WIDE Technical-Report in 2008

WIDE-CNRS間の交換留学活動報告

wide-tr-mawi-widecnrs-kanai-00.pdf

WIDE Project : http://www.wide.ad.jp/

If you have any comments on this document, please contact to ad@wide.ad.jp

WIDE-CNRS間の交換留学活動報告

金井 瑛 (kanai@sfc.wide.ad.jp)

平成 20 年 1 月 29 日

1　概要

WIDE Project はフランス国立科学研究センター（CNRS）との研究協力の一環として、両組織間で人的交流・学術的交流を目的とした、学生の交換留学制度を設けている。私はこの交換留学生として、2007 年 09 月 18 日から同年 11 月 10 日にかけて約 2ヶ月間渡仏した。この期間中、私はフランス南部トゥールーズにある Laboratoire d'Architecture et d'Analyse des Systemes(LAAS) で受け入れていただいた。期間中は主に、LAAS の研究者である Philippe Owezarski 氏と博士課程学生である Ion Alberdi 氏に研究の体制を用意していただいた。特に Ion 氏と私はボットネットを共通のテーマとして取り扱っているため、多くの知識を共有し、また、今後に向けた研究体制を整えることができた。

2　イベントへの参加

本留学では、LAAS での共同研究に加えて、パリで開催された Mozilla24 中継スタッフとしての参加および CNRS-WIDE ミーティングへの参加をした。本節ではそれぞれの取り組みについて述べる。

2.1　Mozilla24 中継スタッフ

私は、期間中の最初の 10 日間を Mozilla24 の中継スタッフとして参加するため、パリに滞在した。Mozilla24 は、日本、フランス、イギリスとタイの４ヵ国間をネットワークで接続し、各会場からの講演やディスカッションを実現するイベントである。このイベントは、ヨーロッパの大手学術ネットワークの１つである Renater、パリを拠点とする大手通信事業者 TELECOM ParisTech ENST(以下 ENST)、映像のプロフェッショナル組織である CERIMES の協力を得て実現した。

Mozilla24 はパリ市内の ENST で開催された。到着後、ENST の Pierre Beyssac 氏、Renater の Simon Muyal 氏, 同組織の Virginie BLANQUART 氏及び Mozilla ヨーロッパ代表の Tristan Nitot 氏らとミーティングを行った。到着後から１週間の準備の間、会場の映像環境及び、日本間との通信環境について確認し、低遅延と高品質な映像・音声を提供できる DVTS を利用した環境を整備した。Mozilla24 では日本を拠点として各国と映像と音声の通信を行うため、ENST からは日本との双方向通信が必要であった。

イベント実施日は約 40 人の受講者がフランス会場に集まり、4 名の講演者がフランスより講演を行った。また、受講者は日本を介したタイや、東京からの講演を聴講した。イベント中は特に大きな障害も発生せず、イベントは成功を収めた。

イベントを通して、Renater や Mozilla、ENST のメンバと交流を深めることができ、今後のフランスで行う CNRS-WIDE の研究活動をより円滑に行えることが期待できる。

2.2 CNRS-WIDE ミーティングへの参加

2007 年 10 月 15 日から 16 日にかけて、フランス中部の Lyon で第 4 回の CNRS-WIDE ミーティングが開かれた。会期が留学期間中であったため、私は LAAS から参加した。本ミーティングでは、WIDE から 11 名、CNRS から 22 名の研究者が参加し、measurement と mobility の 2 部構成でそれぞれの研究成果を発表し合った。本ミーティングでは、CNRS 側の measurement の研究者と話す機会が多くあり、インターネット上の攻撃特性に関する情報や、DDoS 検知の手法などについて、有益な議論を交わすことができた。

3 成果

本留学中に私は LAAS の Philippe 氏からご指導を頂きながら、Ion 氏との共同共同研究に取り組んだ。本節では、私が LAAS で取り組んだ活動の成果について述べる。

攻撃者により管理された多数のノードが構成するボットネットはその可変性と分散性などから対応が極めて難しい。私はこれまで、フロー型検知手法を用いたボット検知手法を研究してきた。フロー型検知手法はホスト毎のフローを保持し、その順序を追跡してボットを検知する。また、Ion 氏は世界中に分散したハニーポットを用いて効果的にボットの情報を取得する研究や、取得したボットを安全に実行する研究に携わってきた。Ion 氏や私のようなボットの研究者は、日々進化するボットの知識を得るために、ハニーポットを用いている。**ハニーポット**は脆弱性を持つホストの挙動を振る舞うソフトウェアであり、ボットネットの研究を進める上で有効なツールである。研究では運用コストや法的な問題から低インタラクション型のボットネットが用いられることが多い。しかし、一般的な低インタラクション型の環境から得られる情報は、セッション型ボット検知機構に対して十分な情報とならない。

