君に感謝します．

参考文献

[1] Andriesse, D. and Bos, H. “An Analysis of the Zeus Peer-to-Peer Protocol”, Technical Report IR-CS-74, rev. April 10, 2014.
[2] Conficker, https://en.wikipedia.org/wiki/Conficker
[3] Conti, M., Dragoni, N. and Lesyk, V. 2016 “A Survey of Man In The Middle Attacks”, 2016, IEEE Communications Surveys & Tutorials, Vol. 18, Issue 3, IEEE, 2027-2051. DOI=10.1109/COMST.2016.2548426
[4] FBI. 2014. GameOver Zeus Botnet Disrupted, https://www.fbi.gov/news/stories/gameover-zeus-botnet-disrupted, June 2, 2014.
[5] Ihaka, R., and R. Gentleman. “R: a language for data analysis and grapics”. 1996, J. Comp. Graph. Stat. 5:299-314. Available via http://www.R-project.org.
[6] JPCERT コーディネーションセンター, “P2P 型ボット分析レポート”, 2007 https://www.jpcert.or.jp/research/2007/P2P_bot_analysis_report.pdf
[7] Pcap4J, https://www.pcap4j.org
[8] Puri, R. “Bots & Botnet: An Overview," 2003, SANS InfoSec Reading Room, http://www.sans.org/rr/whitepapers/malicious/
[9] Spafford, E. H. and Zamboni, D., “Intrusion detection using autonomous agents”, 2000, Elsevier, Computer Networks vol.34, pp.547-570.
[10] Yamanoue, T., Oda, K., Shimozono, K., “Capturing Malicious Bots using a Beneficial Bot and Wiki”, 2012, In Proceedings of the 40th annual ACM SIGUCCS conference on User services (Memphis, Tennessee, USA. 15-19 Oct. 2012). ACM, New York, NY, 91-96. DOI=https://doi.org/10.1145/2382456.2382477
[11] Takashi, Y., Kentaro, O., Koichi, S., “A Malicious Bot Capturing System using a Beneficial Bot and Wiki,” 2013, Journal of Information Processing (JIP), vol.21, No.2, pp.237-245.
[12] Yamanoue, T., Oda, K., Shimozono, K., 2013. “An Inter-Wiki Page Data Processor for a M2M System, “ 2013, In Proceedings of the 4th International Conference on E-Service and Knowledge Management (ESKM 2013), Advanced Applied Informatics (IIAIAAI), 2013 IIAI International Conference on.(Matsue, Shimane, Japan, 31 Aug- 4 Sep. 2013) IEEE, Los Alamitos, CA. 45-50. DOI= https://doi.org/10.1109/IIAI-AAI.2013.48
[13] Yamanoue, T., Oda, K., Shimozono, K., “Experimental Implementation of a M2M System Controlled by a Wiki Network,” 2014,In Applied Computing and Information Technology,Studies in Computational Intelligence,Springer, Vol.553,121-136.
[14] Yamanoue, T., “Monitoring Servers, With a Little Help from my Bots, “ 2017, In Proceedings of the 45th annual ACM SIGUCCS conference on User services (Seattle, Washington, USA. 01-04 Oct. 2017). ACM, New York, NY, 173-180. DOI=https://doi.org/10.1145/3123458.3123461
[15] 渡辺 拳竜, 池部 実, 吉田 和幸, “DNS ログ解析による DGA を用いたマルウェア検知のための予備調査” 2015, インターネットコンファレンス 2015 (IC2015) ポスター発表, 113-114.
[16] Wataru T., “A detection system which DNS traffic features to detect domains which are related to botnets”, 2015.