マルチメディア通信と分散処理 111－8
コンピュータセキュリティ 20－8
（２００３．２．２７）

DHCPを用いた受動的フィンガープリンティング手法の提案と実装

白畑 真† 土本 康生†† 村井 純†

概要

これまでリモートノードの Operating System(以下 OS) を特定するフィンガープリンティング手法では、OS と IP アドレスの対応付けは変化しないことが前提とされている。しかし、動的 IP アドレス割当が行われているネットワークにおいては、あるノードに割当られた IP アドレスが時間の経過に伴い別のノードに割当られるケースが考えられる。従って、フィンガープリンティングによって OS 情報と IP アドレスの対応付けに矛盾が生じる可能性がある。
本手法ではこの問題点を解決するため、DHCP メッセージをキャプチャすることで IP アドレスの割当がどのように行われているかを把握できる点に着目し、動的な IP アドレス割当環境下においても常に新しい情報が得られる受動的フィンガープリンティング手法を提案する。また、実際に本手法を実装し、実ネットワークでの運用と評価を行った。

New scheme for passive OS fingerprinting using DHCP message

Shin Shirahata† Yasuo Tsuchimoto†† Jun Murai†

Abstract

At this point, many scheme are proposed for OS fingerprinting. In these scheme is an assumption that the combination of the OS information and the IP address are never changed. Unfortunately, is a dynamic IP address allocation environment, an IP address that os allocated to a certain node could be reallocated to another node with the passage of time. Therefore, there may be a contradiction between the combination of the OS information and the IP address from fingerprinting.
To resolve this contradiction, We introduce a new passive OS fingerprinting scheme that focus attention on the dynamic IP address allocation by sniffing DHCP messages. We have implements a new OS fingerprinting tool using this scheme, and evaluated it on a working network.

1 はじめに

インターネットセキュリティの意識の高まりとともに、リモートノードで稼働する OS を特定する OS フィンガープリンティングについても関心が高まっている。これまでセキュリティ監査や侵入準備行為などとして、どちらかといえば攻撃者サイドの技術として捉えられることの多かった OS フィンガープリンティングであるが、ネットワーク管理者の立場から積極的に OS フィンガープリンティングの情報を活用する研究も進められている [1]。

このように管理者が OS 情報を知ることは重要であるが、その際、どのように管理者が OS 情報を取得するかが課題となる。小規模なネットワークならまだしも、大規模なネットワークにおいては管理者が個々のノードの OS を一台一台調査することは多大な労力を要する。特に動的 IP アドレス割当環境下では IP アドレスの割当状況が時間とともに変化するため、管理者が個々に調査を行うことは非現実的といえる。

2 既存の OS フィンガープリンティング手法

2.1 既存の手法

現在の OS フィンガープリンティングの手法は、大きく能動的手法と受動的手法に分類できる。能動的手法では、リモートノードに対して

† 慶應義塾大学環境情報学部
Keio University, Faculty of Environmental Information
†† 慶應義塾大学政策メディア研究科
Keio University,
Graduate School of Media and Governance

IP パケットや TCP セグメント、ICMP メッセージを送信し、ノードからの応答を元に OS を判別する方法が採られている [2]。一方、受動的手法では、フィンガープリンティングのためにリモートノードに対してパケットを送信せず、ノードが送信する TCP セッションをキャプチャし、シーケンス番号の変化やウィンドウサイズ、IP TTL などの情報からノードの OS を判別する方法が採られている [3]。

代表的な能動的フィンガープリンティングツールの実装としては nmap[4] や X-Probe[5] が挙げられ、受動的フィンガープリンティングツールの実装としては siphon[6] や p0f[7] が挙げられる。

3 フィンガープリンティング手法の制約

既存のフィンガープリンティングによって得られた情報を活用する際、特定の IP アドレスが特定のノードで継続的に利用されることが前提条件とされている。通常、サーバマシンなどのノードには IP アドレスの固定割当が行われるため、フィンガープリンティングによって得られた結果は不変である。そのため、固定 IP アドレス割当環境下においては、フィンガープリンティングによって得られた OS 情報と IP アドレスの対応付けの一貫性が保たれると考えられる。

しかし、DHCP のような動的な IP アドレス割当を行っている環境下においてはその限りではない。例えば、ノード A に割当られていた IP アドレスが一定時間経過後に別のノード B に割当られた場合では、特定の IP アドレスに対するフィンガープリンティングの結果は正確なノードの情報を反映しているとはいえない。従って、得られた情報を活用する際には、フィンガープリンティングを行った後に生じた IP アドレスの割当状況の変化によって、OS 情報と IP アドレスの対応付けに矛盾が生じる可能性がある。

