DDoS 攻撃模擬ツールの試作

2012SE104 川上　健人　2012SE134 黒田　涼介
指導教員：後藤　邦夫

1　はじめに

近年 IT 技術の発展により世界中で PC やスマートフォンなどの通信機器が広く普及している．誰でも簡単にインターネットを利用することができるため，コミュニケーションもインターネットで図っていることもある．誰でも使うことができる一方で，PC やスマートフォンのセキュリティや PC に対する知識が乏しい人がサイバー犯罪に巻き込まれることが増えてきた．様々なサイバー犯罪がある中で有効な対策が難しいとされる DDoS(Distributed Denial of Service) について研究する．DDoS 攻撃は攻撃元が複数あり，多くの場合 bot を使用したり，踏み台を使用するため攻撃元が特定できない．そのため有効な対策が無く脅威である．対策が 3 つ存在するが，軽減するだけであり，どれも有効な対策ではない．

本研究では IPv4，IPv6 において UDP Flood，ICMP Flood，TCP SYN Flood，TCP Connection Flood 攻撃を選んで実行できるよう C++ でひとつにまとめる．また，まとめたプログラムを GUI で実行可能にすることを目標とする．2014 年度大平，山下の卒業研究「IPv6 のための異常トラフィック生成プログラムの試作と評価」[3] ではこのプログラムの完成，さらに IPv6 に関係する異常トラフィックを用いた脅威を再現・試作するという課題が残っていた．先輩の研究に引き続き動作一般的なソケットでは無く raw ソケットを使い，事実上無制約なパケットを生成する．それを実際に送信することで負荷を与えて，通信を妨害する．

しかし，raw ソケットプログラミングにより自由度が向上する半面，普段意識することのないプロトコルの詳細まで設定しなければならない．そのため，raw ソケットプログラミングに慣れていない人にとっては実装が容易ではない．

DDoS 攻撃の対策を研究，テストをするためには模擬攻撃が必要である．外部に被害や迷惑をかけないために閉じたネットワークでする必要があるため，確認・模擬実験には，ネットワークエミュレータの CORE(CommonOpen Research Emulator)[2] を使用する．CORE を使用するにあたり実験環境には Linux10.04 を用いる．GUI を作成するために wxglade[1] を用いる．wxglade を使用するにあたり Linux14.04 を用いる．

なお，川上は主に GUI の作成，黒田は主にネットワーク構成を担当する．攻撃プログラムは二人で作成した．

2　DDoS 攻撃の概要

現在の DDoS 攻撃の種類や対策について説明する．

2.1　DDoS 攻撃の対策

DDoS 攻撃は攻撃元が特定できないため，有効な対策が無く脅威である．3 つの対策があるが，軽減はできるもののどれも有効ではない．1 つ目の対策は同じ IP アドレスからのアクセス回数を制限すること．攻撃者が複数の PC を操作し攻撃を仕掛ける DDoS 攻撃ではこの対策の効果は期待することができない．2 つ目の対策は web サイトのサービス対象者が国内だけであれば，海外からのアクセスを制限すること．しかし国内の PC を使って攻撃された場合は全く意味をなさない．3 つ目対策は大規模のアクセスに耐えるためのサーバにすること．これは直接的な対策では無い上に，実現する事が非常に困難である．

2.2　IP Spoofing

通常，ホストやネットワーク同士の接続において，ホストは特定の IP アドレスに制限をかけて接続先をフィルタリングする．しかし，IP Spoofing 攻撃とは攻撃元が IP ヘッダ部の SrcIP アドレスを攻撃対象のアドレス空間のものに偽装することで，攻撃対象へのアクセス制限を突破する攻撃手法であるこの攻撃は攻撃対象のシステムログに嘘の Src アドレスが残るため攻撃元を特定することが非常に困難である．本研究ではランダムな嘘の Src アドレスを作成し，IPSpoofing による DDoS 攻撃手法を提案することで，セキュリティリスクの明確化と低減化を図る．

2.3　DDoS 攻撃の種類

DDoS 攻撃には大きく分けて 2 種類の攻撃がある [4]．一つ目は大きなパケットを大量に送るなどして，サーバまでの接続回線容量を埋め尽くすネットワークの回線の帯域を狙う攻撃である．
もう一つは攻撃対象のサーバを過負荷に陥れるサーバーを狙う攻撃である．本研究では，UDP Flood, ICMP Flood, TCP SYN Flood，TCP Connection Flood を試作する．以下で実現が最も困難である TCP Connection Flood について説明する．

