
**BRNO UNIVERSITY OF TECHNOLOGY**
**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**

**FACULTY OF INFORMATION TECHNOLOGY**
**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**

**DEPARTMENT OF INFORMATION SYSTEMS**
**ÚSTAV INFORMAČNÍCH SYSTÉMŮ**


# LAWFUL INTERCEPTION: IDENTITY DETECTION

**ZÁKONNÉ ODPOSLECHY: DETEKCE IDENTITY**

**PHD THESIS**
**DISERTAČNÍ PRÁCE**


**AUTHOR** — **Ing. LIBOR POLČÁK**
**AUTOR PRÁCE**

**SUPERVISOR** — **prof. Ing. MIROSLAV ŠVÉDA, CSc.**
**ŠKOLITEL**



**BRNO 2017**

# Abstract

Internet has became a regular communication channel between law offenders performing malicious activities. Courts or prosecutors authorise lawful interception that targets individual suspects. Lawful interception systems have to identify traffic of the suspects. However, the identifiers that appear in each network packet are short-lived, dynamically assigned, and a single communication session may be split into multiple flows identified by different identifiers. A lawful interception system has to immediately detect that a new identifier covered by an intercept appeared or disappeared. This thesis describes identification in modern computer networks compatible with lawful interception. The focus is on two partial identity detectors: IPv6 address assignment tracking based on monitoring of neighbor discovery and clock-skew-based remote computer identification. Additionally, this thesis proposes identity graphs that link partial identities according to optional constraints that reflect the wording of a warrant. Results of partial identity linking can be utilised by lawful interception systems. The results of this thesis are also applicable in network forensic, and in networks controlled according to user roles.

# Abstrakt

Komunikace předávaná skrze Internet zahrnuje komunikaci mezi pachateli těžké trestné činnosti. Státní zástupcí schvalují cílené zákonné odposlechy zaměřené na podezřelé z páchání trestné činnosti. Zákonné odposlechy se v počítačových sítích potýkají s mnoha překážkami. Identifikátory obsažené v každém paketu jsou koncovým stanicím přidělovány po omezenou dobu, nebo si je koncové stanice dokonce samy generují a automaticky mění. Tato dizertační práce se zabývá identifikačními metodami v počítačových sítích se zaměřením na metody kompatibilní se zákonnými odposlechy. Zkoumané metody musejí okamžitě detekovat použití nového identifikátoru spadajícího pod některý z odposlechů. Systém pro zákonné odposlechy následně nastaví sondy pro odposlech komunikace. Tato práce se převážně zabývá dvěma zdroji identifikačních informací: sledováním mechanismu pro objevování sousedů a detekcí identity počítače na základě přesností měření času jednotlivých počítačů. V rámci dizertačního výzkumu vznikly grafy identit, které umožňují spojování identit s ohledem na znění povolení k odposlechu. Výsledky výzkumu je možné aplikovat v rámci zákonných odposlechů, síťové forenzní analýzy i ve vysokoúrovňových programově řízených sítích.



# Keywords

Lawful interception, IPv6 address tracking, neighbor discovery tracking, clock-skew-based fingerprinting, partial identity linkage.

# Klíčová slova

Zákonné odposlechy, detekce přiřazení adres IPv6, dentifikace počítače pomocí odchylky měření času, spojování částečných identit.


# Reference

POLČÁK, Libor. *Lawful Interception: Identity Detection.* Brno, 2017. PhD thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Švéda Miroslav.

# Lawful Interception: Identity Detection

## Declaration

Hereby I declare that this PhD thesis was prepared as an original author's work under the supervision of prof. Ing. Miroslav Švéda, CSc. with expert supervision of Ing. Petr Matoušek, PhD, M.A. During the work on this PhD thesis I managed a group of students investigating lawful interception; I cooperated with Ing. Tomáš Martínek, PhD, and students of bachelor and master degree study program. Some tools and ideas were developed as a part of this cooperation under my supervision. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

.....
Libor Polčák
May 24, 2017

## Acknowledgements

This PhD thesis was supported by the project VG20102015022 (Modern Tools for Detection and Mitigation of Cyber Criminality on the New Generation Internet) supported by the Ministry of the Interior of the Czech Republic. This work was also supported by The Ministry of Education, Youth and Sports of the Czech Republic from (1) the National Programme of Sustainability (NPU II); project IT4Innovations excellence in science — LQ1602 and (2) the project TeamIT - Building Competitive Research Teams in IT — EE2.3.09.0067.

I would like to thank reviewers of my papers for interesting comments that moved my PhD research forward. The rest of the acknowledgment is written in Czech.

Děkuji mé ženě Markétě za lásku a podporu a ostatním členům rodiny za podporu při vypracování práce.

Děkuji mému školiteli Mirkovi Švédovi za konzultace k práci, zajímavé diskuze a pomoc s jazykem. Děkuji mému školiteli specialistovi Petrovi Matouškovi za vedení projektu Sec6Net, za dlouhé hodiny strávené při čtení práce, poznámky k práci a diskuze k práci. Děkuji kolegům Ondrovi Ryšavému, Matěji Grégrovi a Vladmimírovi Veselému za diskuze k práci a podnětné myšlenky. Děkuji ostatním členům skupiny NES@FIT za podporu.

Děkuji mým studentům za vypracované bakalářské a diplomové práce, především Báře Frankové, Martinu Holkovičovi, Radku Hranickému a Jakubovi Jiráskovi za přispění k mé publikační činnosti. Děkuji Tomovi Martínkovi za to, že přispěl k vedení studentů v rámci projektu Sec6Net.

Děkuji Honzovi Kořenkovi a dalším členům projektu Sec6Net za sondy, které na vysokých rychlostech využívají vytvořený nástroj systém pro zákonné odposlechy SLIS.

# Contents

# List of abbreviations

**CC** Content of Communication, page 10.

**CGN** Carrier Grade NAT, page 12.

**DAD** Duplicate Address Detection, page 9.

**DHCP** Dynamic Host Configuration Protocol, page 7.

**DHCPv6** Dynamic Host Configuration Protocol version 6, page 8.

**DUID** DHCPv6 Unique Identifier, page 9.

**ETSI** European Telecommunications Standards Institute, page 9.

**HE** Happy Eyeballs, page 14.

**IAN** Identity Aware Networks, page 6.

**IID** Interface Identifier, page 8.

**IP** Internet Protocol, page 5.

**IPv4** Internet Protocol version 4, page 5.

**IPv6** Internet Protocol version 6, page 6.

**IRI** Intercept Related Information, page 10.

**ISP** Internet Service Provider, page 5.

**IXP** Internet Exchange Point, page 12.

**LAN** Local Area Network, page 5.

**LEA** Law Enforcement Agency, page 9.

**LI** Lawful Interception, page 5.

**MLD** Multicast Listener Discovery, page 9.

**NA** Neighbor Advertisement, page 9.

**NAT** Network Address Translation, page 6.

**NC** Neighbor Cache, page 17.

**ND** Neighbor Discovery, page 8.

**NS** Neighbor Solicitation, page 9.

**NS-DAD** Neighbor Solicitation issued during DAD, page 9.

**pcf** PC Fingerprinter, page 25.

**RA** Router Advertisment, page 9.

**RS** Router Solicitation, page 9.

**SDN** Software-defined networking, page 7.

**Sec6Net** Modern Tools for Detection and Mitigation of Cyber Criminality on the New Generation Internet, page 7.

**SIMS** Sec6Net Identity Management System, page 7.

**SLAAC** Stateless Address Autoconfiguration, page 8.

**SLIS** Sec6Net Lawful Interception System, page 7.

**WAN** Wide Area Network, page 6.

# Chapter 1

# Introduction

Throughout the evolution of humanity, the identity of a particular person was strongly connected to its physical presence and appearance [84]. Each individual was typically identified by their physical features, voice and similar characteristics. Later, authorities introduced unique identifiers, such as passport numbers and personal ID numbers.

With the advent of information technologies, so-called digital and virtual identities emerged. As people employ digital and virtual identities remotely, these are not connected to the appearance. Typically, visitors are not required to validate their identity by official identifiers when they access services offered on the Internet. As a result, people create and manage a score of digital identities; some of them are connected to activities such as work, leisure, and social networking. Some of these identities are only loosely connected, and it is not obvious to a remote observer that all belong to the same person.

Computer networks present many novel means for communication of individuals. People communicate via e-mail, voice, or instant messaging applications. Not only are the computer networks used for benign activities but also malicious users, criminals, and terrorists use the same networks and protocols [9, 43, 45, 61]. To mitigate these threats, standards for *lawful interception* (LI), originally developed for telephone networks, were transformed for computer networks [24]. In the European Union, the European Council allowed LI in 1996 [94]. Later, the Council of Europe approved the *Convention on Cybercrime* [15]. The Czech Republic adopted the European law [63].

The aim of LI is to gather complete and indisputable evidence of criminal activities. In telephone networks, the subject of an interception (the intercept target) can be identified by his or her telephone number; both public switched telephone network operators and mobile carriers can accurately identify the subject and the data related to the intercept.

In computer networks, identification of an LI target is more complicated [18]. It is necessary to distinguish the traffic of the intercept target from the traffic of other users of the network. Typically, there is not a static unique identifier for each person that is available across networks.

A MAC address identifies a network interface of a computer on a *local area network* (LAN), the interception in the network of the *internet service provider* (ISP) has to rely on other identifiers.

Nowadays, *Internet Protocol* (IP) is the only protocol commonly used on the Internet to address hosts across network boundaries. Each computer carries one or more IP addresses. Since 1983, the dominant version of IP has been 4 (IPv4) [51]. However, its address space is limited and insufficient for today requirements [87]. The inevitable exhaustion of the IPv4 address space was evident in the early 1990s. Consequently, efforts to conserve the

address space emerged. *network address translation* (NAT) hides a LAN [85] or a *wide area network* (WAN) [101] behind a limited number of IPv4 addresses. Small offices and home networks usually get one public (globally unique) IPv4 address each. The deployment of the IPv4 successor — IP version 6 (IPv6) [19] is currently in the process [20]. Each IPv6 host can simultaneously use as many addresses as it can handle [68]. Hence, IPv6 addresses are dynamically configured, short-lived and often random.

This thesis investigates methods for computer and person identification in modern networks. The focus is on methods applicable in LI.

Nevertheless, the technical aspects are only one side of the coin. On the other, there are the ethical principles and the right to privacy. The Constitutional Order of the Czech Republic [13] acknowledges the right to privacy. Most of the other countries have similar guarantees. The contrast between the respect for privacy and the need to counter serious crime and terrorists raised a worldwide controversy [31, 99]. Although this thesis primarily focuses on technical aspects of LI, it also emphasises privacy-related questions, especially in the Section 2.3.

## 1.1 Research objectives

This PhD research is divided into the following areas:

1. Overview of LI standards and state-of-the-art of the identification in computer networks — during the research, we identified several challenges arising in modern computer networks [72, 80].

2. Methods for local and remote identification with the main focus on methods for identification in IPv6 networks — during the research, we proposed the IPv6 address assignment tracking deployable on IPv6 LANs [78, 79] and studied clock-skew-based remote computer identification and its applications [73, 74, 81].

3. Methods that link identity-related information of a single person or a computer together — we proposed identity graphs, a formal mechanism that links discovered identities that belong to a single subject [80].

4. Applications of the mechanisms studied in the research topics mentioned above — the methods are applicable in LI [80, 82] and *Identity Aware Networks* (IAN) [83].

The following research hypotheses arise for the second and third part of this PhD research:

**Hypothesis 1.** *The detection of IPv6 address assignments is possible from Neighbor Discovery traffic [69] even in networks with MLD snooping enabled (neighbor discovery traffic is multicasted rather than broadcasted).*

**Hypothesis 2.** *Short-term clock skew estimates can uniquely identify computers for LI.*

**Hypothesis 3.** *Identity information revealed at different locations can be linked by applying unambiguous rules.*

## 1.2 Achieved results

This PhD research was a part of the research project VG20102015022 *Modern Tools for Detection and Mitigation of Cyber Criminality on the New Generation Internet* (Sec6Net)[1] supported by the Ministry of the Interior of the Czech Republic. Based on the study of related work, challenges in modern and future networks were identified and presented at ISS Europe[2] 2013; and later published [72, 80]. The methods for identification were divided into two groups: local and remote.

Local methods are applicable on LANs. Usually, they monitor local traffic or require access to nodes deployed on the LAN. IPv6 provides a new decentralised mechanism that enables hosts to generate IPv6 addresses without any authority, such as *Dynamic Host Configuration Protocol* (DHCP) server. As a part of this PhD research, we analysed the behaviour of different operating systems and proposed IPv6 address assignment tracker deployable on LANs [46, 78, 79]. The IPv6 address assignment tracking translates IPv6 address management traffic to a sequence of symbols announcing the activity and inactivity of an IPv6 address. The output of the timed transducer is suitable for generating messages for LI. The proposed tracking fulfils Hypothesis 1.

Remote identification methods are usually less accurate than local methods. Nevertheless, their benefit is that they do not depend on the information available only in the network of the subject to be identified. Clock-skew-based identification [59] seemed to be a powerful tool for remote identification. However, its evaluation revealed limits [34, 54, 73, 81] of its applicability. Consequently, Hypothesis 2 was rejected.

As a part of the Sec6Net research project, other identification methods were studied, either by me or under my supervision. As a result, several modules for identity detection were developed [53, 82, 92]. These modules became a basis for a tool called *Sec6Net Identity Management System* (SIMS) [76]. SIMS gathers information about identities from several partial identity detectors and links them together.

The linkage of identities is based on the study of the identifiers used in computer networks and their relations. Based on this study, identity graphs of network identifiers were proposed [80] and implemented [49]. Identity graphs can link different identifiers based on conditions specified by a warrant for LI. This thesis extends the original identity graphs with more operations, including time-related operations and support for inaccurate partial identity detectors. Identity graphs confirm Hypothesis 3.

Identity graphs were applied in the LI system called *Sec6Net Lawful Interception System* (SLIS) [77] developed as a part of the Sec6Net project. In addition, *software-defined networking* (SDN) proved to be useful [47, 83] for prototyping other applications benefiting from the identification mechanisms developed during this PhD research.

## 1.3 Organisation of this thesis

Chapter 2 presents the terminology utilised in this thesis and essential background. Chapter 3 focuses on challenges in identification of LI suspects. Chapter 4 deals with identification in IPv6 LANs and Hypothesis 1. Chapter 5 evaluates remote clock-skew-based identification and Hypothesis 2. Chapter 6 proposes formal rules in conformance with Hypothesis 3. Chapter 7 concludes the thesis.

[1] https://www.fit.vutbr.cz/~ipolcak/grants.php?id=517

[2] Intelligence Support Systems for Lawful Interception, Electronic Surveillance and Cyber Intelligence Gathering, http://www.issworldtraining.com/ISS_EUROPE/

# Chapter 2

# Essential background

This chapter provides the essential background and terminology utilised in this thesis. Section 2.1 focuses on identity theory and partial identities. Section 2.2 provides essential background for identification in IPv6 networks. Section 2.3 establishes the terminology used in LI.

## 2.1 Partial identities

Pfitzmann and Hansen [71] define an identity as a subset of features that sufficiently distinguishes a subject of the identification (for example, a person or a computer) from all other subjects. A *partial identity* represents the subject in a specific context or a role.

If a group of subjects is not identifiable within the system, the set of indistinguishable subjects is called the *anonymity set* and the subjects are *anonymous* within the anonymity set.

## 2.2 Internet Protocol version 6

As the FIDIS report [32, Chapter 2] notes, IP addresses identify either end hosts, or, in the case of NAT, the translator. This thesis focuses on identification methods related to IPv6. Sections 3.2 and 3.3 describe challenges in IPv6 and dual stack networks. Chapter 4 proposes the IPv6 addresses assignment detection. This section provides the necessary background.