動的 IP アドレス割当環境下においては、能動的手法を用いる場合には常に最新の IP アドレス割当状況に対応した OS 情報の取得は困難である。既存の受動的手法を用いる場合においても、TCP セッションの情報から OS の判別を行っているため、TCP による通信が行われない限り OS の判別を行うことは不可能である。また、ネットワーク全体のフィンガープリンティングを行う場合には、外部ネットワークとの境界においてポートミラーなどによりトラフィックデータを入手する必要がある。

3.1 フィンガープリンティング手法に求められる要件

前節において説明した制約から、ネットワーク管理者から OS フィンガープリンティングを利用する場合の求められる要件を明らかにする。

整合性　前項で述べたとおり、フィンガープリンティングの結果を用いるためには、OS と IP アドレス情報の対応付けが正しくなされていることが重要である。そのため、動的 IP アドレス割当環境下において、常に最新の OS と IP アドレスの対応付けを保つこと。

正確性　ノードの OS の種類を誤認せず、得られる情報が正確であること。

効率性　トラフィックの発生量や対象ノードの負荷が最小限となること。

迅速性　迅速にフィンガープリンティングの結果が得られること。

規模性　大規模なネットワークにおいても利用できること。特に、フィンガープリンティングの処理にかかる時間が短くなければならない。

調査可能性　フィンガープリントの対象ノードや経路上の機器 (ファイアウォール等) の設定に影響されずに、結果が得られること。

能動的フィンガープリンティング手法では、調査に用いるパケットに対する応答を得ること

ができれば、インターネットに接続された全てのノードの調査が可能であるが、経路の途中でフィンガープリンティングに利用するパケットがフィルタされている場合などでは正確な結果を得ることが困難である。例えばICMPを用いた実装であるX-Probeは、ICMPの利用が制限されている中継ノードやエンドノードが存在する場合にはOSの判別ができない。

4 設計

4.1 DHCPメッセージを用いたフィンガープリンティング手法

これまで述べてきた通り、既存のフィンガープリンティング手法では動的IPアドレス割当環境を考慮されていなかった。本研究ではこれらの問題を解決するため、上記に挙げた要件に適合する動的IPアドレス割当環境に適合したフィンガープリンティング手法として、DHCPメッセージに着目したフィンガープリンティング手法を提案する。

DHCPに着目した理由として、DHCPクライアントは必ずDHCPREQUESTメッセージを送信するため、フィンガープリンティングのために必要なデータが必ず取得できる点が挙げられる。これにより経路上やDHCPクライアントの設定に依存せずフィンガープリンティングを行うために必要なデータが取得できる利点がある。

4.2 DHCPメッセージの構造

図1に示すように、DHCPメッセージの中でDHCPクライアントの実装による差異が認められるのは、DHCP Option[11] フィールドである。中でも、

- 55 (Parameter Request List)
- 60 (Vendor class identifier)

はDHCPクライアントによる差異が大きいため、DHCPクライアントからOSの特定する際の有益な情報となる。

図1: DHCPメッセージの構造

Vendor class identifier

Vendor class identifierはDHCPクライアントがベンダの種類や設定情報を識別するために用いることができる情報である。従って表1に示すように、この値にはDHCPクライアントのOSのベンダ名やバージョンを入れる実装が多い。例えばVendor class identifierにMSFT 5.0という値が入っている場合[12]、当該ノードはMicrosoft Windows 2000またはXPで稼働していると推測できる。また、uDHCPのようにDHCPクライアント名が値として与えられる場合にも、当該DHCPクライアントが特定のOSでしか動作しない場合には、DHCPクライアントの種類からOSを特定できる。

表1: Vendor class identifierの値とOSの一例

Data	OS
MSFT 5.0	Windows 2000 / XP
MSFT 98	Windows 98
Linux 2.4.18-17.7.x	Linux (Kernel 2.4.18)
Mac OS J1-9.2.2	MacOS 9.2.2
uDHCP	uDHCP client (Linux)

ただし、このVendor class identifierはオプションであるため、このオプションを送信しないDHCPクライアントも存在する。

Parameter Request List

図 2 に示すように、Parameter Request List は DHCP クライアントが特定の種類の設定情報を要求する際に利用するパラメータである。この要求パラメータには複数の値を含めることができる。

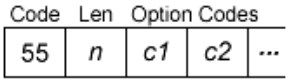

図 2: Parameter Request List

表 2 に示すように、DHCP クライアントの実装により、その値の内容や順序が異なるため、Vendor class identifier の値だけでは OS を特定が困難な場合においても OS を特定可能である。Parameter Request List も DHCPREQUEST においてはオプションであるが、DNS サーバの IP アドレスやサブネットマスク [13] など、TCP/IP ネットワークでは必須ともいえる情報を要求する際に必要なため、ほぼすべての DHCP クライアントが実装している。