2.3.1　TCP Connection Flood 攻撃

TCP SYN Food 攻撃では確立しない中途半端な TCP 接続を発生させるが，この攻撃ではコネクションを確立する．コネクションを確立させた後は何もしない．コネクション大量にを確立させつづけことでメモリを占領する．TCP Connect Flood 攻撃の際に困難であるのは，どのような方法によって偽装のホストから盗聴して対象から送信された SYN，ACK を手に入れるかである．攻撃の流れを図 1 に示す．

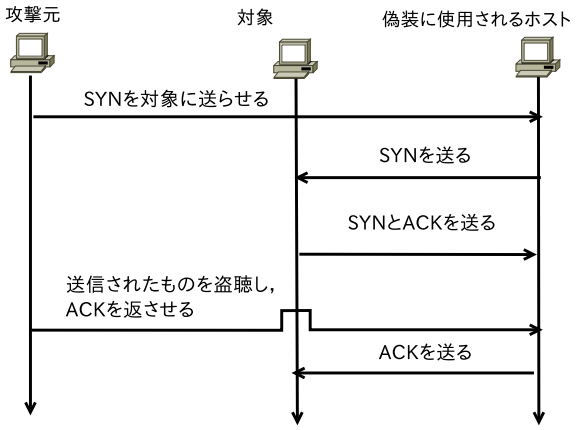

図 1　攻撃の流れ

1. 嘘の Src アドレスを使って対象に SYN パケットを送信する
2. 対象はそれを受けて SYN と ACK を嘘の Src アドレスに送信する
3. 攻撃元は偽装に使用したホストから SYN と ACK を盗聴する
4. 嘘の Src アドレスから ACK を返させる

3　システムの概要

作成する攻撃プログラムの利用想定，実験するネットワーク構成を説明する．

3.1　攻撃プログラムの使用想定

本研究では IPv4，IPv6 それぞれにおいて UDP Flood，ICMP Flood，TCP SYN Flood，TCP Connection Flood 攻撃を選択して攻撃できるものを作成する．攻撃のパラメータは IP のバージョン，攻撃に使用するプロトコル，攻撃の間隔，Src，DstIP の範囲，Src，Dst ポートの範囲を設定できるようにし，操作性向上のために wxglade を用いて GUI を作成した．wxglade とは python で書かれたオープンソースのツールで wxWidgets を使用して GUI を作成できる．wxglade は GUI を作成すると python の他に C++ のコードが出力できる．作成した GUI を図 2 に示す．

図 2　wxglade で作成した GUI

3.2　攻撃プログラムの処理の流れ

はじめに選択された IP のバージョンの raw socket を作る．次に設定された Src，Dst アドレス範囲からサブネットマスクを作成する．そのサブネットマスクを元にホスト部をランダムに変化させ，アドレスを作成する．設定された Src，Dst ポートの範囲からランダムにポートを作成する．その後攻撃の間隔を設定し，最後にそれらを格納し，チェックサム計算をした後，ペイロードを送信する．大まかなプログラムの流れは上記の通りで，それぞれ関数に分けてプログラムを書く．各関数の細かい処理の流れは後の節で述べる．

3.3　ネットワーク構築

攻撃プログラムは外部接続があると危険であるためネットワークエミュレータの CORE 上で動作させる．CORE とは，GUI 機能を持っているネットワークエミュレータで，GUI を持つため簡単に仮想ネットワークを構築することができる．実世界で再現が困難なことテストが可能である．本研究の実験に用いるネットワーク構成図を図 3 に示す．

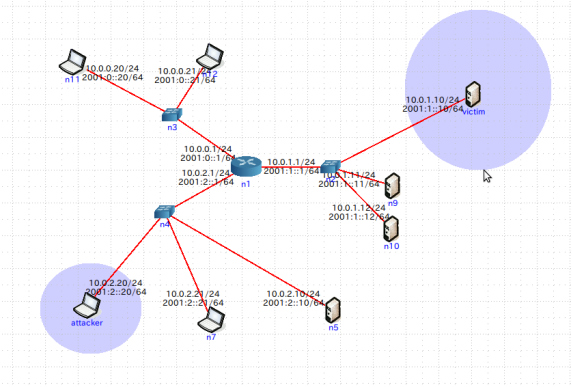

図 3　ネットワーク構成

なお，本研究ではプログラミング言語に C++，OS は Linux Ubuntu 10.04 LTS を用いる．

4　システムの実現

実験環境を作る CORE，C++ で統合した模擬攻撃プログラムの説明をする．

4.1　クラスの設計

本研究ではクラスを 4 個に分けてプログラムを構成する．二つは wxglade によって出力される MyApp，MyFrame．MyApp は GUI を起動するためのクラスで，MyFrame は GUI で入力した値を受けとるものである．三個目は DDoSTester で，これは raw ソケットを生成，嘘のランダムアドレス作成など攻撃に必要なパラメータを設定，ペイロードを送信するクラスである．最後のクラスは MainThread で，これは MyFrame が受け取った値を DDoSTester に投げる GUI と攻撃プログラムの架け橋に