本留学では、この問題を解決するトラフィック合成機構の実用化に向けた開発を行った。

3.1 現在の問題点

まず、一般的な低インタラクション型ハニーポットを用いたボットトラフィック収集の環境を図 1 に示す。

図中では、IP アドレス IPa を持つマシン上で低インタラクション型ハニーポットが稼働している。ハニーポットの稼働しているマシンの通信はすべてトラフィックミラーポイント MPa でミラーリングされ、蓄積される。MPa のトラフィックダンプデータにはハニーポットが対応したすべてのボットウェアに対してのトラフィックが蓄積される。ハニーポットで蓄積したボットウェアは手動で IP アドレス IPb を持つ実行マシンにコピーする。法的な問題から実行マシンからのトラフィックは管理者の設定したポリシーによって、フィルタアウトされる。また、実行マシンの通信はトラフィックミラーポイント MPb によってミラーリングされ、蓄積される。

2 つに分かれた PCAP ファイルのパケットは例えば *tcpmerge* 等の既存のソフトウェアを利用して合成できる。*tcpmerge* は 2 つの PCAP ファイル内のパケットの時間情報を保持したまま、パケットを結合する。しかし、結合に際して、**PCAP ファイルの持つ情報の差**の問題が発生する。

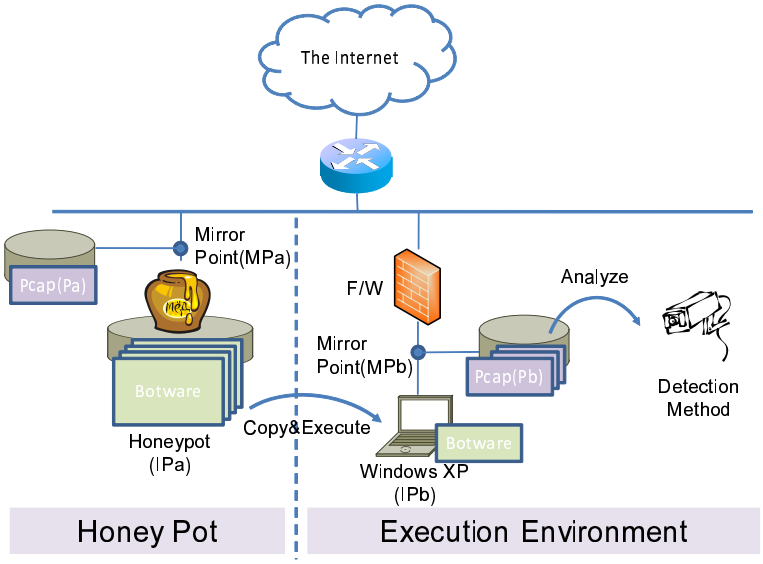

図 1: 一般的なハニーポット環境の例

また、問題を解決しトラフィックを結合したのみでは **IP アドレスの差**と**時間情報の差**の 2 つの問題が発生する。これら 3 つの問題は、フロー型ボット検知機構の評価に大きな影響を及ぼす。

- **保持されているボットネット情報の差**
  MPa のトラフィックダンプデータにはトラフィックをダンプしている期間中のあらゆるボットウェアのトラフィックが含まれる。しかし、MPb のトラフィックダンプデータは実行毎にトラフィックをダンプするため、ボットウェア毎に分かれたダンプファイルとなっている。現在、私が取り組んでいるフロー型ボット検知手法の評価に利用するには、攻撃からボットの活動トラフィックまで全てのトラフィックが必要である。そのため、MPa のデータからボット毎のトラフィックを抜き出す必要がある。

- **IP アドレスの差**
  ハニーポットマシンと実行環境マシンは保持する IP アドレスが異なる。そのため、トラフィックを結合しただけでは実際とは異なったマシンが通信しているようなトラフィックが生成される。たとえば、図 1 の例では、ダウンロードまでのトラフィックは IPa が、ボットウェアの実行トラフィックは IPb が外部と通信をしているトラフィックとなる。

  フロー型ボット検知機構では、内部ホストのフローの順序を利用して検知するため、IPa と IPb の行う別の通信として扱われ、正常な評価が実現できない。

- **時間情報の差**
  実行環境トラフィックは、ボットウェアを収集した直後に収集されていないので、実際の動作と時間情報に差が出る。この様子を図 2 に示す。ただし、$Tr1$ はハニーポットトラフィック、$Tr2$ は実行環境トラフィック、$Tr3$ は時間情報を保持したまま結合したトラフィック、$Tr4$ は実際のトラフィックとする。また、$ts1$ はハニーポットトラフィックの開始時刻、、$te1$ はハニーポットトラフィックの終了時刻、$ts2$ は実行環境トラフィックの開始時刻、$te2$ は実行環境トラフィックの終了時刻とする。図はそれぞれのトラフィックに含まれるパケットを時系