表 2: Parameter Request List の値と OS の一例

Parameter Request List	OS
01,03,15,06,44,46,47	Windows 95
01,15,03,44,46,47,06	Windows NT 4.0
01,03,06,15,112,113,78,79	MacOS X
01,28,02,03,15,06,12	ISC dhclient (various UNIX)

5 実装

5.1 実装

図 3 に示すように、本実装では、DHCP クライアントが IP アドレスを取得する直前に送信する DHCPREQUEST メッセージ [10] を分析し、フィンガープリンティングを行った。ただし、この時点では DHCP クライアントが稼働するノードの IP アドレスは不明であるため、MAC アドレスと OS の情報を対応付け、メモリ上に格納する。

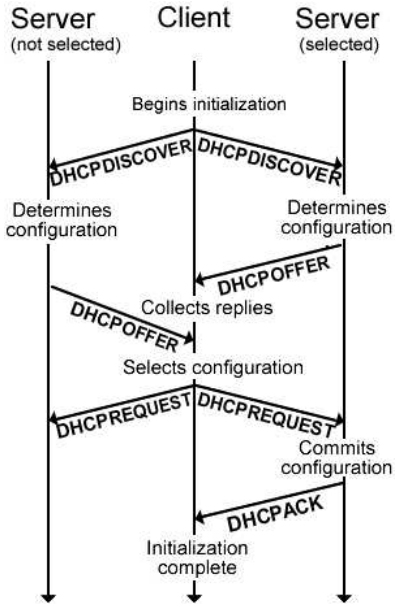

図 3: DHCP による IP アドレスの割当手順

次に、フィンガープリンティングの結果を IP アドレスと対応付ける必要がある。そのため、すでに得られた MAC アドレスから IP アドレスを求めることが必要となる。本実装では、図 4 に示すように、DHCP クライアントが IP アドレス取得後、Gratuitous ARP と呼ばれる特別な ARP リクエストをブロードキャストすることを利用する。この ARP リクエストは IP アドレスの重複を検出するため、および他のノードの ARP キャッシュを更新させるために送信される [9]。これにより、OS 情報を IP アドレスに対応付けることができた。

また、フィンガープリンティングの結果得られた情報を他のシステムから利用可能にするため、IP アドレスと OS 名、実行時刻を RDBMS に格納できるオプションを用意した。

5.2 実装環境

Trustix Secure Linux 1.5 (kernel 2.2.22, RedHat Linux 6.2 ベース) および RedHat Linux 7.2 (kernel 2.4.18) 上で、C 言語を用いて実装

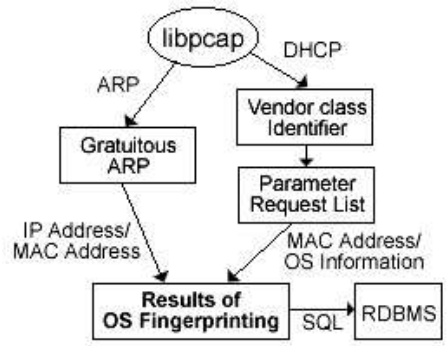

図 4: プログラムの構造

を行った。パケットキャプチャのためのライブラリとして、Libpcap[8] を用いた。また、RDBMS を利用するためのライブラリとして libmysqlclient を用いた。

6 評価

整合性 IP アドレスを割当るために利用される DHCPREQUEST メッセージを用いて OS フィンガープリンティングを行うため、IP アドレスと OS 情報の対応付けの整合性が保たれる。ノードに複数の OS がインストールされている場合においても、OS の起動の際に DHCPREQUEST メッセージが送信されるため、問題は生じない。

正確性 一部の DHCP クライアントは、設定により Vendor class identifier や Parameter Request List の値を変更できるため、フィンガープリンティングツールに対し、OS を偽装することが可能である。しかし、通常の利用の範囲では Vendor class identifier の内容や Parameter Request List の値の順番を変更することは考えられないため、OS を正確に判別できるといえる。

効率性 本手法はネットワーク上の DHCP メッセージおよび ARP パケットをキャプチャし、分析する受動的な手法である。従ってフィンガープリンティングに伴うトラフィックは発生しない。

迅速性 DHCP クライアントは DHCP のリース期限が切れるまでに IP アドレスの利用延長を行う。その際、DHCP クライアントは必ず DHCPREQUEST メッセージを送信するため、本手法では遅くともリース期限内には全てのクライアントをフィンガープリンティングの対象とすることができる。

規模性 今回、MMX Pentium 233 MHz、メモリ 160MB のマシンにおいて、tcpdump の出力ファイルからデータを読み込み、出力を /dev/null にリダイレクトしてテストを行った。6361 個の DHCPREQUEST メッセージから OS フィンガープリンティングを 10 回行った結果の平均が 3.44 秒となった。1 個の DHCPREQUEST メッセージあたりのデータ処理に要する時間は 0.5ms 程度であったため、大規模なネットワークにおいても本手法は適用できると思われる。