なるクラスである．

4.2 DDoSTester.cpp

本研究のメインとなる攻撃プログラムである．細かい処理毎に関数に分けて作成し，それらをまとめたクラス．

4.2.1 raw ソケットの生成

raw ソケットとは通常のソケットとは異なりヘッダ情報を含んだデータを送受信できる．しかし，raw ソケットを扱うためには普段意識することのないプロトコルの詳細まで設定しなければならない．そのため，raw ソケットプログラミングに慣れていない人にとっては実装が容易ではない．

raw ソケットの生成

```
if (ver == IPv4){
    raw4sock = socket(AF_INET,
               SOCK_RAW, IPPROTO_RAW);}
else if (ver == IPv6)
    raw6sock = socket(AF_INET6,
               SOCK_RAW, IPPROTO_RAW);

  pfsock = socket(AF_PACKET,
           SOCK_RAW, htons(ETH_P_ALL));
```

4.2.2 嘘の Src アドレス作成

IP Spoofing 攻撃では Src アドレスを偽装する必要があるため嘘の Src アドレスを生成するプログラムが必要である．嘘のアドレスをランダムで 100 個用意する．指定したアドレス範囲でランダムにアドレスを生成するために，まず IP アドレスとプリフィックス長からサブネットマスクを作成する．ソースコードは以下の通りである．

IPv4 のサブネットマスク作成のコード

```
size_t found = addressRange.find("/");
  if (found == string::npos) {
   cerr << "/ not found in addressRange" << endl;
   return false;
  }
string netAddrStr = addressRange.substr(0,found);
string prefixStr  = addressRange.substr(found+1);

  if (ver == IPv4){
inet_pton(AF_INET, netAddrStr.c_str(), &saddr);
//  アドレスを文字列からアドレス構造体に変換し saddr
にコピーする
smask.s_addr = htonl(0xffffffff <<
            (32-atoi(prefixStr.c_str())));
//  32 ビットからプリフィックス分だけ 1 を左にシフト
させる
```

サブネットを作成した後にランダムでアドレスを作成する．

IPv6 ランダム Src アドレス作成のソースコード

```
  struct in6_addr randomAddr;

  for (int j=0; j < 4; j++){
//アドレスを 32 ビットずつに 4 回分けて生成する
   randomAddr.s6_addr32[j]
      = (addr.s6_addr32[j] & mask.s6_addr32[j])
        //この部分でネットワーク部を固定
      | ((u_int32_t) (2147483648*drand48())
      &  ~mask.s6_addr32[j]);
      //この部分でホスト部をランダムに作成
    }
    return randomAddr;
  }
```

4.2.3 sendPayload

上記で作成してきたランダムアドレスを sendbuf を用いてペイロード送信するために IP ヘッダなどに適切なパラメータを埋める．fill4checksum() と fill6checksum() でチェックサムを確認し，sendto 関数を使用し送信する．使用ソケット，メッセージ，メッセージのサイズ，フラグ，接続先のアドレス，そのアドレスのサイズの情報を格納している．

4.2.4 チェックサム

チェックサムとは，誤り検出方法の一つである．パケットを送受信する際にパケットが正しいかの判断材料にできる．チェックサムは，対象となるパケットに対し，1 の補数和を取り，さらにそれの 1 の補数を取る．本研究では以下のアルゴリズムを用いてチェックサムを計算する．

チェックサムのアルゴリズム

```
void checksumadjust(unsigned char *chksum,
            unsigned char *optr, int olen,
            unsigned char *nptr, int nlen);
{
     long x, old, newdata;
     x=chksum[0]*256+chksum[1];
     x=~x & 0xFFFF;
            :

     while (nlen){
         newdata=nptr[0]*256+nptr[1]; nptr+=2;
         x+=newdata & 0xffff;
         if (x & 0x10000) { x++; x&=0xffff; }
         nlen-=2;}
     x=~x & 0xFFFF;
     chksum[0]=x/256; chksum[1]=x & 0xff;}
```

このアルゴリズムを用いるためのクラスが fillV4checksum，fillV6checksum である．プロトコルを判別した後，checksumadjust によりチェックサム計算する．

4.2.5 run()