### 2.2.1 IPv6 addressing

IPv6 introduces several new mechanisms for address assignments. For example, *Stateless Address Autoconfiguration* (SLAAC) [96] is mandatory for all IPv6-enabled nodes in the network. SLAAC is a part of IPv6 *Neighbor Discovery* (ND). SLAAC allows an end device to generate as many IPv6 addresses as it needs, for example, for privacy concerns [4, 37, 68], as long as another device on the LAN does not use any of the addresses. The addresses are not leased but generated by end devices without any registration of the generated IPv6 address. The device concatenates the network part of the address, learnt from a router, with an auto-generated, usually 64-bits-long, *interface identifier* (IID) [16].

Besides SLAAC, *Dynamic Host Configuration Protocol version 6* (DHCPv6) exists in IPv6 networks. However, the support for DHCPv6 is optional. In addition, even when

active, the presence of the DHCPv6 server has to be signalled through ND messages and the leased IPv6 addresses have to be validated by ND.

In DHCPv6-controlled IPv6 networks, a *DHCPv6 Unique Identifier* (DUID) identifies a specific machine. Whenever a computer leases an IPv6 address from a DHCPv6 server, the computer identifies itself with the DUID. As a consequence, the DHCPv6 server in the network can link leases supplied to different interfaces of the same computer, for example, wired and wireless.

### 2.2.2 Neighbor Discovery

For the understanding of the contribution of this thesis in the IPv6 identification realm, it is necessary to get familiar with the ND process.

ND [69] allows a node to discover routers and other hosts in the network. In addition, the node also learns prefixes in use in the network and other information.

ND depends on the multicast groups created on the LAN. There are role-specific groups, for example, all nodes (*ff02::1*), all routers (*ff02::2*). ND also depends on *solicited-node multicast groups* that are created automatically whenever a node generates a new IID. Switches on the LAN can treat multicast messages as broadcast or build multicast trees in case that they are able to perform *Multicast Listener Discovery* (MLD) snooping [11].

When a host connects to a network, it tries to discover routers in the network by issuing *Router Solicitation* (RS) requests. Routers in the network reply by a *Router Advertisment* (RA) [69] message; either to the all host multicast group or by a unicast reply. RA includes one or more on-link prefixes and zero or more SLAAC prefixes configured in the network. Optionally, the RA instructs the host to get additional configuration from a DHCPv6 server. The host constructs IPv6 addresses from the advertised SLAAC prefixes and generated IIDs [96].

Before a newly generated tentative address can be used for communication, it is necessary to test its uniqueness in the network. *Duplicate Address Detection* (DAD) [66, 96] is a process during which the host sends a *Neighbor Solicitation* (NS) message to the special solicited-node multicast group [44] corresponding to the tentative address. In this thesis, NS issued during DAD is denoted as NS-DAD. If the tentative address is already configured on another host, the another host replies with a *Neighbor Advertisement* (NA) to the multicast group for all nodes in the network (*ff02::1*). If the tentative address is not already used by another device, there is no reply to the NS-DAD. The non-conflicting address changes state to either the preferred state or the deprecated state. The host can use the address for communication. To avoid race conditions in address assignments, RFC 4862 [96] orders that each host has to join the solicited-node multicast group before it sends the NS-DAD.

MLD capable routers periodically query all multicast groups for active subscribers. If any subscriber exists, one of them replies. In case that all IIDs containing the lowest 3 bytes of the solicited-node multicast group have been dropped, there is no MLD reply to the query. However, in the case of a collision in solicited-node multicast addresses, the group stays active while at least one of the colliding IIDs is active.

## 2.3 Basics of lawful interception

*European Telecommunications Standards Institute* (ETSI) has standardised LI [22–30] for the European Union. Figure 2.1 depicts high-level architecture for LI. A warrant authorised by court instructs an ISP to intercept data for a particular *Law Enforcement Agency* (LEA).

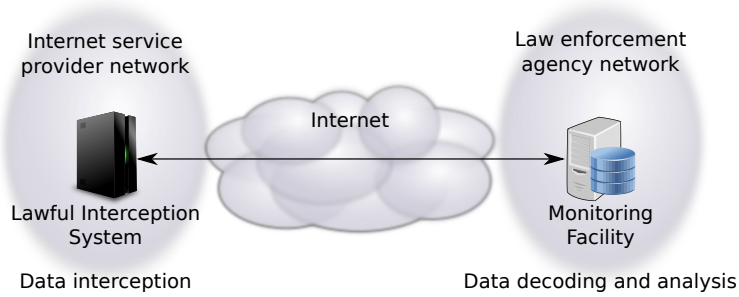



Figure 2.1: Top level architecture of an LI system: intercepted data are passed to the LEA for further analysis.

The warrant authorising an interception may require either interception of *intercept related information* (IRI) (metadata about the communication) or both IRI and *content of communication* (CC) (a copy of the communication of the target of the intercept — the suspect).

An LI system generates IRI records based on the observed traffic that the intercept target transmits or receives. Before a session is initiated, IRI *report* is used to signal any information, such as session establishment parameters. Immediately after a session is opened, the LI system signals to LEA session parameters in IRI *begin*. Any additional information about the running session is sent as IRI *continue*. The LI system signals the end of the session by IRI *end*. Should any additional information arise, the LI system generates IRI *report*. In summary, the following regular expression denotes IRI records that are generated for a single communication session: $report^* \; begin \; continue^* \; end \; report^*$.

Identification of an intercept target has to be unambiguous [27]. ETSI does not provide a fixed list of identifiers to be used for identification. An interception warrant lists either (1) directly network-related identifiers that are the object of the interception or (2) other unique identification of the suspect by another identifier, for example, by his or her name and home address. In the latter case, authorised personnel of the ISP has to determine unique network-related identifiers of the suspect [28]. Identifiers related to the partial identity of the suspect can change over time. An LI system has to link configured stable identifiers with temporary identifiers used in the network.

The FIDIS report [32] manifests that metadata, such as protocol headers, associated with current protocols leak many identifiers that can be directly or indirectly linked to specific persons. Moreover, cross-layer linkability reveals additional information about identities of specific devices or persons. However, even though the report mentions cross-layer linkability, FIDIS did not study the cross-layer linkability in detail. One of the goals of this thesis is to link identifiers found in network protocols and at the same time link the partial identities identified by the identifiers. Chapter 6 describes identity graphs that allow cross-layer linkability based on identifiers that appear in all layers of the TCP/IP model.

# Chapter 3

# Challenges of identity detection in modern computer networks

Computer networks connect various computers together. User behaviour and typical usage of network change with rising number of interconnected devices [42]. Communication infrastructures became distributed, highly complex and service-oriented [97]. Consequently, network protocols evolve. With new protocols, new challenges for identity detection emerge.

In the past, a typical household owned a single desktop computer. The advent of laptops, smartphones, tablets, wearables and other network-enabled small devices increased the number of devices operated in a single household. Nowadays, shared computers are less common. Often, a person exclusively uses several devices that access the Internet and browse web pages.

As a result, the knowledge that a device produced some traffic is often sufficient to link the traffic to the person that exclusively uses the device. However, as the person typically owns several devices, the traffic of a single identified device represents only a fraction of the traffic produced by the owner. Hence, the identification of a particular device represents only a partial identity of its owner.

IP addresses are prominent device identifiers [9]. However, one of the biggest problems in networking in the last years is the depletion of the IPv4 address space [87, 95]. Nevertheless, IPv4 is still dominant protocol used on the Internet [57]. The shortage of IP addresses is countered by NAT, which typically hides many devices behind a single IP address or a pool of IP addresses. The share of IPv6 traffic is increasing. As already reported [41], the identity detection in IPv6 is different compared to IPv4.

This chapter lists challenges of user identification in current networks with a focus on LI; each section represents one challenge. This chapter (and thesis) considers both IPv4 and IPv6. The final section of this chapter explains the relation of this thesis to the challenges listed in this chapter.

This chapter is based on the ideas presented in papers *On Identities in Modern Networks* [80] and *Challenges in Identification in Future Computer Networks* [72]. I am the author of the text in both papers.

## 3.1 Network address translation

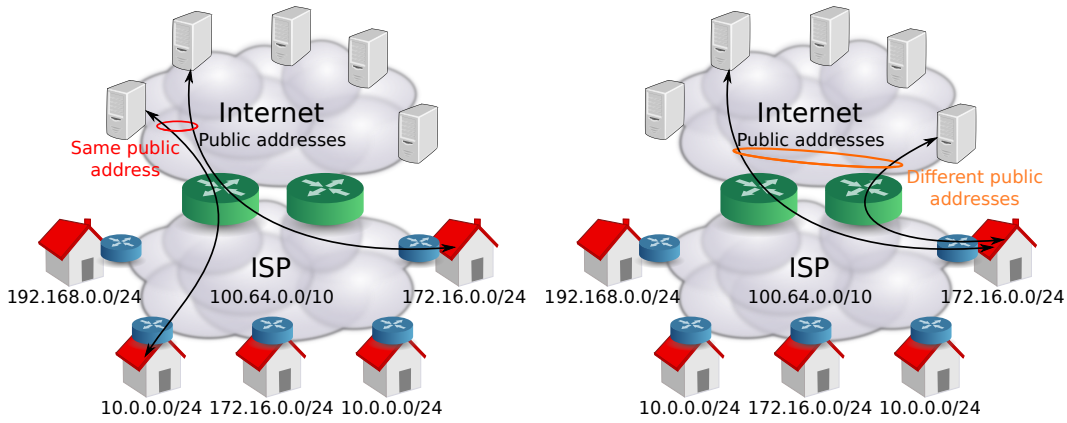
The first challenge for identity detection in current networks concerns NAT. A typical household customer buys a connection to the Internet from a local ISP present in the

area of the household. The ISP is typically local or nationwide. An ISP is connected to the Internet either at an *Internet Exchange Point* (IXP)[1] or through another transit ISP (typically small ISPs).

The ISP typically forms a contract with one representative of the household who becomes (de jure) the customer. However, for LI, it is important [3] to identify the traffic of a specific member of the household as the person who signed the contract does not have criminal liability for the content of the traffic produced by other members of the household.

Until recently, a typical household received one public IPv4 address. All members of the household shared the IPv4 address as the home network was connected to the ISP with a router providing NAT. NAT maintains a table that defines a bijection between local and global identifiers of the flows traversing the network boundary. Hence, all traffic of the household used to be identified by a single global IP address. However, the depletion of the IPv4 address space does not allow ISPs to assign a public IPv4 address to all customers (if the ISP does not have enough IPv4 addresses from the past). As a result, ISPs employ a second layer of NAT for the traffic destined to the Internet. The two-layer network address translation is called *Carrier Grade NAT* (CGN).

Figure 3.1 shows the challenges in identification of traffic origin arising in a CGN-enabled network. In the first example, users from two different households try to reach the external servers at the same time. It is possible that CGN translates both requests to a single public IPv4 address. The second example shows that the traffic of a single household can be identified with different public IPv4 addresses. In this case, the traffic is routed through distinct translators that do not share the same public address space. Note that the exact behaviour of CGN in both cases is not specified as it depends on the network configuration and state.




a) Traffic of two household can share the same public IPv4 address.

b) Traffic of a single household is not identified with a single IPv4 address.

Figure 3.1: Challenges in networks with CGN.

LI warrants unambiguously specify the traffic to be intercepted. Depending on the exact wording of a warrant, it can be legally forbidden to capture all traffic of a single IP address when the court gives permission to intercept traffic of a single person, and the person is connected behind NAT. Therefore, LI performed in a CGN network needs to take the multiple address translations into account.

[1] For example, http://nix.cz

## 3.2 Addresses in IPv6 Networks

IPv6 penetration in the Czech Republic is one of the highest in the world [33]. Thus, it is necessary that an LI system deployed in the Czech Republic identify IPv6 users. This section focuses on IPv6 addresses and the IPv6 address space. Section 3.3 outlines challenges related to the coexistence of IPv4 and IPv6.

The IPv6 address space is much larger compared with the IPv4 address space. In contrast to IPv4 where a computer network interface is usually identified by a single IPv4 address, in IPv6, each interface typically uses several IPv6 addresses [12].

Figure 3.2 shows an example of a single computer that generates several IIDs for a single interface. Typically, the IID of the link-local address of a specific interface is *stable* [14, Section 4]. However, the globally routable addresses can be temporal [68]. Current operating systems (Windows, Mac OS X, iOS, user-friendly Linux distributions) generate both stable addresses and temporary addresses. Temporary addresses are random and periodically regenerated. Windows, Mac OS X, iOS, and Ubuntu generate a new temporary address daily. Additionally, the Wi-Fi network reauthorization or a network cable disconnect and reconnect often triggers regeneration of temporary addresses.

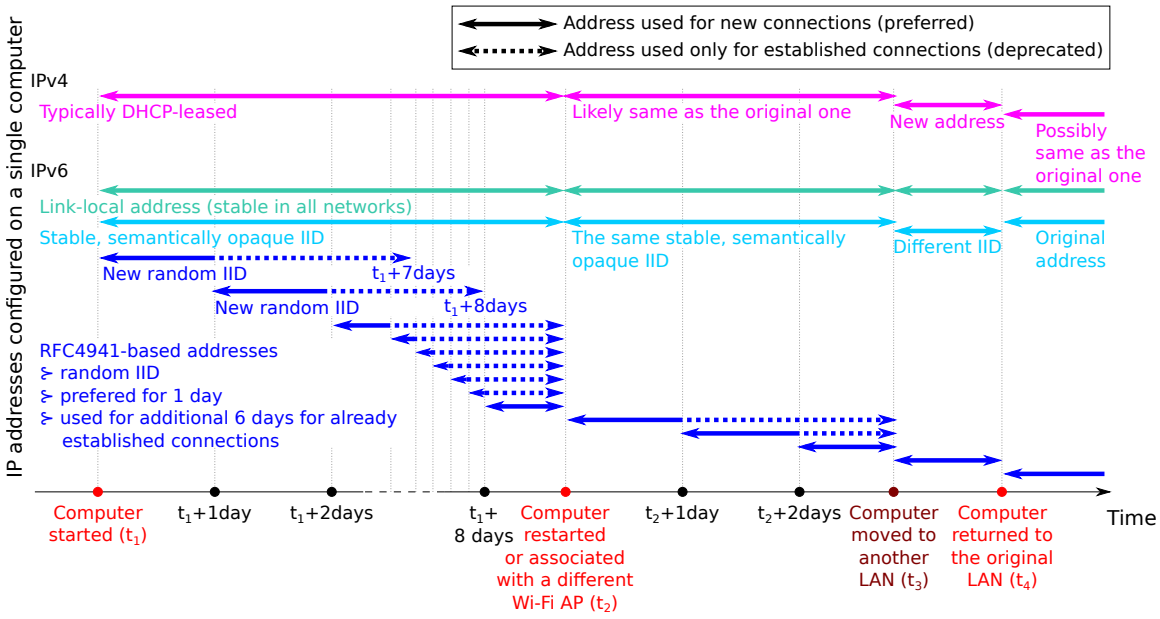



Figure 3.2: A single network interface usually operates several IPv6 IIDs while it uses only one IPv4 address.

As Figure 3.2 shows, IPv6 address assignment tracking needs to take into account that a single network interface can operate many IPv6 addresses at the same time, possibly tens of IPv6 addresses [12]. Additionally, the IPv6 addresses associated with the interface change over time. Hence, IPv6 address assignment tracking should incorporate time.

In principle, the IPv6 LI is similar to IPv4 LI [10, Section 4.3]. The absence of NAT simplifies transport layer flows processing — two flows originating from the same IPv6 address at the same time belong to the same interface. Hence, it is straightforward to intercept data of the local computer (identified by one IPv6 address) [10, Section 4.3]. However, as a single interface can be configured with multiple IPv6 addresses that change

over time, the challenge lays in the identification of all IPv6 addresses belonging to the same interface over time.

## 3.3 Dual-stack networks

Today, most of the web domain names resolve to IPv4 addresses only [1]. Therefore, most IPv6-enabled networks also support IPv4 to provide connectivity to IPv4 servers (typically through NAT or CGN). The simultaneous support for IPv4 and IPv6 is usually called dual stack. Hosts supporting both IPv4 and IPv6 create another challenge for LI: it is necessary to intercept traffic of both protocols.