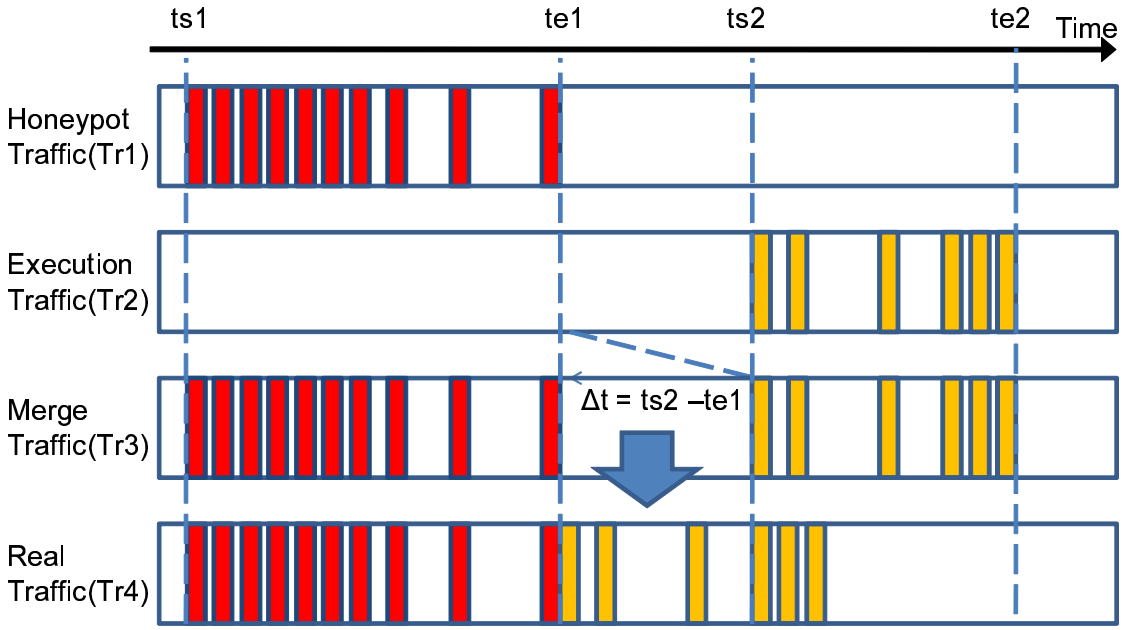

図 2: 実機のトラフィックとハニーポット環境のトラフィックに生じる時間情報の差

列に表現している。Δt は式 1 のように表わされる。

$$\Delta t = ts2 - te1 \tag{1}$$

一般的なハニーポット環境においては、ボットウェアのダウンロード時間とボットウェアの実行やその準備に時間を要するが生じるため、$Tr2$ の全トラフィックは実際に比べて実行までに要した時間 Δt だけ遅れた時間をもつパケットとなる。実際のトラフィックでは、ボットウェアのダウンロード後、即座にボットウェアが実行されるため、$te1$ と $ts2$ の差は極めて小さい。

フロー型ボット検知機構を含む多くのトラフィック監視するセキュリティ機構では、フロー間の間隔が長いとそれらを連続したフローとして扱わない。そこで、$Tr2$ は Δt だけ時間を早くして $Tr1$ と結合し、実際のトラフィックに近づける必要がある。

3.2 アプローチと実装

本留学では、前節で挙げた問題を解決するために、ハニーポットへのパッチおよびいくつかのネットワークツールを作成し、それらを組み合わせて問題を解決したトラフィックを合成するプロトタイプ実装を行った。

- **ハニーポットへのパッチ**
  現在、私が WIDE ネットワーク上で用いているハニーポットと Ion 氏が LAAS ネットワーク上で用いているハニーポットはオープンソースソフトウェアの低インタラクション型ハニーポット Nepenthes である。保持されているボットネット情報の差の問題を解決するためには、ハニーポットトラフィックからボット毎のトラフィックを抜き出せなければならない。これに

は、ハニーポットの出力するログを利用する。しかし、Nepenthes はボット毎のトラフィックを抜き出すのに必要な情報を出力しない。そのため、私は Nepenthes に対して必要な情報を出力するように変更を加えた。変更が必要な情報を挙げる。

- **UNIXTIME**
  各ログは発生した時間を現地時間で保持している。しかし、PCAP ファイルは UNIX タイムでパケットの発生時間を保持している。そのため、時間情報の統一のためにログに UNIXTIME を保持させるようにした。
- **IP アドレスとポート番号の情報**
  ログからパケットを抜き出すには通信に関する IP アドレス情報とポート番号が必要となる。そのため、各ログに関連する IP アドレスとポート情報を保持させるようにした。
- **識別子**
  Nepenthes は攻撃以降のイベントがどの攻撃イベントによって発生したかを保持しない。たとえば、ボットの本体をダウンロードするイベントがどの攻撃によって発生したかが不明である。あるボットウェアのトラフィックのみを適切に抜き出すにはログの関係が分かる必要がある。そのため、各ログ間の関係をユニークな識別子を各ログに保持するようにした。