TCP Connection をするために run() を作成する．run() は受信するものである．SYN と ACK を返すプログ

ラムの部分になる．if() を使用し，IPv4，IPv6 両方を作成する．以下に run() の作成できた部分まで示す．IPv4 か IPv6 でなければ continue で while() の先頭に戻す．IPv4 か IPv6 であるならば，次に TCP であるか確認する TCP でなければ，continue で while() の先頭に戻す．TCP であるならば，SYN と ACK が 1 であるか確認する SYN と ACK が 1 でなければ while() の先頭に戻す．SYN と ACK が 1 ならば次の処理をする関数を作る．

5 実験

実験の手順，実験の結果について示す．実験する攻撃は IPv4，IP6 の UDP Flood，ICMP Flood，TCP SYN Flood，TCP Connection Flood である．

5.1 実験手順

実験の手順は以下の通りである．

1. CORE 内でネットワークを構築する
2. Attacker から Victim に攻撃プログラムを実行する．
3. Victim で tcpdump コマンドを使用し，観測する

IPv4 アドレスで実験する場合は SrcIP のアドレス範囲は 192.168.0.0/16，DstIP のアドレス範囲は 10.0.0.1/32，Src ポートの範囲は最大 61000，最小は 60000，Dst ポートの範囲は最大最小共に 8000，攻撃間隔は 1.0 秒に設定する．また IPv6 アドレスで実験する場合は SrcIP のアドレス範囲は 2001:fc00::/64，DstIP のアドレス範囲は 2001:1::10/64 に設定する．

5.2 実験結果

IPv4 の TCP SYN Flood 攻撃の実験結果を以下に記述する．

IPv4 TCP SYN Flood Attacker

```
Random V4 addr: 192.168.54.114
Random Port: 60375
Random V4 addr: 10.0.1.10
Random Port: 8000
Interval(usec): 1388271
-------------
Random V4 addr: 192.168.24.98
Random Port: 60513
Random V4 addr: 10.0.1.10
Random Port: 8000
Interval(usec): 115156
-------------
Random V4 addr: 192.168.241.56
Random Port: 60109
Random V4 addr: 10.0.1.10
Random Port: 8000
Interval(usec): 1594761
.................
```

Attacker では設定した Src アドレスの範囲，Src ポートの範囲で嘘のランダム IPv4 アドレス，ポートが生成されている事が確認できた．Victim 側では以下のようにキャプチャできた．

IPv4 TCP SYN Flood Victim

```
tcpdump -l

15:43:34.926022 IP 192.168.73.96.60228
> 10.0.1.10.8000: Flags [S],
seq 1599409180:1599409260, win 60000, length 80

15:43:35.703877 IP 192.168.76.12.60738
> 10.0.1.10.8000: Flags [S],
seq 1399793690:1399793770, win 60000, length 80

15:43:36.118276 IP 192.168.77.95.60592
> 10.0.1.10.8000: Flags [S],
seq 3030214733:3030214813, win 60000, length 80
.....................................
```

嘘の Src アドレス，ネクスト・ヘッダ，ペイロード長が正しく表示されており，これらのパケットが Attacker から Victim に届いていることが確認できた．そのため IPv4 の TCP SYN Flood 攻撃は成功していることが分かる．

6 おわりに

本研究では，CORE 内の Attacker で攻撃プログラムを実行し，Victim で tcpdump コマンドを用いて観測したところ，IPv4 の UDP Flood，TCP SYN Flood，ICMP Flood IPv6 の ICMPv6 Flood は実験成功した．また攻撃のパラメータを設定し，攻撃を開始できるよう一つのクラスにまとめることができた．wxglade で GUI を作成したが，送信開始の処理が完成できなかった．また，DDoSTester.cpp の run() 部分が完成しなかったため，IPv4，IPv6 ともに TCP Connection Flood 攻撃は観測できなかった．run() を作成したら TCP Connection Flood も観測できると考えられる．

参考文献

[1] Marcello Semboli and Alberto Griggio and Carsten Grohmann: wxGlade manual (accessed Jan. 2016). http://wxglade.sourceforge.net/manual/.

[2] U.S. Naval Research Laboratory Networks and Communication Systems Branch: Common Open Research Emulator (accessed Dec. 2015). http://www.nrl.navy.mil/itd/ncs/products/core.

[3] 大平峻也，山下瑞樹：IPv6 のための異常トラフィック生成プログラムの試作と評価，南山大学システム創成工学科 2014 年度卒業論文 (2015).

[4] 田村奈央：ぜい弱性に揺らぐインターネット (accessed Jan. 2016). http://itpro.nikkeibp.co.jp/article/COLUMN/20090225/325453/.