Originally, IPv6 was preferred and only when the IPv6 connection did not succeed for several seconds, the host switched to IPv4. Later, *Happy Eyeballs* (HE) [102] allowed smooth fallback to IPv4. HE can seamlessly switch between IPv4 and IPv6 without any user interaction. Therefore, a part of the content fetched from a single web server can be downloaded via IPv4 and the rest via IPv6. The multiplexing introduces an additional need for linking of IPv4 and IPv6 partial identities.

In addition, without HE, dual-stacked machines prefer IPv6. Hence, IPv6-enabled servers are accessed via IPv6 while for IPv4-only servers the host falls back to IPv4. As web pages often contain external content and DNS is accessed separately, one session may be carried over both IPv4 and IPv6 even without HE.

## 3.4 Application layer protocols

LI standards include support of application layer protocols [29, 30]. The recent advent of smartphones introduces new proprietary protocols for instant messaging, voice communication, and other forms of communication. Obviously, these applications can be misused by criminals. Therefore, a modern LI system and ISS has to allow support for new application layer protocols.

Typical mobile applications use a proprietary application layer protocol or a customised version of a standardised protocol [2, 5, 91]. Some applications use a telephone number as the user identifier, other applications use custom nicknames or other identifiers. Hence, the support of current mobile instant messaging requires the ability to support different protocols and different identifiers.

## 3.5 Legal requirements

Based on the discussions with the Czech law enforcement agents, wordings of LI warrants vary. Some warrants allow interception of data identified by a specific IP address only. Other warrants demand interception of all traffic of a particular computer or a particular user. Some warrants allow linking identities whereas other warrants do not.

Another legal requirement concerns the covert nature of LI. The goal is to gather evidence for a court trial. Consequently, the interception has to be performed without the knowledge of the intercept target. Therefore, the location of the LI system has to take into account that is is not always possible to deploy its components into the most suitable location, such as the LAN where the intercept target is connected.

Another legal requirement is to differentiate between interception of IRI and CC [22]. For example [45, page 17], IRI intercepts allow monitoring of a suspect, linking his or her

identity, and learning the communication parties of the suspect. An interception of the whole communication is legal only when the warrant allows interception of CC.

## 3.6 The challenges and this thesis

As obvious from previous sections of this chapter, there are many challenges in LI in today networks. As Torres et al. [97] observed, an LI system of the future needs to process many partial identity detectors. Consequently, Chapter 6 proposes a highly distributed mechanism.

Figure 3.3 shows the problem decomposition employed by this thesis. The decomposition provides a basis for a modular, distributed and extensible IMS built from independent parts. Several partial identity detectors provide identity-related information from which an identity graph is automatically constructed. Each partial identity detector can be based on an independent mechanism. Chapters 4 and 5 describe two distinct partial identity detectors. Chapter 6 focuses on identity graphs that allow partial identity linking.

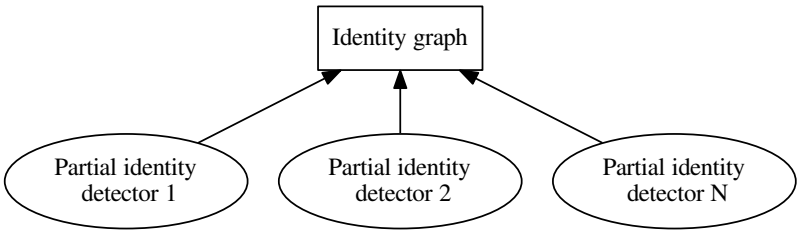



Figure 3.3: The proposed IMS is modular, extensible and it can be distributed.

The architecture depicted in Figure 3.3 tackles the challenges listed in this chapter by specific partial identity detectors focused on a specific source of identification. Each partial identity detector can be deployed in the most suitable location in the network. Consequently, the modular and extensible architecture is suitable to deal with different application protocols (see Section 3.4) as creating a new module does not require changes in other partial identity detectors.

However, it is not always possible to use a specific mechanism. For example, consider a court order to intercept data of a specific person. While it might be feasible to deploy the IPv6 address assignment tracking described in Chapter 4 to identify the intercept target in some locations, such as an international company, it is not possible to deploy the tracking in the home network of the subject as the interception would not be performed covertly — the subject is likely to recognize that an LI system was deployed in his house, see also Section 3.5. Hence, this thesis also considers remote identification.

To address the legal requirements that are presented in Section 3.5, this thesis focuses on both the local and remote identification. Figure 3.4 depicts the possible deployment of partial identity detectors. The benefits of local and remote detectors follow:

(a) User identification on LANs (see Figure 3.4 - a): The monitoring node is located on the same LAN as computers that are to be identified and intercepted. Hence, the learnt information is usually sound and complete.

(b) User identification outside the LAN of the intercept subject (see Figure 3.4 - b): Lately, users connect to the Internet using several Internet providers simultaneously (for example, using Wi-Fi and mobile carrier networks). Sometimes it is necessary to monitor

the movement of intercept subjects that use different networks, such as public WiFi hotspots, hotel WiFi. As the deployment of an LI system in every network is not legally and technically possible, the remote monitoring is beneficial. The downside of remote identification is that some identifiers, such as MAC addresses, are not available and the identification may not be accurate.

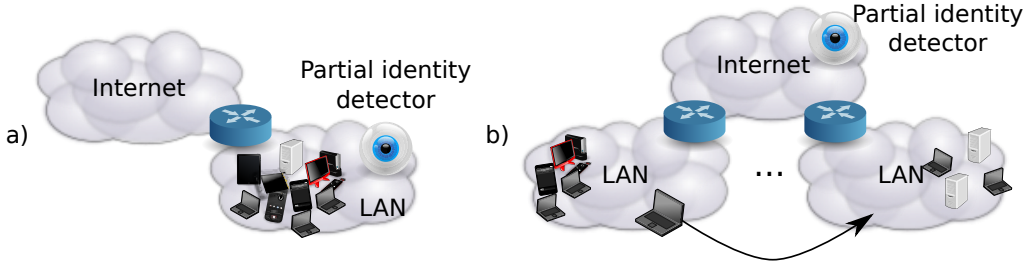



Figure 3.4: Local (a) and remote (b) deployment of partial identity detectors.

This thesis focuses in detail on two partial identity detectors:

1. Chapter 4 focuses on a local partial identity detection — IPv6 address assignment tracking on LANs. The proposed approach monitors ND messages on a LAN. The detection is passive for end devices and does not require any software or hardware modifications of switches and routers in the network. The method is suitable to detect all IPv6 addresses configured on a LAN, see Section 3.2.

2. The Clock-skew-based identification method studied in Chapter 5 provides remote identification. The method was reported to be fast [93] and handled passively [59]. During this PhD research, I extended the method to link all IPv4 and IPv6 addresses of a computer based on clock skew. Hence, the method addresses the challenges described in Sections 3.2 and 3.3.

Besides these two partial identity detectors, Chapter 6 discusses additional partial identity detectors developed as a part of the Sec6Net project under my supervision [82].

Chapter 6 introduces identity graphs, a graph model that stores detected identifiers. Identity graphs link detected identifiers based on formally defined rules. The goal is to link identifiers on a warrant with dynamic identifiers that are the most suitable for identification of data to be intercepted, such as IP addresses [45, page 272]. Identity graphs support several types of linking constraints that were defined based on the possible network topologies and warrants wording. Identity graphs support multiple IP addresses of a single computer and the rules for identity graphs construction support NAT. Hence, Identity graphs address challenges described in Sections 3.1–3.5.

The methods presented in this thesis in Chapters 4 and 5 were included into SLIS [77, 82] developed as a part of the Sec6Net project.

# Chapter 4

# Identification on IPv6 LANs

This chapter focuses on the IPv6 address learning mechanism deployable on a LAN that is the main contribution of this thesis for accepting Hypothesis 1. The goal of the mechanism is to identify IPv6 and MAC address bindings on LANs, and consequently, identify all IPv6 addresses in use on all interfaces on a LAN. The mechanism can be deployed as a standalone solution that provides similar information to DHCP logs in IPv4, or, it can be utilised as one of the partial identity detectors for identity graphs described in Chapter 6.

The proposed IPv6 address assignment tracking mechanism examines the messages exchanged during ND and tracks the state of the discovered IPv6 addresses. For each IPv6 address, the mechanism learns the MAC address associated with the interface that uses the IPv6 address. The core of the mechanism is formally described in the full text of this thesis as a timed transducer. The timed transducer executes transitions based on input symbols and time out events. The input symbols are constructed from messages received from the LAN; each symbol corresponds to a single message. The output of the transducer provides the address management information compatible with IRI *begin* and *end* records described in Section 2.3.

Firstly, we performed a study of current operating systems behaviour [75]. The study confirmed our expectations about several inconsistencies between ND implementations in different operating systems. Section 4.2 summarises the results of the study.

Based on the study, we proposed [78] an extended finite state machine. The original paper [78] was selected for an extended version that was published as *Host Identity Detection in IPv6 Networks* [79]. I was the main author of the papers; I wrote the text, the ND study was conducted under my supervision. In this thesis, I transformed the vague description of the extended finite state machine and defined the mechanism as a timed transducer. The timed transducer is defined in Section 4.3. As a part of the Sec6Net project, under my supervision, we developed a tool called *ndtrack* [48] that implements the IPv6 address assignment tracking mechanism.

## 4.1 Related work in IPv6 identification

Section 2.2 introduces IPv6 addressing, IIDs, and ND including SLAAC. This section provides an overview of methods for local identification in IPv6 networks.

All IPv6-enabled hosts maintain a *neighbor cache* (NC). NC is a table that stores the bindings of IPv6 and MAC addresses of computers, with which the host has active or recently finished communication on a LAN. Grégr et al. [41] poll routers in the Brno Uni-

versity of Technology network and download NC of the routers in the University network to learn the IPv6 addresses of connected end hosts in the network. However, the polling increases the workload carried by the routers. Hence, the polling cannot be too frequent as it would result in negative performance impact. The method described in this chapter (1) detects a new address immediately, including addresses that are utilised for intra-LAN communication only, (2) does not poll routers or switches and thus does not increase their load periodically, and (3) detects that addresses are released by end hosts even during the time when the addresses are still present in the NC of a router.

Groat et al. [38] studied DHCPv6 for monitoring the identity of users on LANs. They focused on possibilities of an adversary that has access to several IPv6 LANs. Nevertheless, Groat et al. [38] focus only on one particular address assignment method — DHCPv6. SLAAC is the default address assignment method in IPv6 whereas DHCPv6 is optional. A module for DHCPv6 lease tracking was developed as a part of the Sec6Net project [82]. The module provides information required to build identity graphs.

Sanguanpong et al. [89] implemented a captive portal for dual-stacked networks. The authentication web page embeds two images, one accessible through IPv4 and one through IPv6. Both images are identified with a unique hash. Consequently, the web server links the IPv4 address and an IPv6 address by the hash supplied HTTP requests that download the images. The web-based dual stack address discovery reveals only the address that is preferred for the communication with the web server at the time of the access to the captive portal. Consequently, the method is not suitable for LI since LI needs to cover all traffic of the suspect.

The messages exchanged during ND can be monitored by *addrwatch* [60]. Although *addrwatch* displays ND messages captured from the network, it does not display the state of the IPv6 addresses. Hence, by looking at the output, it is not clear what interface has currently assigned an IPv6 address $A$ and if it is the same interface that had the IPv6 address $A$ assigned at a particular time. Additionally, *addrwatch* does not track the lifetime of IPv6 addresses. Consequently, *addrwatch* does not display any line when an IPv6 address expires. The timed transducer defined in Section 4.3 also tracks ND messages. In contrast to *addrwatch*, the timed transducer keeps information about the state of an IPv6 address. Additionally, the timed transitions allow the timed transducer to automatically track the tentative and valid period of an IPv6 address.

An active adversary can *ping* multicast groups such as the all host multicast group *ff02::1*. However, some operating systems do not reply to these *ping* requests [36]. An alternative solution is to send invalid or unspecified options in the packets [36]. Nevertheless, each host typically replies from a link-local address. The IPv6 address assignment tracking described in this chapter is passive for monitored devices and detects all IPv6 addresses of the hosts.

## 4.2 Study of Neighbor Discovery implementations

This section describes the implementation of ND in different operating systems. The study [75] focuses on Windows, Linux, Mac OS X, FreeBSD, OpenBSD, and Solaris. The study of ND implementations in current operating systems reveals that there are some differences. Nevertheless, it is possible to provide a mechanism that is compatible with the majority of operating systems.

An IPv6 address can be used for communication in the *preferred* and the *deprecated* state. A monitoring device has to detect the transitions between (1) the *tentative* and

the *preferred* state, (2) the *preferred* and the *invalid* state, and (3) the *deprecated* and the *invalid* state. These transitions change the availability of the address for communication of the monitored host.

Linux hosts follow the standard sequence [96] most closely. A monitoring of IPv6 address management traffic of Linux hosts can detect:

1. A change between the *invalid* and the *tentative* state by observing MLD reports joining a *solicited-node multicast group*, and the following NS-DAD in the *solicited-node multicast group*.

2. A change between the *tentative* and the *preferred* state when there is no reply for the NS-DAD for a period defined by *RetransTimer* [69].

3. A change between the *deprecated* and the *invalid* state can be computed from the valid lifetime contained in RAs [69]. Alternatively, in the presence of an MLD querier, the address is invalid when there is no reply for an MLD query to the *solicited-node multicast group*.

4. The change between the *preferred* and the *deprecated* state can be computed from the preferred lifetime contained in RAs [69]. However, this is not necessary for the timed transducer proposed in Section 4.3 as the address can be used for communication in both states.

The same messages can be observed by monitoring Apple Mac OS X. Additionally, a change from the *tentative* to the *preferred* state of a Mac OS X host can be detected from an unsolicited NA issued during DAD.

FreeBSD follows the standard as well, so it is possible to detect the transitions between (1) the *invalid* and the *tentative* state, (2) the *tentative* and *preferred* state, and (3) the *deprecated* and *invalid* state. The only exception is a static address for which it may not be possible to detect the change between the *invalid* and the *tentative* state as FreeBSD sends NS-DAD before it joins the *solicited-node multicast group*.

Windows Server 2008 R2, Windows 7 and later send an NS-DAD before they subscribe to the *solicited-node multicast group* for the address. However, it is possible to detect the change from the *tentative* to the *preferred* state from the additional unsolicited NAs.

Solaris sends NS-DAD before it subscribes to the *solicited-node multicast group*. Hence, the change from the *invalid* state to the *tentative* state may not be detected. However, as it issues unsolicited NAs during DAD, a monitoring node can detect the change from the *tentative* state to the *preferred* state.

OpenBSD does not join *solicited-node multicast groups*. Therefore, it is not possible to observe NS-DADs in its tentative phase unless they are broadcasted by the network (no MLD snooping). We decided that the market share of the operating system is too low and the timed transducer proposed in Section 4.3 does not detect OpenBSD in networks with MLD snooping.

As expected, all operating systems performed DAD when they leased a new IPv6 address from a DHCPv6 server. However, the behaviour diverged during the lease refresh:

- Linux distributions did not perform DAD during lease refresh.

- Solaris repeated DAD while it resubscribed to the solicited-node multicast group corresponding to the leased address during each lease refresh.

- All versions of Windows repeated DAD during each lease refresh, but they did not explicitly resubscribe to the solicited-node multicast group corresponding to the leased address.

RFCs do not mandate the repeated DADs. Hence, all tested operating systems behaved in conformance with the expectations and followed the advice by RFC 3315 [8, Paragraph 18.1.8] and requirement in RFC 4862 [96, Subsection 5.4]. Consequently, the IPv6 address assignment tracking proposed 4.3 tracks also DHCPv6-leased IPv6 addresses.