この結果、図 3 のようなログが出力される。この例では、各ログに UNIXTIME が記述され、

```
(12102007 22:21:36 info handler dia) Time: 1192195296,
Info: KNOWN DCOM(2) ATTACK RECEIVED, UniqueId: 1,
Local: 203.178.xxx.xxx:135, Remote: 203.174.yyy.yyy:1258
12102007 info mgr submit Time: 1192195331

(12102007 22:21:36 info handler dia) Time: 1192195296,
Info: BLINK DOWNLOAD START, UniqueId: 1,
Local: 203.178.xxx.xxx:33286, Remote: 203.174.yyy.yyy:54791

(12102007 22:22:11 info mgr submit) Time: 1192195331,
Info: SUBMITED, UniqueId: 1,
FileName: 2aa59ba4251795deda72738d1c67be7c,
FileType: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
```

図 3: パッチを当てた Nepenthes のログ例

ネットワーク通信を行ったログには、通信先の IP アドレスと使用したポート番号が記述されている。また、それぞれのログは UniqueId と定義されたユニークな識別子を保持しており、これらが関連したログであることが分かる。

- **PCAP Address Replacer**
  PCAP Address Replacer は C 言語で記述されたパケット内部の IP アドレスと MAC アドレ

スを書き換えるコマンドラインソフトウェアである。パケットに含まれるコマンドラインから指定したアドレスを置き換え、また、IP チェックサムおよび、L4 のチェックサムを再計算する。IP は IPv4 および IPv6 をサポートする。また、L4 プロトコルは TCP と UDP をサポートする。

- **PCAP Appender**
  PCAP Appender は C 言語で記述された 2 つの PCAP ファイル内のパケットの時間を調整し、それぞれの PCAP ファイルを結合するソフトウェアである。このソフトウェアは、infile1 のトラフィックを $Tr1$、infile2 のトラフィックを $Tr2$ として扱い、$Tr2$ の各パケットの時間軸を $-\Delta t$ ずらすことで、$Tr4$ に近いトラフィックを生成して、outfile に書き出す。

- **PCAP Slicer**
  PCAP Slicer は保持されているボットネット情報の差を解決するためのシェルスクリプトである。本スクリプトはパッチを当てた Nepenthes のログから、ボット毎の攻撃フローとダウンロードフローの通信先 IP アドレスとポート番号を抜き出す。そして、ハニーポットのトラフィックからそれぞれのトラフィックを抜き出し、PCAP Address Replacer と PCAP Appender を用いてトラフィックを合成する。

3.3 現状と今後

前節の実装は大きくパッチを当てたハニーポットと、トラフィック合成プログラムの 2 つに分かれる。このうち、パッチを当てたハニーポットは、WIDE 内のハニーポットおよび、Ion 氏の協力を得た LAAS 内のハニーポットで稼働中である。

我々は、将来的には収集したボットウェアを自動的に実行しトラフィック合成できることを期待している。しかし、ボットウェアの自動的な実行は、法的な側面や、外部への攻撃を防ぐ難しさの障害から、今回取り組むことが出来なかった。

現在、Ion 氏は自動なボットウェアの実行に利用できるファイアウォールの研究に取り組んでいる。今後も共同研究を進めていき、本成果と Ion 氏の成果を組み合わせて、自動的なボットウェアトラフィックの収集機構の構築を目指したい。

4 まとめ

本留学では、Mozilla24 への中継スタッフとしての参加及び、LAAS での共同研究に取り組んだ。Mozilla24 では Renater、ENST や Mozilla のメンバと共に作業に取り組み、また交友を深めた。これにより、フランスにおける研究だけではない様々な活動において、WIDE プロジェクトがより円滑に協力できる関係を築けた。LAAS での共同研究では、Ion 氏との共通の問題点に取り組み、また LAAS のスタッフと交友を深めることができた。トラフィック合成に関する問題に対する取り組みは、今後の双方の研究活動に有用なものであり、その方向性について議論できたことは非常に有意義な経験であった。現在、Ion 氏は WIDE ネットワークにハニーポット環境を構築しており、また、私も今回の留学で LAAS の実験ネットワークにハニーポット環境を構築させていただいている。トラフィックを監視するという活動の性質から、双方の交友が深まったことは今後の活動の際にも貴重な機会であった。今後もお互いの情報を共有しつつ、研究を進めていきたい。