調査可能性 本手法においては、同一ブロードキャストドメイン内に存在するすべての DHCP クライアントを対象にフィンガープリンティングを行える。さらに、初回 IP アドレス取得時に DHCP クライアントは DHCPREQUEST メッセージをブロードキャストするため、フィンガープリンティングを行うノードは DHCP クライアントと同一ノードに設置するだけでよく、ネットワークの境界でのポートミラーを行う機材を必要としない。

また、ブロードキャストドメイン外からは本手法によるフィンガープリンティングを行えないため OS 情報が外部に露呈する危険性はなく、ネットワーク管理目的での利用に適している。

7 おわりに

7.1 応用例

フィンガープリンティングの結果得られた情報の応用例としては、脆弱性監査や統計調査、IDS(侵入検知システム) との連携が考えられる。まず、脆弱性監査においてはノードの OS の情

報を収集することでリスク要因が抽出され、効率的な検査が可能になる。次に統計調査では、ネットワーク上のノードで稼働する OS の種類を調査し、そのネットワークでどの OS がどの程度使われているかといった統計調査が行える。IDS との連携では、IDS の精度向上を実現できる。IDS ではさまざまな種類の攻撃を検知できるが、攻撃手法のほとんどは特定の OS のみに特化した手法であるため、OS を考慮しないで警告を発する IDS では誤検出 (False positive) が多く含まれることになる。そこで各攻撃手法によって影響を受ける OS の情報を追加し、IDS が OS の情報と照合することで誤検出を低減し、警告情報の精度を向上できる [1]。例えば、Nimda Worm[14] によって影響を受ける OS は Microsoft Windows のみであることが事前にわかっていれば、Linux で稼働する Web サーバが Nimda Worm による攻撃を受けたとしても IDS が警告を行う必要は低い。

7.2 今後の課題

今後の課題としては、DHCP メッセージを元にフィンガープリンティングを行うため、固定 IP アドレス割当を受けたノードに対するフィンガープリンティングを行うことはできないという制約が挙げられる。そのため、固定 IP アドレスを持つノードに対しては他の手法を用いる必要がある。また、MAC アドレスの解決に Gratuitous ARP を利用しているため、Gratuitous ARP を行わないノードが存在した場合にはフィンガープリンティングの最終的な結果を得ることができない。

8 まとめ

本論文では DHCP を用いることで常に最新の IP アドレス割当情報を反映するフィンガープリンティングの手法を提案し、その手法を用いた実装を行った。この結果、DHCP を用いた動的 IP アドレス割当環境下において、IP アドレスと OS の対応付けに矛盾が生じることなくフィンガープリンティングの結果を得ることができた。

参考文献

[1] Burak Dayioğlu, Attila Özgit. “Use of Passive Network Mapping to Enhance Signature Quality of Misuse Network Intrusion Detection Systems”, Nov 2001

[2] Fyodor. “Remote OS detection via TCP/IP Stack FingerPrinting”, Phrack Magazine Volume 8, Issue 54, Dec 1998 http://www.insecure.org/nmap/nmap-fingerprinting-article.html

[3] Craig Smith, Peter Grundl, Subterrain Siphon Project. “Know Your Enemy: Passive Fingerprinting”, Sep 2001 http://project.honeynet.org/papers/finger/

[4] Fydor. ”Nmap - Free Stealth Port Scanner For Network Exploration & Security Audits”, http://www.insecure.org/nmap/

[5] Ofir Arkin, Fyodor Yarochkin.“X-probe”, http://www.sys-security.com/html/projects/X.html

[6] Subterrain Security Group. “siphon”, http://siphon.datanerds.net/, 2000

[7] Michal Zalewski, William Stearns. “p0f’, http://www.stearns.org/p0f/, 2000

[8] libpcap, http://www.tcpdump.org/

[9] W. Richard Stevens, 橘 康雄 訳, 井上 尚司 監訳. ”詳解 TCP/IP Vol.1 プロトコル”, ピアソン・エデュケーションジャパン, 2000

[10] R. Droms, “RFC2131: Dynamic Host Configuration Protocol”, IETF, Mar 1997

[11] S. Alexander, R. Droms “RFC2132: DHCP Options and BOOTP Vendor Extensions”, IETF, Mar 1997

[12] Microsoft, “Microsoft DHCP Vendor and User Classes”, http://support.microsoft.com/?kbid=266675, 2002

[13] IANA, “BOOTP AND DHCP PARAMETERS”, http://www.iana.org/assignments/bootp-dhcp-parameters, 2002

[14] CERT Coordination Center. “CERT Advisory CA-2001-26 Nimda Worm”, Sep 2001, http://www.cert.org/advisories/CA-2001-26.html