As a result, we decided to construct timed transducer. The input symbols of the timed transducer represent IPv6 address management messages discussed in this section. The output symbols of the timed transducer contain information about the assignments of an IPv6 address. The goal is to detect the change from the *invalid* state to the *tentative* state, and consequently, the change from *tentative* state to the *preferred* state as described above for a Linux host. Additionally, the proposed IPv6 address assignment tracking based on timed transducer utilises unsolicited NAs to detect that an unobserved address shifted to the *preferred* state. The timed transducer should delay transitions from both the *preferred* and the *deprecated* states to the *invalid* state as we observed that some replies to MLD queries arrived after 0.7–0.8 seconds instead of the advertised 0.1 second,

## 4.3 Proposed IPv6 address assignments tracking

Based on the study of the DAD implementation in different operating systems described in Section 4.2, we proposed [78, 79] and implemented [48] the address assignment tracking on IPv6 LANs.

This section overviews the tracking mechanism. The full definition of the timed transducer is specified in the full text of this thesis.

A single proposed timed transducer tracks a single IPv6 address $A$. Informally, the proposed timed transducer stores the MAC address identifying the interface to which IPv6 address $A$ is being assigned in the states of the timed transducer. At the input, the proposed timed transducer reads symbols constructed from NS-DAD messages related to the IPv6 address $A$, NA messages of $A$, MLD queries for the solicited-node multicast group corresponding to $A$, and MLD reports (replies) for the queries. The timed transducer outputs which MAC address is linked to the IPv6 address $A$ and which MAC address is no longer linked to the IPv6 address $A$.

As already mentioned, a single timed transducer described in this section tracks only a single IPv6 address $A$. To track all IPv6 addresses in the network, one timed transducer is required for each IPv6 address. The need for a vast number of automata is reduced in practice. It is sufficient to store and track only addresses that are not in the initial state. Hence, there is not any substantial memory overhead.

The proposed timed transducer is executed on a monitoring node as depicted in Figure 4.1. The monitoring node is connected to any switch in a network as any other host. To ensure that the multicast ND traffic is delivered to the monitoring node, the node has to join the multicast group for all nodes (*ff02::1*), all MLDv2-capable routers (*ff02::16*), and all solicited-node multicast groups detected by analysing MLD reports in the multicast group for all MLDv2-capable routers. A symbol converter transforms the IPv6 address management messages in these multicast groups to the input symbols of the timed transducer. The proposed timed transducer creates the output that is available for processing by other software, for example, to construct identity graphs, see Chapter 6.

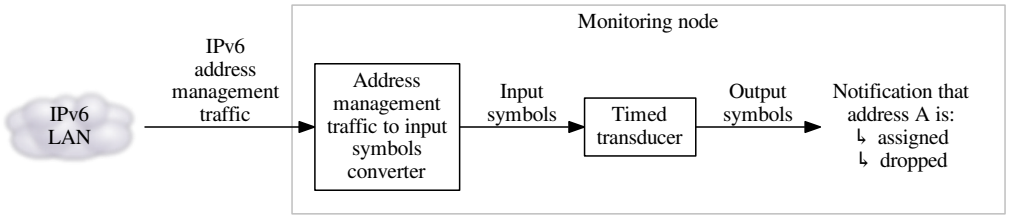



Figure 4.1: Monitoring node observes address management traffic in the network. Besides the proposed timed transducer, the IPv6 address assignment tracking incorporates a converter that transforms network messages to the input symbols of the proposed timed transducer.

A monitoring node in a network cannot accurately detect all events and state changes happening on network nodes [7]. Therefore, the proposed timed transducer focuses on detecting substantial states of address assignments. Specifically, it detects that an IPv6 address (1) is not used in the network, (2) is in the tentative state, or, (3) is considered preferred or valid by a host in the network.

## 4.4 Evaluation of the IPv6 address assignment tracking

This section describes the experiments with *ndtrack* [48], a tool that follows the proposed approach for IPv6 address assignment tracking.

### 4.4.1 A test in a network with MLD Snooping of solicited-node multicast groups

Firstly, we validated the behaviour of *ndtrack* in a laboratory. We deployed a network with active MLD snooping of the solicited-node multicast groups. There was *ndtrack* running on a monitoring node in the network. Additionally, a testing computer was connected to the switch with MLD snooping enabled.

In the first set of experiments, *ndtrack* did not join detected solicited-node multicast groups. In the second set of experiments *ndtrack* joined the multicast groups as described in Section 4.3. Each set of experiments tested several operating systems.

Table 4.1 summarises the results of the experiments. When *ndtrack* did not join the solicited-node multicast groups, the switch did not propagate NS-DADs to the monitoring node. As a result, *ndtrack* did not identify computers that follow the recommended sequence of messages during the DAD. Windows 8, Mac OS X, and Solaris send additional NAs (as revealed in Section 4.2). The additional NAs allowed *ndtrack* to discover the address assignments even if it did not join the solicited-node multicast groups. When *ndtrack* joined the specified multicast groups during the DAD launched by the tested computer, *ndtrack* successfully detected all operating systems except OpenBSD and static addresses in FreeBSD as expected.

OpenBSD was not detected as expected by Section 4.2. It does not join the solicited-node multicast group corresponding to the tentative address. Consequently, *ndtrack* did not join the solicited-node multicast group. As a result, the switch with activated MLD snooping did not propagate the NS-DADs to the monitoring node.

Table 4.1: Effectivity of *ndtrack* in networks with active MLD snooping of solicited-node multicast groups (✓means detected).

| Monitoring node joined multicast groups | Static addresses | | SLAAC addresses | |
|---|---|---|---|---|
| | No | Yes | No | Yes |
| Windows 7 and earlier | - | ✓ | - | ✓ |
| Windows 8 | ✓ | ✓ | ✓ | ✓ |
| Linux | - | ✓ | - | ✓ |
| Mac OS X | ✓ | ✓ | ✓ | ✓ |
| FreeBSD | - | - | - | ✓ |
| OpenBSD | - | - | - | - |
| Solaris | ✓ | ✓ | ✓ | ✓ |

As already described in Section 4.2, for static addresses, FreeBSD sends NS-DAD before it joins the solicited-node multicast groups corresponding to the tentative address. Therefore, *ndtrack* joined the solicited-node multicast groups corresponding to the tentative address after the tested computer send NS-DAD. As a result, the monitoring node did not discover the address assignment.

### 4.4.2 Network with stateful DHCPv6

The next experiment tested stateful DHCPv6. DHCPv6 clients were running Windows 7, 8, 2008 R2, Ubuntu 12.10, and Solaris (one computer for each operating system). As expected, *ndtrack* detected all address assignments. The proposed approach detects DHCPv6 leases.

### 4.4.3 Comparison to Other Methods

Figure 4.2 shows the network that we used to compare *ndtrack* with NC polling and *addrwatch* [60]. We tested MLD snooping of solicited-node multicast groups both enabled and disabled. Note that NC polling does not depend on MLD snooping.

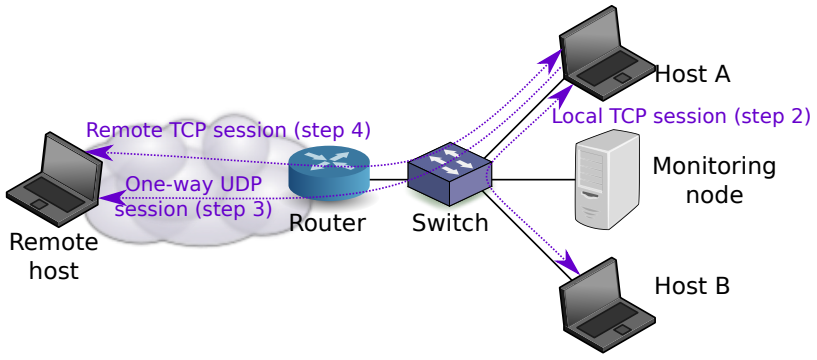



Figure 4.2: The test case that compares the *ndtrack* with other methods.

The test case followed several steps for each method. (1) Two Linux hosts were deployed in the network. (2) Host A opened a connection to host B. The hosts transferred a file in this connection. (3) Host A initiated a one-way UDP connection outside the network. (4) Host A opened a TCP session to the remote host. (5) Hosts A and B disconnected.

During the experiment, we monitored NC of the router, and the outputs of *addrwatch* and *ndtrack*. Table 4.2 compares the methods.

Table 4.2: Comparison of our approach with other methods (✓means detected).

| | NC polling | addrwatch | | *ndtrack* | |
|---|---|---|---|---|---|
| MLD snooping | Does not matter | Inactive | Active | Inactive | Active |
| A, B connected | - | ✓ | - | ✓ | ✓ |
| Local TCP | - | ✓ | - | ✓ | ✓ |
| One-way UDP | - | ✓ | - | ✓ | ✓ |
| Remote TCP | ✓ | ✓ | - | ✓ | ✓ |
| A, B disconn. | - | - | - | ✓ | ✓ |

Both hosts were successfully identified by *ndtrack* even with MLD snooping active for solicited-node multicast groups. Additionally, *ndtrack* detected that the addresses were no longer used by observing MLD queries and replies.

### 4.4.4 Real network deployment

The final experiment aimed at long-term monitoring (almost a month) of SLAAC in a network with MLD querying enabled. The network spans two buildings and is available for all employees of the faculty.

We successfully validated that *ndtrack* detected IPv6 addresses of devices under our control among other devices of our colleagues that were active in the network. We also validated that the addresses are correctly identified as no longer assigned after the hosts disconnect or stop using the addresses (see Figure 4.3 for the statistics).

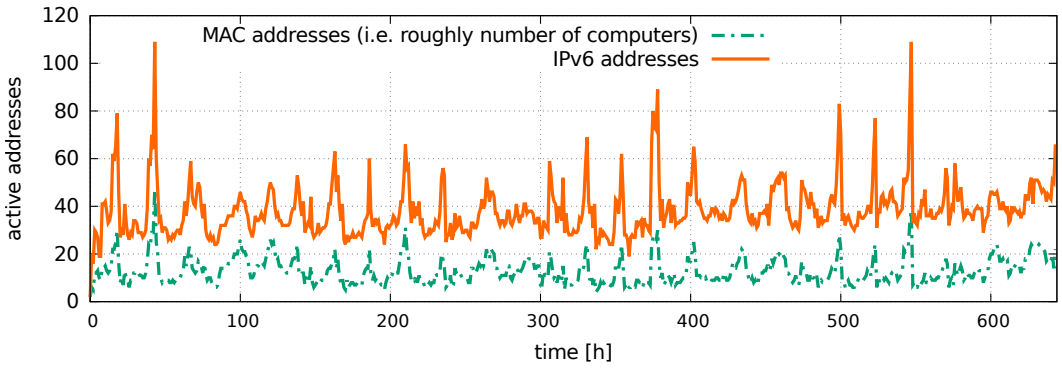



Figure 4.3: Real network monitoring. The number of known addresses rises during working hours and drops at night.

The monitoring node was deployed in building B. To further test the timing of the detection, we connected a device to the network in building A. All addresses of the tested device were correctly identified and later dropped when we disconnected the device.

## 4.5 Considerations about the IPv6 address detection

In wireless networks, packet loss is frequent, Yourtchenko and Nortmark measured [104] that a duplicate address is correctly detected in wireless networks only in about 80 % cases. In unjammed wired networks, packet loss is rare, but problems such as network split may hide some ND messages from the observing node [100].

RFC 4541 [11] describes MLD snooping, during which a switch inspects MLD reports and learns the multicast groups that devices on each port listen to. Consequently, the switch can forward multicast traffic only to links with listeners. Most switch vendors implement MLD snooping [100]. Vyncke et al. [100] argue that currently, switches do not have enough memory to track link-local multicast groups. Consequently, switches flood traffic in solicited-node multicast groups to all ports. The proposed IPv6 address assignment tracking based on timed transducers deals even with networks where switches perform MLD snooping for link-local multicast groups since it subscribes to all MLDv2-capable routers multicast group (*ff02::16*) and all solicited-node multicast groups detected by analysing MLD reports in the multicast group for all MLDv2-capable routers. However, in networks spanning a wide area, the delays can cause that a switch in the network receives the MLD report (join) too late and consequently misses the NS-DAD.

## 4.6 Chapter conclusion

Compared to the related work described in Section 2.2, the proposed IPv6 address assignment tracking based on timed transducer detects all addresses of each node, signals newly assigned addresses immediately, does not poll routers in the network, and distinguishes between states when an IPv6 address was dropped or is merely not used. However, the biggest downside of the proposed IPv6 address assignment tracking is its reliability on the visibility of ND traffic. Each lost message can create inconsistencies between the timed transducer state and the network. The proposed IPv6 address assignment tracking works even in networks with MLD snooping active for solicited-node multicast groups (IPv6 multicast is not broadcasted).

With these considerations in mind, the proposed IPv6 address assignment tracking based on timed transducer fulfils Hypothesis 1. If desired the IPv6 address assignment tracking can be further extended in the following way:

- Assignments can be validated with additional NSes. Nevertheless, the additional messages are visible to end hosts, and consequently, such extension is not suitable for the LI use case.
- In combination with NC polling [41], the monitoring node can reveal assigned IPv6 addresses that were not detected, for example, because some of the traffic was lost and did not reach the monitoring node.

This thesis focuses on instant recognition of address assignments that is transparent to the hosts. Hence, these extensions are not considered in this work.

# Chapter 5

# Clock-skew-based remote computer identification

Sometimes, it is not possible to deploy partial identity detectors to the LAN of a target of an intercept. Imagine a home network with several individuals or a network of people sharing a flat. In a small network, a deployment of an additional device allowed by an official warrant can raise suspicion of the intercept target.

This PhD research has studied the clock-skew-based remote computer identification method presented by Kohno et al. [59] as formulated in Hypothesis 2. The method identifies computers based on *manufacturing deviations* [32, Section 2.1] causing small differences in time measurement on each host in the network.

A *fingerprinter* (observer) monitors timestamps of *fingerprintees* (identification subjects). By default, each machine has a constant built-in error in the time measurement. The fingerprinter estimates the error and utilise the estimation as an identifier. TCP timestamps allows transparent clock-skew-based identification by any observer of the TCP flows.

However, during the evaluation of the clock-skew-based identification, we have learnt that the applicability is lower than expected. We [81] revealed that clock value changes, such as those caused by NTP [65], influence clock skew. Later [73], we focused more on the applicability of clock-skew-based measurements in an identification scheme suitable for LI. The final paper [74] formally evaluated the requirements for an accurate clock skew estimation. Additionally, the final paper presented a method that remotely links IPv4 and IPv6 addresses of the same computer. Moreover, the paper [74] elaborated on various use cases of clock-skew-based identification and presented a scenario in which a user can mimic arbitrary clock skew. This chapter is based on the content of the papers on the clock-skew-based identification [73, 74, 81]. I am the author of the text of the papers.

For clock skew measurements, we developed a tool *PC Fingerprinter* (pcf)[1] The basis of the tool was implemented as a master thesis of Jirásek [54] under my supervision. I rewrote the code into C++ and added support for IPv6, linkage of multiple addresses, and computers with unstable clock skew. Later, bachelor thesis of Franková [34] focused on clock skew measurements and comparison of timestamp sources under my supervision.

Even though this PhD research has revealed that there is only a limited applicability of the method for LI, it improved understanding of several specifics that were not covered by other papers in the area of clock-skew-based computer identification. The contribution of this PhD research in the area of clock-skew-based remote computer identification is:

[1] https://github.com/polcak/pcf

1. The formal requirements for accurate clock skew measurement (see Section 5.3).
2. The discovery that NTP makes clock skew unstable (see Section 5.4).
3. The method that links the IPv4 and all IPv6 addresses of a single computer (see Section 5.5). (However, Beverly and Berger [6] presented a similar method based on the same ground on a conference held earlier than the paper *Clock-Skew-Based Computer Identification: Traps and Pitfalls* [74] appeared in the journal. Nevertheless, the paper [74] was submitted before the conference.)
4. The guide to mimic a constant clock skew of another computer (see Section 5.6).
5. The study of a short-term measurement in a moderately sized real network (see Section 5.7).

## 5.1 Clock skew computation

The following description of the clock skew computation is based on my text that appeared in our paper [74].

Let us denote the time reported by clock $C$ at time $t$ (as defined by national standards, that is the *true time*) as $\mathrm{R}_C(t)$. The offset is the difference between two clocks: $\mathrm{off}_{C,D}(t) \equiv \mathrm{R}_C(t) - \mathrm{R}_D(t)$. Assume that $\mathrm{off}_{C,D}$ is a differentiable function in $t$, then, *clock skew* $\mathrm{s}_{C,D}$ is the first derivative of $\mathrm{off}_{C,D}$. Clock skew is measured in *parts per million* (ppm), meaning that every million time units, the difference between the clocks and the true time increases (or decreases) by the specified amount of units. For example, 20 ppm means that every second, the clock error increases by 20 $\mu$s.

Consider $C$ to be the clock of the fingerprinter (observer) and $D$ to be the clock of the fingerprintee (the monitored node) as depicted in Figure 5.1. $\mathrm{R}_D$ is not observable by the fingerprinter, instead, it sees packets marked with timestamps delayed by $\epsilon(t)$, $\epsilon(t)$ denotes the delay observed at time $t$. The delay $\epsilon(t)$ is composed of the processing time at both the fingerprinter and the fingerprintee and the network delay. If $\epsilon$ was constant, the first derivative of $\mathrm{off}^{\epsilon}_{C,D}(t) \equiv \mathrm{R}_C(t) - \mathrm{R}_D(t - \epsilon(t))$ would have been equal to the first derivative of $\mathrm{off}_{C,D}$. Unfortunately, $\epsilon$ is not a constant.

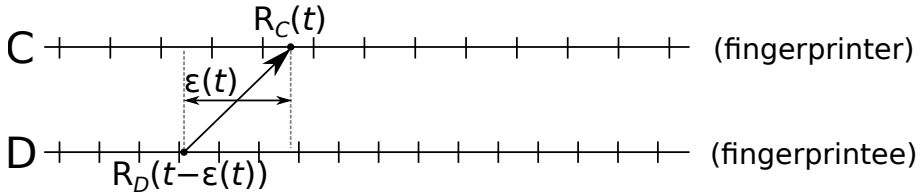



Figure 5.1: Each timestamp of the fingerprintee is delayed by the network and the network stacks of the end hosts.

Let us represent observed timestamps from the fingerprintee as *offset points* $(x, y)$ where $x$ is the observation time, either $\mathrm{R}_C(t)$ or the elapsed time since the start of the measurement, that is $\mathrm{R}_C(t) - \mathrm{R}_C(t_{\mathrm{start}})$; and $y$ is the observed offset $\mathrm{off}^{\epsilon}_{C,D}(t)$. Kohno et al. proposed to estimate clock skew by the slope of the upper bound of all offset points. They have shown that the slope of the upper bound is similar to the slope of the $\mathrm{off}_{C,D}$. Consequently, the first derivatives are similar and the clock skew can be estimated by computing the slope of the upper bound of all offset points.

## 5.2 Related work in clock-skew-based identification

Several timestamp sources have been studied by other researchers, including TCP [59], ICMP [17, 59], HTTP [67, 105], IEEE 802.11 [52, 62], and custom timestamps [50]. Some timestamps can be observed passively whereas other timestamp appear as a result of active probing of the fingerprintee initiated by the fingerprinter.

Table 5.1 shows the timestamp sources. Passive fingerprinting is transparent to the fingerprintees whereas active fingerprinting generates more traffic into the network which can reveal the fingerprinting to the fingerprintee. Applicability in LI requires passive methods.

| Method | Type | Frequency |
|---|---|---|
| ICMP | Active | 1 kHz |
| TCP | Passive | 10-1000 Hz (OS-dependant) |
| Application layer protocols | Active or passive | Method/OS-dependant |
| Application-generated timestamps | Active | Method/OS-dependant |
| IEEE 802.11 beacon frames | Passive | 1 MHz |

Table 5.1: Comparison of timestamp sources.

Kohno et al. [59] observed that TCP and ICMP timestamps are not influenced by time changes on the fingerprintee including running NTP [65] daemon. However, Kohno observed changes in clock skew caused by external factors such as temperature. Nevertheless, we [81] found that TCP timestamps created by the Linux kernel are influenced by time modifications since 2007. Ding-Jie Huang et al. [50] observed jump points in timestamp values caused by time adjustments and roaming in wireless networks. Section 5.4 provides more details on unstable clock skew caused by NTP and time adjustments in general. Algorithm 5.1 detects changes in observed clock skew and timestamp values including jump points.

Recently, Beverly and Berger [6] and Scheitle et al. [90] applied clock skew monitoring to active detection of IPv4 and IPv6 sibling addresses learnt for a host name from DNS. Their goal is to determine if both addresses belong to the same machine. The goal of this research is detection of IPv4 and IPv6 addresses belonging to the same hosts. However, our research is passive. Our paper [74] was in a review process when Beverly and Berger published their paper [6][2].

## 5.3 Accuracy of clock skew measurements

The application of clock-skew-based identification in LI requires that the clock skew of a computer is computed quickly and with high accuracy. The application in multi-factor identification [50] suggests that a quick identification is possible. This section investigates the accuracy both formally and in an empirical study of 24,071 measurements. This section is based on the text from the paper *Clock-Skew-Based Computer Identification: Traps and Pitfalls* [74]. I am the author of the text.

The upper bound of the offset points crosses at least two offset points by definition. Let us denote one of these points as an offset point $X$. Its coordinates are $X = [t_X, \mathrm{off}^{\epsilon}{}_{C,D}(t_X)]$ where $t_X$ is the observation time. In addition, let us denote the amount of time elapsed

[2]Moreover, *pcf* estimates clock skew for IPv6 addresses since 2012, see https://github.com/polcak/pcf/commit/c828ea6e39326776704e9bbdfbcefdb1218c9449

during period $T$ (of the national standards *true time*) on clock $C$ as $\mathrm{E}_C(T)$. According to Formula 5.1, the y-coordinate of $X$ equals to $\mathrm{off}_{C,D}(t_X) - \mathrm{E}_D(\epsilon(t_X))$.

$$\begin{aligned} \mathrm{off}^{\epsilon}{}_{C,D}(t) &= \mathrm{R}_C(t) - \mathrm{R}_D(t - \epsilon(t)) = \\ &= \mathrm{R}_C(t) - \mathrm{R}_D(t) - \mathrm{E}_D(\epsilon(t)) = \\ &= \mathrm{off}_{C,D}(t) - \mathrm{E}_D(\epsilon(t)) \end{aligned} \tag{5.1}$$

Consider the two offset points $K', L'$ located on the upper bound $\mathrm{b}(t)$. The offset points were observed at time $t_1$ and $t_2$. The points $K', L'$, and the point $M' \equiv [t_2, \mathrm{off}_{C,D}(t_1) - E_D(\epsilon(t_1))]$ form a right triangle. In addition, consider the right triangle defined by points $K \equiv [t_1, \mathrm{off}_{C,D}(t_1)], L \equiv [t_2, \mathrm{off}_{C,D}(t_2)$, and $M \equiv [t_2, \mathrm{off}_{C,D}(t_1)]$. $K$ and $L$ are located on the line representing real offset $(\mathrm{off}_{C,D}(t))$ at time $t_1$ and $t_2$. Hence, the points $K$ and $L$ represent the real offset between the clock of the fingerprinter and the fingerprintee at the time $t_1$ and $t_2$ that is not biased by the delay $\epsilon$.

From the definition of the tangent function, Formula 5.2 defines the observed clock skew whereas the real clock skew is defined by Formula 5.3.

$$\mathrm{s}_{\mathrm{observed}} \equiv \tan \alpha' = \frac{(\mathrm{off}_{C,D}(t_2) - \mathrm{E}_D(\epsilon(t_2))) - (\mathrm{off}_{C,D}(t_1) - \mathrm{E}_D(\epsilon(t_1)))}{t_2 - t_1} \tag{5.2}$$

$$\mathrm{s} \equiv \tan \alpha = \frac{\mathrm{off}_{C,D}(t_2) - \mathrm{off}_{C,D}(t_1)}{t_2 - t_1} \tag{5.3}$$

Combining Formulae 5.2 and 5.3,

$$\mathrm{s}_{\mathrm{observed}} = s + \frac{\mathrm{E}_D(\epsilon(t_1)) - \mathrm{E}_D(\epsilon(t_2))}{t_2 - t_1} \tag{5.4}$$

expresses the dependency of the observed clock skew on the real clock skew and the error introduced by the observed volatile latency $\epsilon$. Observe that Formula 5.4 expresses the observed clock skew as a sum of the real clock skew and the error

$$e \equiv \frac{\mathrm{E}_D(\epsilon(t_1)) - \mathrm{E}_D(\epsilon(t_2))}{t_2 - t_1}. \tag{5.5}$$

Let us denote $\Delta t \equiv t_2 - t_1$ and $\Delta\epsilon \equiv |\mathrm{E}_D(\epsilon(t_1)) - \mathrm{E}_D(\epsilon(t_2))|$. Additionally, let us denote the expected accuracy of a clock skew measurement as $A$. To satisfy the accuracy requirement, the observed error $e$ has to be lower than $A$ as displayed in

$$\frac{\Delta\epsilon}{\Delta t} \leq A. \tag{5.6}$$

Formula 5.6 shows that the quality of the estimation depends on the stability of the network latency $\Delta\epsilon$ and the elapsed time $\Delta t$. Note that $\Delta t$ is weakly lower than the total duration of the measurement as $\Delta t$ cannot be bigger than the measurement duration. Hence, a longer measurement can yield more accurate estimates as a longer measurement can detect offset points with similar $\Delta\epsilon$ and bigger $\Delta t$.

Supposing that the fingerprinter observes only packets going through one network path, the latency introduced by wires and fibres is constant. Papagiannaki et al. studied the single hop delay [70]. Their conclusion is that "there is at least one packet that experiences no queueing in each one minute interval" on a hop (an intermediary device). Based on

the study of Papagiannaki et al., a higher number of packets can yield an upper bound for which the $\Delta\epsilon$ is low. Both longer duration of a measurement $\Delta t$ and additional packets that can experience lower $\epsilon$ relax the conditions on the fluctuation of $\epsilon$ as shown in Formula 5.6.

Additionally, the software generating timestamps may increase the variance of $\epsilon$ when an already obtained timestamp value is delayed by a data processing algorithm that constructs the network message.

Even more, computer clocks are updated by a constant value. This creates quantisation error and it is an important factor in the clock skew computation. Without synchronised sampling [105], it takes longer to compute the clock skew of a source with a lower frequency compared to a high frequency timestamp source.

To examine the relation established by Formula 5.6 between the duration of a fingerprinting $\Delta t$ and the observed network latency $\Delta\epsilon$, we monitored timestamps generated by our laboratory computers, all running Red Hat Enterprise Linux 6.6 with Linux kernel 2.6.32. We monitored TCP timestamps and custom timestamps obtained by standard calls in JavaScript (*Date.getTime()*) and Python (*time.time()*).

In total, we evaluated 9,959 experiments with TCP timestamps, 10,016 experiments with custom Python-generated timestamps, and 4,096 experiments with JavaScript-generated timestamps. During each experiment, we computed current clock skew $s_t$ of the computer based on the actual conditions (for example, temperature). Then, we determined the number of examined timestamps and the elapsed time, after which the clock skew estimation did not leave the $s_t \pm 1$ ppm interval.

Figure 5.2 depicts the minimal, maximal, and median value of the error in clock skew estimation. The quality of the estimation improves over time, as expected by Formula 5.6.

The observation confirms the expectations raised by Formula 5.6. The longer an experiment lasts, the less likely it is to get a timestamp that introduces an offset point that incorrectly shifts the upper bound of all offset points. Nevertheless, besides time duration, the variance of delay $\epsilon$ also plays a major role.

## 5.4 Influence of time manipulations on clock skew

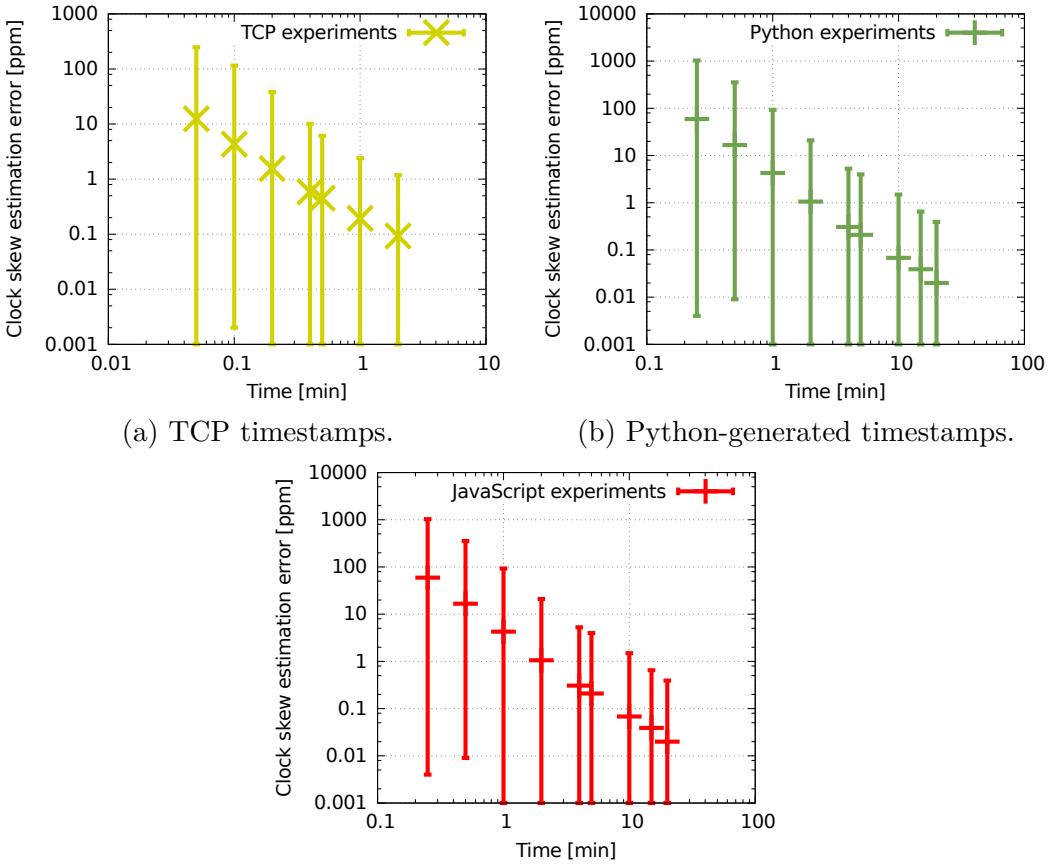
Experiments in the original paper describing clock-skew-based identification [59] suggest that NTP and other time manipulations do not influence TCP timestamps. However, our experiments with Linux hosts revealed computers with variable clock skew. We observed both sudden changes in timestamp values and little changes in clock skew without visible jumps in timestamp values. Our observations were confirmed by other researchers [50, 90].

The NTP client continuously calls a system call *ntp_adjtime()* that modifies the number of crystal oscillator ticks per second to compensate for the inbuilt error. The goal is to converge to the true time slowly and then maintain the synchronised state. As a result, applications running on the system do not observe any sudden changes in system time.

Another use case for NTP is a one-time synchronisation. In this scenario, an NTP client queries an NTP server a replaces the local time with the time on the server with a system call *settimeofday()*. In this case, applications running on the client observe a sudden change in system time. However, *ntpdate*[3] calls *adjtime()* when the difference between remote time and local time is lower than 0.5 seconds.

Across operating systems, timestamps inserted in user space are obtained by system calls that represent time as observed by the user. The purpose of NTP is to provide time

[3] http://www.ntp.org/




(a) TCP timestamps.

(b) Python-generated timestamps.

(c) JavaScript-generated timestamps.

Figure 5.2: Minimal, maximal, and median value of the error during clock skew estimation.

as close to the true time as possible. Hence, every timestamp learnt in the user space carrying the system time is influenced by NTP and other time changes. However, system calls returning time after last boot and similar information are exception.

Our experiments with various operating system showed that time changes are visible in timestamps produced by user space applications in all tested operating systems — Linux, Windows, OpenBSD, Apple iOS, and Apple Mac OS X. At TCP level, NTP does not change clock skew of a Windows computer. Both Linux and OpenBSD propagate NTP changes to TCP timestamps. Apple operating systems exhibit highly unstable clock skew on TCP level. We were not successful [73] in fingerprinting Apple devices from TCP timestamps.

Table 5.2 summarises the influence of time changes on observed clock skew. Computers with stable clock skew can be linked over time because the clock skew does not change even if the computer changes its location or IP address. For observable changes of time stamp values caused by time manipulation, a fingerprinter can detect jump points, estimate clock skew between jump points, and consequently learn the clock skew of the computer. Then the fingerprinter can track the computer over time in different locations and with different IP addresses, similarly to stable clock skew. Offset points of computers with unstable clock skew do not form a line.

Table 5.2: Influence of time manipulations on clock skew fingerprinting.

| | TCP | | | Userspace timestamps |
|---|---|---|---|---|
| | Continuous NTP | *adjtime()* | *settimeofday()* | |
| Linux | Unstable, 0 ppm | Observable | Stable | Observable changes |
| OpenBSD | Unstable, 0 ppm | Observable | Stable | Observable changes |
| Windows | Stable | | | Observable changes |
| Apple | Highly unstable, large | | | Observable changes |

## 5.5 Applicability for IPv6 addresses

Network layer does not modify the payload produced by upper layers. Consequently, both TCP timestamps and timestamps inserted by applications are not influenced by network layer. Hence, this PhD research has pursued the idea that IPv4 and IPv6 addresses are linkable via clock skew [34, 74]. IPv4 and IPv6 address linking based on active fingerprinting was independently published by Beverly and Berger [6], see Section 5.2 for more details.

This PhD research does not consider clock skew to be stable over time. Instead, *pcf*, the fingerprinting tool developed as a part of this PhD research, continuously computes clock skew for each active IP address in the network based on the Algorithm 5.1.

**Algorithm 5.1. Proposed clock skew estimation**

1. Initialize *previous batch* to empty.
2. Collect a batch of timestamps. The batch can be determined by duration or the count of timestamps.
3. For each new batch, estimate the clock skew based on offset points generated for the timestamps in the *current batch*. If the *previous batch* is empty, set *current batch* as *previous batch* and go to step 2.
4. Compare the estimate for *current batch* to the previous estimate. If the estimate differs by more than 10 ppm (by default), clock skew changed or a jump point was detected, go to step 6. Otherwise, go to step 5.
5. Append offset points in *current batch* to the *previous batch* and estimate clock skew based on all merged offset points. Go to step 2.
6. Ignore *current batch* as the observed clock skew changed or the batch contains a jump point. Go to step 1.

For each batch created by Algorithm 5.1, *pcf* creates a triplet consisting of:

- the clock skew estimate,
- initial time from which the clock skew estimate is valid (observed time of the first offset point in the batch),
- final time until which the clock skew estimate is valid (observed time of the last offset point in the batch).

Hence, for stable clock skew, *pcf* detects a single triplet; for unstable clock skew, *pcf* detects multiple triplets, each valid for a particular period. For stable clock skew, *pcf* continuously improves the clock skew estimate (see Formula 5.6) and prolongs its validity. For unstable clock skew, *pcf* detects jump points and periods where clock skew changed.

This PhD research considers two sequences of clock skew triplets (each observed for an IP address) to be linkable, meaning both IP addresses belongs to the same anonymity set, which (hopefully) represents a single computer, if one of the following requirements holds [74]:

1. Both clock skew estimates are stable (both sequences have a single triplet) and both estimates are within the range of $\pm 1$ ppm. In this case, it does not make a difference if both addresses are active during the same period or not.

2. Both clock skew estimates were within the range of $\pm 1$ ppm during the periods when both addresses were active. In this case, there has to be at least one period during which both addresses were active; when the clock skew of one of the addresses changed, the other clock skew changed soon to a similar value, for example, a jump point was detected for both IP addresses.

The first requirement is the same as originally used by Kohno et al. [59]. The latter requirement deals with changes in clock skew as Section 5.4 presents.

Tests of Algorithm 5.1 revealed:

- JavaScript-generated timestamps carried over IPv4 and IPv6 converged to the same values.

- TCP timestamps converged to the same values as JavaScript-generated timestamps for all operating systems except Apple.

- TCP timestamps of Apple computers exhibited very large clock skew. Generally, the computed values for IPv4 and IPv6 did not match. However, there are observable changes in both IPv4 and IPv6 traces generated by an Apple computer.

## 5.6 Guide to mimic clock skew of a different computer

A possibility to influence the observable clock skew of a computer is a major downside for the identification of the intercept target during LI. Hence, this section focuses on the possibility to mimic an arbitrary stable clock skew [74].

An NTP daemon maintains a file called *driftfile* where it stores current correction for the built-in clock error. The goal is to maintain accurate time even after the NTP daemon is restarted. After the restart, the daemon sets the correction stored in the *driftfile*.

To mimic clock skew of another computer, an adversary can follow the Algorithm 5.2.

**Algorithm 5.2. Mimicking clock skew of a different computer** [74]:

1. Run an NTP daemon and find the clock skew of the attacking computer, for example, from the *driftfile*.

2. Find the clock skew of the victim, preferably by fingerprinting the victim computer by the attacking computer (still running the NTP daemon).

3. Add the clock skew learnt in step 2 to the value stored in the *driftfile* in step 1. Use the sum as the correction of attacking computer clock, for example, by restarting the NTP daemon on the attacking computer and immediately stopping the daemon.

By applying Algorithm 5.2, we were able to reproduce clock skew of a different computer in our laboratory [74].

A user that tries to evade clock-skew-based detection can randomise clock skew of his or her computer during boot or as a part of network access procedure following Algorithm 5.3.

**Algorithm 5.3. Randomising stable clock skew of a computer**:

1. Generate a random clock skew value, preferably in the range of observed clock skews in real network reported in Section 5.7 or by Lanze et al. [62].

2. Apply the generated random clock skew, for example, by starting and immediately stopping the NTP daemon.

A fingerprinter of a user with randomised clock skew by Algorithm 5.3 estimates different clock skew after each execution of the Algorithm 5.3. Between the changes, the fingerprintee exhibits stable clock skew.

## 5.7 Real world measurements

The final test of clock-skew-based identification aims at real world deployment. Originally, we performed the experiment for IPv4 [73] and later, we expanded the test for IPv6 [74]. We passively fingerprinted devices in our faculty network based on TCP timestamps. The goal was to observe the network traffic in the same manner as an LI system that tries to identify the computers. During the testing, we did not focus on specific devices. For privacy reasons, we did not compare the gathered information to external sources.

Figure 5.3 shows the fingerprinted distribution of clock skew estimations for IPv4 and IPv6 addresses. The majority of observed clock skew is close to zero. Hence, the devices close to 0 ppm are indistinguishable by the clock skew value.

Note that for 92 IPv4 addresses and 30 IPv6 addresses, the estimated clock skew was lower than -1000 ppm or higher than 1000 ppm. Such addresses exhibited similar behaviour as Apple devices in our laboratory. We do not consider addresses with clock skew below -1000 ppm and above 1000 ppm as the laboratory measurements [73] show that the observed values from TCP timestamps of Apple operating systems do not match clock skew estimation from JavaScript-generated timestamps and visual observations.

The estimated clock skew for 80 % IPv4 addresses and 79 % IPv6 addresses were in the range of -100 ppm to 100 ppm. Let us focus on these addresses.

Figure 5.4 shows the distribution of clock skew estimates for the range of -100 ppm to 100 ppm; for each clock skew estimate (x-axis), it reports the number of other estimates in the anonymity set defined by the $\pm 1$ ppm range, that is possibly of the same computer. The largest anonymity set for IPv4 contains 223 IPv4 addresses; the largest anonymity set for IPv6 contains 72 IPv6 addresses. Both largest anonymity sets are close to 0 ppm. Hence, computers having assigned 34.5 % of all IPv4 addresses and 40.7 % of all IPv6 addresses are indistinguishable. Most probably, these addresses were assigned to computers running NTP.

Even without hosts running NTP continuously, the majority of devices appear in the same anonymity set with multiple other devices. Between -100 ppm and 100 ppm, only 21

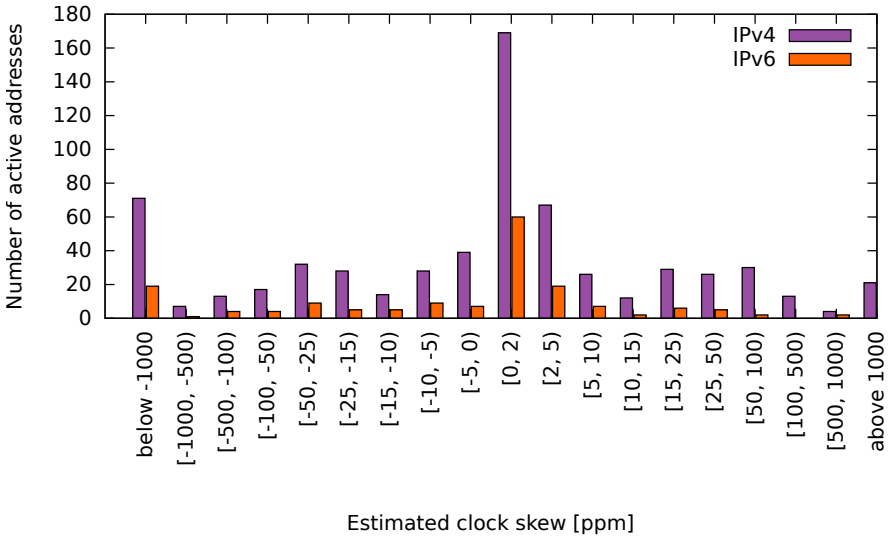



Figure 5.3: Histogram of clock skew distribution in real network. Note that the range of bins is not uniform.

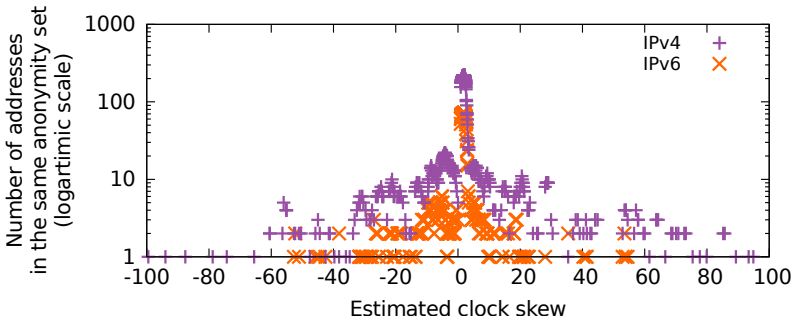



Figure 5.4: The estimated clock skew and number of addresses with similar clock skew in the range of −100 ppm to 100 ppm.

IPv4 addresses were not in an anonymity set with another IPv4 address. Quick, unique identification of computers in a moderately sized network is not possible with restrictions compatible with LI. These observations confirm observations of Lanze et al.[62].

## 5.8 Applications of clock-skew-based identification

Although the real deployment of clock-skew-based identification for LI was not successful, other applications for clocks-skew-based identification exist. This section focuses on the applicability of clock-skew-based identification.

A fingerprinter can aim at short-term fingerprinting, for example, during multi-factor authentication [50] or identification for LI. Nevertheless, long-term fingerprinting can reveal additional information, such as geographical position [67], and link addresses based on clock skew changes [90].

Based on the fingerprinting duration and active or passive observations of timestamps (see Table 5.1), it is possible to classify fingerprinters into four categories [74].

1. Passive short-term fingerprinting aims on quick identification of a fingerprintee with the applications in LI or rogue access points identification [62]. Both this PhD research and the research of Lanze et al. [62] show a limited applicability in small networks only. Fingerprintees can spoil the fingerprinting by clock skew modifications, such as the clock skew randomization achieved by Algorithm 5.3.

2. A passive long-term fingerprinter can observe changes in clock skew caused by NTP or temperature changes. Consequently, the fingerprinter can link all IP addresses used by a specific computer, learn geographical location [67] or disclose computers behind network address translator [59].

3. An active short-term fingerprinter can use clock-skew-based identification during multi-factor authentication [50] or improve the estimates [93] compared with short-term passive fingerprinter. However, the disadvantages connected to the limited uniqueness of clock skew and possible influence of the fingerprintee on its clock skew also apply to active short-term fingerprinting.

4. An active long-term fingerprinter can improve the quality of the clock skew estimates [93] compared to passive-long-term fingerprinter. Active long-term fingerprinting can be applied in deanonymization [105], virtual PC and honeypot detection [59], and IPv4 and IPv6 linkability [90].

## 5.9 Chapter conclusion

This chapter describes remote computer identification based on clock skew estimation. Previous research indicates that unique short-term identification is possible [50, 59, 93] and that passive clock skew evaluation is possible [59]. However, our research revealed that the main source of timestamps for passive fingerprinting, TCP, has became influenced by NTP changes. This chapter identifies that this change impacts mostly short-term fingerprinting. The results also suggests that long-term clock-skew-based-fingerprinting is a viable choice for deanonymization [105], virtual PC and honeypot detection [59], and IPv4 and IPv6 linkability [90].

The study of real network clock skew distribution, presented in Section 5.7, shows that most of the real devices exhibit clock skew in the range of -100 ppm to 100 ppm with the majority of devices close to 0 ppm. Consequently, a short-term fingerprinter cannot reliably identify computers solely by clock skew.

Clock-skew-based identification does not work in conditions restricted by Hypothesis 2. Remote short-term passive fingerprinting based on TCP timestamps is not reliable. However, clock-skew-based identification can be employed in the following use cases:

1. A long-term passive fingerprinter can link addresses of NTP-enabled hosts by observing the shape of the upper bound.

2. An active fingerprinter can combine clock-skew-based fingerprinting with another identification method, such as browser fingerprinting.

Nevertheless, both modification does not allow clock-skew-based identification to be applicable in LI for a moderately sized network. Short-term fingerprinting is applicable in small networks, especially in a controlled environment where users cannot manipulate time.

# Chapter 6

# Identity graphs

The ability to link identifiers is crucial to identify all data of the intercept targets. For example, the interception of CC is often based on IP addresses because IP addresses appear in the header of every IP datagram [45, page 272]. Nevertheless, the IP addresses are assigned dynamically. Hence, an LI system has to link a stable identifier such as a RADIUS username, a network access identifier, a cable modem identifier, or a stable MAC address to the dynamically assigned IP address [45, page 272]. Recently, the significance of application-layer-identifier-based LI grows [3, 45, 98, 103].

The input identifier is a long-term identifier identifying a user, a household, a computer, or a computer network interface. Depending on the wording of the warrant (see Section 3.5 for the problem statement), some linking might be allowed or forbidden.

To incorporate identification methods that are not completely accurate, the model should include accuracy, which disqualifies linkage through multiple inaccurate observations. For digital forensic, the model should be aware of time so that the dynamic identifiers can be tracked over time and the model should support time-related reasoning on the gathered knowledge.

This chapter defines identity graphs that allow cross-layer linkage of information from various partial identity detectors in conformance with Hypothesis 3. The proposed model is based on a graph representation of identity information; vertices represent identifiers and edges reflect the linkage between two identifiers. Identity graphs support operations that link identifiers following constraints based on the wording of a warrant and other parameters, such as the inaccuracy of the linkage. The definition of identity graphs and the operations is extensible. Hence, this chapter provides a framework that is tweaked for LI. For other applications, the framework can be extended, for example, by specifying more categories of identifiers or definitions of other operations.

An earlier version of the proposed identity graphs was implemented as a part of the Sec6Net project [77, 82] based on my idea of identity linking in graphs. Formal description of identity graphs (as implemented as by the Sec6Net project) was published in the paper *On Identities in Modern Networks* [80]. I am the author of the text describing identity graphs and I developed the formulae in the formal description of identity graphs with a minor help of my co-workers. The implementation of identity graphs, as described in this thesis, is available on GitHub[1]. Compared to the original model, identity graphs defined in this thesis (1) have a notion of time, (2) allow better handling of inaccurate partial identity detectors (such as clock-skew-based identification described in Chapter 5), (3) allow

[1] https://github.com/polcak/linking

additional custom operations – constraint functions, and (4) support identifiers of resources such as chat rooms.

Identity graphs are generic and they support identifiers that appear in all layers of the TCP/IP model. As a part of SLIS developed by the Sec6Net project, identity graphs were validated for identity linking based on the following partial identity detectors: DHCP [21], DHCPv6 [8], ND [69] (see Chapter 4), RADIUS [86], PPPoE [64], XMPP [88], IRC [55, 56], OSCAR (proprietary protocol for instant messaging), YMSG (proprietary protocol for instant messaging), SMTP [58], clock skew (see Chapter 5), and Software Defined Networking controllers — OpenDaylight[2] and Pox[3].

## 6.1 Related work in identity linking

Section 2.3 describes ETSI standards. The linkage has to be unambiguous. This chapter defines identity graphs that support various identifiers including those listed by ETSI [27]. In addition to unambiguous partial identity detectors, identity graphs also support inaccurate partial identity detectors. However, inaccurate partial detectors are not mandatory and linkage using information based on inaccurate partial identity detectors can be completely disabled.

Digital forensic literature, for example, Casey [9, page 650–651], emphasise the need to link identities and to select identifiers of the traffic to be intercepted based on the specific network parameters. Identity graphs allow custom queries with arbitrary input identifier and constraints. Casey also mentions that for some intercepts, investigators are authorised to monitor only a specific traffic, for example, web. Identity graphs allow interception of both (1) a single session identified by an application layer identifier and (2) all traffic of the computer where a user authenticates using the same application layer identifier. The distinction between these two use cases can be applied on a per intercept basis. Identity graphs allow setting specific constraints for each query.

Casey [9] describes log files that are available on various devices in the network including DHCP servers, RADIUS servers, ARP tables of network devices. Identity graphs support multiple diverse partial identity detectors including log file analysers.

Casey [9, section 3.3] considers digital investigation with estimated levels of certainty. Identity graphs support partial identity detectors that estimate their inaccuracy. The identifiers are linked based on the inaccuracy.

Hoffman and Terplan [45, page 176] note: "Unfortunately, there are very few known products on the market that support data mining, evidence finding, and correlation functions". Identity graphs aim to fill the gap in identity linking and correlation.

Torres et al. [97] studied identity management systems. As one of the key aspects, they list law enforcement interoperability. A modern LI system needs to process many input partial identity detectors. Identity graphs are based on this observation.

## 6.2 Detection of partial identities

Partial identities can be detected from many sources distributed in the network. These identifiers do not necessarily identify the same subject. There are many sources of identity-

[2] https://www.opendaylight.org/

[3] https://openflow.stanford.edu/display/ONL/POX+Wiki

related information. This PhD. research studied and created traffic parsers and analysers [82], log file analysers [92], and extensions for network-related programs [35].

Obviously, traffic analysis methods are applicable only to unencrypted traffic, or when encryption keys are available to partial identity detectors. Both log file analysis and program extensions can be deployed even when the traffic is encrypted. Log files contain the identifiers unencrypted and the program extensions have access to unencrypted identifiers in the internals of the network-related programs.

An identity graph incorporates identity-related information revealed by multiple partial identity detectors. A goal of each partial identity detector is to provide a connection between two or more identifiers; each identifier represents a different partial identity.

Identity graphs incorporate time $\mathcal{T}$. Additionally, we suppose that $\mathcal{T}$ includes its supremum that we denote as $\infty$. As noted during the description of identity graphs, $\infty$ represents an unknown time in the future.

Partial identity detectors produce messages reflecting changes in observed partial identities; each message contains: Timestamp $t \in \mathcal{T}$ of the message. The identifiers that the partial identity detector revealed are linked. The information about the events concerning the linkage; possible events are *begin*, *continue*, or *end*. The inaccuracy of the linkage. In this thesis, positive real numbers form the domain of the inaccuracy ($Inaccuracy \equiv \mathcal{R}_0^+$). Note that the messages allow transformation into IRI records (*IRI begin*, *IRI continue*, *IRI end*) that are passed to LEA.

Table 6.1 provides a list of identifiers their typical durability, and the subject which the identifier identifies. Note, that the table lists typical values. In some use cases, the behaviour differs, for example, recent Apple and Android mobile devices randomise MAC addresses. Final LI system deployment has to consider such nuances.

Table 6.1: Network identifiers.

| **Indentifier** | **Durability** | **Identified subject** |
|---|---|---|
| MAC address | Typically long-term | Computer network interface |
| IPv6 address | Typically short-term | Computer network interface |
| IPv4 address | Dynamic | Computer network interface |
| DHCP client ID | Typically long-term | Computer |
| DHCPv6 DUID | Typically long-term | Computer (a single OS) |
| RADIUS or PPP username | Long-term | User or household |
| TCP or UDP flow ID | Short-term | Session |
| Application layer username | Long-term | User |
| URI | Dynamic | Web resource |
| Switch ID and port ID | Typically long-term | Computer, household |

This thesis defines the following categories of identifiers:

***L4Flow*** identifies a transport layer flow. Every packet of the flow carries the identifier in header fields. LI probes developed as a part of the Sec6Net project can intercept traffic based on *L4Flow* identifiers.

***IPAddr*** — IP addresses (typically short-term duration, assigned dynamically) identify an interface of a network node. Every IP packet carries source and destination IP addresses. LI probes developed as a part of the Sec6Net project can intercept traffic based on *IPAddr* identifiers.

***IfcOrComp*** — Long-term identifiers of computers or network interfaces, such as MAC addresses, DUIDs, clock skew values.

***AAAUser*** — Authentication usernames of protocols such as RADIUS, PPP identify a set of network devices controlled by a unique user or a household depending on the network.

***L7User*** — Application layer usernames (for example, login names, account identifiers, e-mail addresses) identify a partial identity of a unique user.

***L7Resource*** — An application layer resource such as a chat room or a web page.

Let us define a set of categories as $Categories \equiv \{L4Flow, IPAddr, IfcOrComp, AAAUser, L7User, L7Resource\}$.

## 6.3 Identity graph definition

**Definition 6.1.** An **identity graph** is an extended multigraph $G \equiv (V, E, endpts, category, attributes)$ where:

- $V$ is a set of vertices. Every vertex represents a network identifier.
- $E$ is a set of edges.
- $endpts$: $E \to \{\{x, y\} : x, y \in V \wedge x \neq y\}$ assigns each edge a pair of different vertices (endpoints).
- $category$: $V \to Categories$ is a total function that maps each vertex to its category.
- $attributes$: $E \to Detectors \times \mathcal{T} \times \mathcal{T} \times Inaccuracy$ is a total function that defines attributes of the edges; where $Detectors$ is a finite state of names of partial identity detectors. For each edge, function $attributes$ stores its partial identity detector, the starting time and end time of the linkage, and the estimated inaccuracy.

Queries in identity graphs are based on paths in the graph dentoted as $\langle v_0, e_1, v_1, e_2, \ldots, v_{n-1}, e_n, v_n\rangle$. In formulae where vertices are not important, vertices are omitted from the representation of the path sequence, for example, $\langle e_1, e_2, \ldots, e_{n-1}, e_n\rangle$ [39]. Let us denote the set of all paths in an identity graph G as $\mathcal{P}(G)$.

**Definition 6.2.** A **constraint function** $L$: $\mathcal{P}(G) \to \{\text{true}, \text{false}\}$ evaluates a path and yields *true* if the path fulfils the requirements given by the constraint and *false* otherwise.

Formula 6.1 defines the generic form of the function *linked* that yields a set of linked identifiers for an input identifier represented by vertex $v_i \in V$ based on a set of constraint functions $L_f$. Examples of constraint functions are provided later in this chapter.

$$linked(v_i, L_f, G) \equiv \Bigg\{ o \in V : (\exists \langle v_0, e_1, \ldots, e_n, v_n\rangle \in \mathcal{P}(G)) : v_0 = v_i \wedge v_n = o \wedge \left( \bigwedge_{f \in L_f} (f(\langle v_0, e_1, \ldots, e_n, v_n\rangle)) \right) \Bigg\} \tag{6.1}$$

### 6.3.1 Constraint functions restricting relations between identifiers

As mentioned in Section 3.5, the wording of a warrant for LI can influence the scope of traffic to be intercepted and the identifiers for which to generate IRI.

#### Constraints revealing components of partial identity

Sometimes an LI warrant lists an identifier $A$ that should be monitored. However, the partial identity represented by the identifier $A$ consists of several partial subidentities with their specific identifiers; each partial subidentity is a subset of the original partial identity.

Equivalence 6.2 defines relation $r_{6.2} \subseteq \mathit{Categories} \times \mathit{Categories}$ that restricts the categories of identifiers on a path of components of a partial identity.

$$\begin{aligned} r_{6.2} \equiv \{&(\mathit{L4Flow}, \mathit{L4Flow}), (\mathit{IPAddr}, \mathit{L4Flow}), (\mathit{IfcOrComp}, \mathit{IPAddr}), \\ &(\mathit{AAAUser}, \mathit{IPAddr}), (\mathit{AAAUser}, \mathit{IfcOrComp}), (\mathit{L7User}, \mathit{L4Flow})\}. \end{aligned} \tag{6.2}$$

Formula 6.3 defines constraint function $l_{6.3}$ applied for a path $p = \langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle$. Constraint function $l_{6.3}$ allows only paths that represent components of a partial identity represented by $v_0$.

$$l_{6.3}(\langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle) \equiv \begin{cases} \text{true} & \text{if } ((\forall i \in [1, n]) : r_{6.2}(\mathit{category}(v_{i-1}), \\ & \quad \mathit{category}(v_i))), \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.3}$$

#### Constraints revealing partial identities of specific computer

Sometimes an LI warrant aims at intercepting all traffic of the computer identified by the identifier $A$. Hence, all partial identities that are linkable to be used on the same computer are covered by the intercept. Such LI warrant is only defined if the identifier $A$ is of the category *IPAddr* or *IfcOrComp* as identifiers of these categories represent a computer or its network interface.

Equivalence 6.4 defines relation $r_{6.2} \subseteq \mathit{Categories} \times \mathit{Categories}$ that restricts the categories of identifiers on a path of identifiers belonging to the same computer.

$$\begin{aligned} r_{6.4} \equiv \{&(\mathit{L4Flow}, \mathit{L4Flow}), (\mathit{IPAddr}, \mathit{L4Flow}), \\ &(\mathit{IPAddr}, \mathit{IfcOrComp}), (\mathit{IfcOrComp}, \mathit{IPAddr})\} \end{aligned} \tag{6.4}$$

Formula 6.5 defines constraint function $l_{6.5}$ applied for a path $p = \langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle$. Constraint function $l_{6.5}$ allows only paths that represent a specific computer.

$$l_{6.5}(\langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle) \equiv \begin{cases} \text{true} & \text{if } \mathit{category}(v_0) \in \{\mathit{IPAddr}, \mathit{IfcOrComp}\} \wedge \\ & \quad \wedge((\forall i \in [1, n]) : r_{6.4}(\mathit{category}(v_{i-1}), \\ & \quad \mathit{category}(v_i))), \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.5}$$

**Constraints revealing partial identities of computers where specific user authenticated or logged in**

An LI warrant that orders interception of all traffic of all computers authenticated by a specific user is applicable only if the input identifier is of the *AAAUser* or *L7User*. An *AAAUser* identifier can be used for authentication of several computers. Similarly, a user can log in using an *L7User* identifier on several computers. Hence, the warrant coveres identifiers of all computers where the user authenticated or logged in.

Equivalence 6.6 defines the relation $r_{6.6} \subseteq \mathit{Categories} \times \mathit{Categories}$ that allows linkage (1) from an *AAAUser* identifier to *IfcOrComp* and *IPAddr* identifiers, (2) from an *L7User* identifier to *IPAddr* identifiers.

$$r_{6.6} \equiv \{(\mathit{AAAUser}, \mathit{IPAddr}), (\mathit{AAAUser}, \mathit{IfcOrComp}), (\mathit{L7User}, \mathit{IPAddr})\} \tag{6.6}$$

Formula 6.7 defines constraint function $l_{6.7}$ applicable for a path $p = \langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle$. Constraint function $l_{6.7}$ allows only paths from a *AAAUser* or *L7User* identifiers traversing identifiers belonging to (1) computers authenticated by the identifier represented by $v_0$ of category *AAAUser* and (2) computers used by a user that accessed the account represented by the identifier of $v_0$ of category *L7User*.

$$l_{6.7}(\langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle) \equiv \begin{cases} \text{true} & \text{if } r_{6.6}(\mathit{category}(v_0), \mathit{category}(v_1)) \wedge \\ & \wedge((\forall i \in [2, n]) : r_{6.4}(\mathit{category}(v_{i-1}), \\ & \mathit{category}(v_i))), \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.7}$$

**Constraints revealing identifiers of all users accessing specific resource**

An LI warrant aiming at monitoring of a specific resource provides an *L7Resource* input identifier. The result is a set of the directly connected *L7User* identifiers.

Formula 6.8 defines the constraint function $l_{6.8}$ applicable for a path $p = \langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle$. Constraint function $l_{6.8}$ allows only paths of two vertices from an *L7Resource* identifier to *L7User* identifiers.

$$l_{6.8}(\langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle) \equiv \begin{cases} \text{true} & \text{if } (n = 1) \wedge \mathit{category}(v_0) = \mathit{L7Resource} \wedge, \\ & \wedge\, \mathit{category}(v_1) = \mathit{L7User} \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.8}$$

**Constraints revealing all user accounts logged in or authenticated from computer or set of computers**

An LI warrant for all user accounts accessed from a computer or a set of computers targets an *IPAddr*, *IfcOrComp*, *AAAUser* or *L7User* identifier $A$. The expected result is a set of identifiers of category *AAAUser* or *L7User*.

Equivalence 6.9 defines the relation $r_{6.9} \subseteq \mathit{Categories} \times \mathit{Categories}$ that allows linkage between (1) an IP address and the username accessed from that address or (2) an *IPAddr* or *IfcOrComp* identifier authenticated by an *AAAUser* identifier.

$$r_{6.9} \equiv \{(\mathit{IPAddr}, \mathit{L7User}), (\mathit{IPAddr}, \mathit{AAAUser}), (\mathit{IfcOrComp}, \mathit{AAAUser}))\} \tag{6.9}$$

Formula 6.10 defines constraint function $l_{6.10}$ applicable for a path $p = \langle v_0, e_1, v_1, \ldots, e_n, v_n \rangle$. Constraint function $l_{6.10}$ allows linking identifiers of the partial identity of the devices (1) selected by the input identifier ($l_{6.5}$) or (2) where the user authenticated or logged in ($l_{6.7}$). Only paths ending by two identifiers of the categories accepted by $r_{6.9}$ are allowed by Formula 6.10.

$$l_{6.10}(\langle v_0, e_1, v_1, \ldots, v_{n-1}, e_n, v_n \rangle) \equiv \begin{cases} \text{true} & \text{if } (l_{6.5}(\langle v_0, e_1, v_1, \ldots, e_{n-1}, v_{n-1} \rangle) \vee \\ & \quad l_{6.7}(\langle v_0, e_1, v_1, \ldots, e_{n-1}, v_{n-1} \rangle) \vee \\ & \quad n = 1) \wedge r_{6.9}(\mathit{category}(v_{n-1}), \\ & \quad \mathit{category}(v_n)) \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.10}$$

**Constraints revealing all accessed resources**

An LI warrant for the accessed resources targets an *IPAddr*, *IfcOrComp*, *AAAUser* or *L7User* identifier $A$.

Equivalence 6.11 defines the relation $r_{6.11} \subseteq \mathit{Categories} \times \mathit{Categories}$ that allows linkage between (1) an IP address and the resource accessed from that IP address, and (2) an application username and the resource accessed by the user.

$$r_{6.11} \equiv \{(\mathit{IPAddr}, \mathit{L7Resource}), (\mathit{L7User}, \mathit{L7Resource})\} \tag{6.11}$$

Formula 6.12 defines constraint function $l_{6.12}$ applicable for a path $p = \langle v_0, e_1, v_1, \ldots, e_{n-1}, v_{n-1}, e_n, v_n \rangle$. Constraint function $l_{6.12}$ allows only paths that end in a vertex representing an *L7Resource* identifier.

$$l_{6.12}(\langle v_0, e_1, v_1, \ldots, v_{n-1}, e_n, v_n \rangle) \equiv \begin{cases} \text{true} & \text{if } (l_{6.5}(\langle v_0, e_1, v_1, \ldots, e_{n-1}, v_{n-1} \rangle) \vee \\ & \quad l_{6.7}(\langle v_0, e_1, v_1, \ldots, e_{n-1}, v_{n-1} \rangle) \vee \\ & \quad n = 1) \wedge r_{6.11}(\mathit{category}(v_{n-1}), \\ & \quad \mathit{category}(v_n)) \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.12}$$

### 6.3.2 Time constraints

In some cases, it is necessary to consider only connections in the graph that are valid at a certain time or in a time range, for example, last month, during a specific day, after a specific time point. Let $t_0, t_1 \in \mathcal{T}$ define a period ($t_1 > t_0$) or a specific moment ($t_0 = t_1$).

Formula 6.13 provides one example of a time-based constraint function template $l_{6.13}$: $\mathcal{T} \times \mathcal{T} \times \mathcal{P} \rightarrow \{\text{true}, \text{false}\}$. All edges on the path have to be valid during the whole period $[t_0, t_1]$.

$$l_{6.13}(t_0, t_1, \langle e_1, e_2, \ldots, e_n \rangle) \equiv \begin{cases} \text{true} & \text{if } ((\forall i \in [1, n]) : \Pi_2(\mathit{attributes}(e_i)) \leq t_0 \wedge \\ & \quad \wedge \Pi_3(\mathit{attributes}(e_i)) \geq t_1), \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.13}$$

Formula 6.14 provides another example of a time-based constraint function template $l_{6.14}$: $\mathcal{T} \times \mathcal{T} \times \mathcal{P} \rightarrow \{\text{true}, \text{false}\}$. All edges on the path have to be valid at least once during the period $[t_0, t_1]$ and the period during the previous identifier is valid on the path.

$$l_{6.14}(t_0, t_1, \langle e_1, e_2, \ldots, e_n \rangle) \equiv \begin{cases} \text{true} & \text{if } [t_0, t_1] \cap [\Pi_2(attributes(e_1)), \\ & \quad \Pi_3(attributes(e_1))] \neq \emptyset \wedge ((\forall i \in [2, n]): \\ & \quad [t_0, t_1] \cap [\Pi_2(attributes(e_i)), \\ & \quad \Pi_3(attributes(e_i))] \cap [\Pi_2(attributes(e_{i-1})), \\ & \quad \Pi_3(attributes(e_{i-1}))] \neq \emptyset) \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.14}$$

Let $l_{6.13}(a, b)$ denote a partially bounded function $l_{6.13}(t_0, t_1, \langle e_1, e_2, \ldots, e_n \rangle)$ where $t_0$ is bounded to $a$ and $t_1$ is bounded to $b$. Similarly, let $l_{6.14}(a, b)$ denote a partially bounded function $l_{6.14}(t_0, t_1, \langle e_1, e_2, \ldots, e_n \rangle)$ where $t_0$ is bounded to $a$ and $t_1$ is bounded to $b$. Such partially bounded functions $l_{6.13}(a, b) \colon \mathcal{P}(G) \to \{\text{true}, \text{false}\}$ and $l_{6.14}(a, b) \colon \mathcal{P}(G) \to \{\text{true}, \text{false}\}$ can be used as constraint functions in Formula 6.1.

### 6.3.3 Inaccuracy constraints

In case that some linking information is based on inaccurate partial identity detectors, for example, clock skew, browser fingerprinting, the linking is not transitive. Inaccuracy constraint functions limit the transitivity.

Function $l_{6.15} : \mathcal{R}_0^+ \times \mathcal{P}(G) \to \{\text{true}, \text{false}\}$ evaluates all edges for observed inaccuracy and allows paths $\langle e_1, e_2, \ldots e_n \rangle$ where each edge has inaccuracy lower than given threshold $I_T$, see Formula 6.15.

$$l_{6.15}(I_T, \langle e_1, e_2, \ldots, e_n \rangle) \equiv \begin{cases} \text{true} & \text{if } (\forall i \in [1, n] : \Pi_4(attributes(e_i)) \leq I_T), \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.15}$$

Formula 6.16 defines path inaccuracy function $\text{Inaccuracy}_\text{P} : \mathcal{P}(G) \to \mathcal{R}_0^+$ of a path as a sum of the inaccuracies detected on all edges along the path $\langle e_1, e_2, \ldots, e_n \rangle$.

$$\text{Inaccuracy}_\text{P}(\langle e_1, e_2, \ldots, e_n \rangle) \equiv \sum_{i=1}^{n} \Pi_4(attributes(e_i)) \tag{6.16}$$

Formula 6.17 defines constraint function template $l_{6.17} : \mathcal{R}_0^+ \times \mathcal{P}(G) \to \{\text{true}, \text{false}\}$ that compares the total inaccuracy along a path with a threshold $I_T$.

$$l_{6.17}(I_T, \langle e_1, e_2, \ldots, e_n \rangle) \equiv \begin{cases} \text{true} & \text{if } \text{Inaccuracy}_\text{P}(\langle e_1, e_2, \ldots, e_n \rangle) \leq I_T, \\ \text{false} & \text{otherwise.} \end{cases} \tag{6.17}$$

Let $L_{6.17}(n)$ denote a partially bounded function $l_{6.17}(I_T, \langle e_1, e_2, \ldots, e_n \rangle)$ where $I_T$ is bounded to $n$. Similarly, let $L_{6.15}(m)$ denote a partially bounded function $l_{6.15}(I_T, \langle e_1, e_2, \ldots, e_n \rangle)$, where $I_T$ is bounded to $m$. Such partially bounded functions $L_{6.17}(n) \colon \mathcal{P} \to \{\text{true}, \text{false}\}$ and $L_{6.15}(m) \colon \mathcal{P} \to \{\text{true}, \text{false}\}$ can be used as constraint functions in Formula 6.1.

### 6.3.4 Other constraint functions

The list of the constraint functions presented above is not exclusive. It is possible to extend the framework and define additional constraint functions based on the application of identity graphs.

### 6.3.5 Identity linking in networks with address translation

For networks performing NAT, in case that CC interception is located after the translator, the translation mapping between the flows has to be included in the identity graph. Hence, a partial identity detector has to reveal the NAT mapping. The translation can be detected (1) by probes before and after the translator, for example, NAT mapping detector of Grégr [40, Chapter 4] or a log file analyser [92], (2) from logs generated by a network address translator or (3) by another mechanism, such as clock-skew-based linking [59].

The full text of this thesis defines rules for identity graph construction that allow linking in networks with NAT and CGN using the previously defined operations.

## 6.4 Validation

Identity graphs described in this chapter were validated in several scenarios during the Sec6Net project and for this thesis.

**Validation as a part of the Sec6Net project** — Originally, identity graphs were developed for SLIS (the LI system developed as a part of the Sec6Net project) under my supervision. Although the Sec6Net identity graphs [82, Chapter 5] support only a part of the operations available in identity graphs as described by this thesis, the deployment in many networks in the Sec6Net project showed the usefulness of the idea behind identity linking in identity graphs. In each deployment, one of the important tasks was the validation of correct linkage. As a part of the Sec6Net project, identity graphs were presented to Czech LEA officers during several demo sessions.

The generality of the identity graphs was proven by many partial identity detectors. Under my supervision, the Sec6Net project developed tools that reveal identity-related information from ten network protocols, two SDN controllers, and from clock-skew-based identification.

**Bachelor and diploma thesis employing identity graphs** — I supervised a bachelor thesis [53] that developed an inaccurate partial identity detector that used HTTP headers to identify browser profiles and consequently the user. Additionally, I supervised a diploma thesis [92] that focused on tunnelling protocols and NAT. Finally, I supervised a diploma thesis [47] that developed an authentication web page that acts as a partial identity detector.

**Validation of identity graphs defined by this thesis** The application of the original identity graphs for inaccurate partial identity detectors (clock-skew-based identification and browser fingerprinting) revealed that some limits for transitivity of the linkability of identifiers are required. The constraint function defined by Formula 6.17 allows limiting linkability of identifiers learnt from inaccurate partial identity detectors. The following experiment evaluates the applicability of the inaccuracy in identity graphs.

The experiment simulates an IPv6-enabled network of a small internet service provider. The internet service provider has 20 customer that accesses the network with 112 devices during the simulated period. Each device authenticates its MAC address with a RADIUS username belonging to the customer, each device leases an IPv4 address and generates IPv6 address. Additionally, each device can generate privacy extension IPv6 addresses [68] at any moment. During the experiment, all address assignments were monitored and an identity graphs containing the assignments was created.

During the experiment, we compared the identity graph created from information available in local and remote monitoring.

The local monitoring employs accurate partial identity sources. Remote monitoring employs inaccurate clock-skew-based identification. Figure 6.1 shows the average number of linkable IP address to an IPv4 address. For local monitoring, the average number is correct. For remote monitoring, the average number of linked IP addresses is only about 5-times higher than in the local monitoring in the worst case. Consequently, this example shows the benefits of the application of inaccuracy threshold. Although it does not provide precise results for LI, it can help a forensic investigator to limit the number of linked IP addresses.

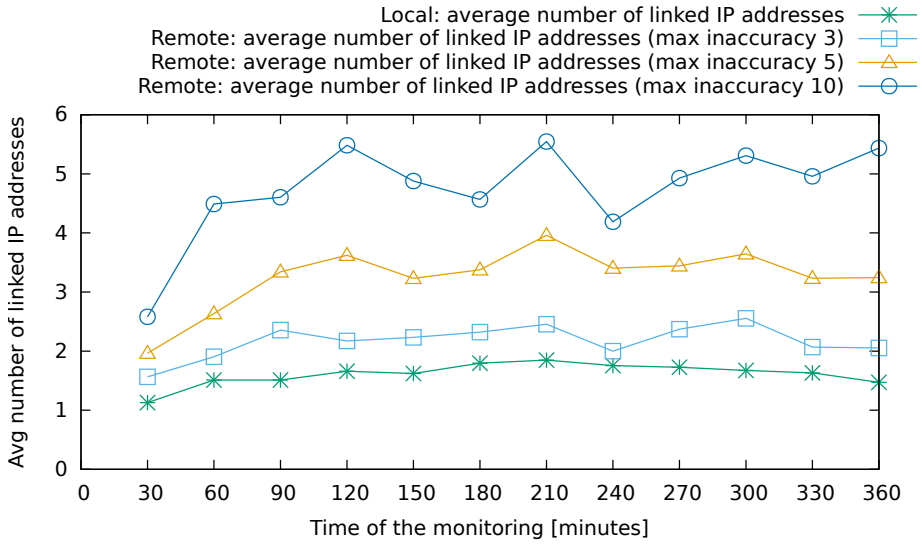



Figure 6.1: The average number of linkable IP addresses to each active IPv4 address during the simulated monitoring.

The example of the simulated network of a small internet service provider also validates the benefits of the time information in identity graphs. It is possible to query information from the past, which is valuable to forensic investigators.

## 6.5 Chapter conclusion

Identity graphs defined in this chapter provide operations that allow an LI system or an investigator to link identifiers based on scope, time, and accuracy. Identity graphs are a framework that allows cross-layer linkability of partial identifiers detected by various partial identity detectors distributed in the network. The operations in identity graph can be extended by defining new constraint function applicable in Formula 6.1 or by extending the set *Categories* or other components of identity graphs.

Identity graphs support various partial identity detectors including traffic analysers, log analysers, and inaccurate partial identity detectors. During this PhD research, identity graphs were constructed from more than 15 different partial identity detectors [47, 53, 82, 92]. In each deployment, the correct linkage in identity graphs was validated.

Identity graphs defined in this chapter validates the Hypothesis 3.

# Chapter 7

# Conclusion

This thesis describes identification in modern computer networks compatible with LI. Several challenges arise for LI in modern networks. (1) The shortage of IPv4 addresses caused a massive deployment of NAT that conserve the IPv4 address space. Consequently, several devices can share a single IPv4 address or a pool of IPv4 addresses. (2) Computers regenerate IPv6 addresses frequently. Additionally, SLAAC, the default IPv6 address assignment, is decentralised. Hence, there is not a single point in the network that contains the address assignment information. (3) Currently, IPv6-only deployment is rare. Dual stack networks can multiplex a single session through both IPv4 and IPv6. (4) LI based on application layer identifiers is becoming more significant. (5) Legal requirements mandate that a small change in the wording of a warrant can influence the traffic to be intercepted. Chapter 3 introduces the challenges in detail.

The IPv6 address assignment tracking based on timed transducers provides a mechanism that detects IPv6 address assignments in networks with MLD snooping (see Chapter 4), which confirms Hypothesis 1. The IPv6 address assignment tracking monitors messages in the network and joins all solicited-node multicast groups in the network. The core of the tracker is formalised as the timed transducer that creates messages that (1) form the basis for IRI records sent to LEA and (2) enable the construction of identity graphs.

The proposed IPv6 address tracking (1) detects all IPv6 addresses of each node on the LAN, (2) signals newly assigned addresses immediately, (3) does not poll routers in the network, and (4) detects that an IPv6 address was dropped. However, the biggest downside of the proposed IPv6 address assignment tracking is its reliability on the visibility of ND traffic. Each lost message can create inconsistencies between the state of the timed transducer and the network. If desired, the IPv6 address assignment tracking can be further extended to validate the consistency, however, this thesis focused on a method that does not inject additional queries to end hosts and detects the IPv6 address assignments immediately.

This thesis also focuses on clock-skew-based identification for a remote computer identification as related research indicates that unique short-term identification is possible [50, 59, 93] and that passive clock skew evaluation is possible [59], see Chapter 5. However, this PhD research has revealed that NTP influences TCP timestamps [81]. The study of real network clock skew distribution shows that most of the real clock skew estimates are close to 0 ppm. Consequently, a short-term fingerprinter cannot reliably identify computers solely by clock skew. Hence, we rejected Hypothesis 2.

Nevertheless, this PhD research provides additional knowledge about the clock-skew-based fingerprinting: (1) the formal requirement for a clock skew measurement, (2) the discovery of the influence of NTP on clock skew, (3) the method that passively links IPv4

and IPv6 addresses of the same computer, (4) the guide that enables a computer to mimic clock skew of another computer, and (5) the study of short-term fingerprinting in a moderately sized dual stack network.

Section 6.3 defines identity graphs and operations in identity graphs that can link partial identities with respect to the warrant wording, time, and inaccuracy restrictions. Identity graphs can be extended by additional operations or categories of identifiers. Identity graphs validate Hypothesis 3.

## 7.1 Future work

Although this thesis provides answers to hypotheses, new questions arise. This section provides directions for possible future research.

For IPv6 address assignment tracking, the biggest challenge is to remove the dependency on ND traffic. One option is to develop a different method based on an entirely new idea. For example, a router vendor can add an option that automatically signals a new entry in an NC. Another option is to combine ND tracking with another method such as NC polling. The idea is to let time transducers detect address assignments in ND traffic and validate the bindings by NC polling. If a timed transducer misses a new assignment, the NC polling learns the assignment with a delay in case there is some ingress traffic for the IPv6 address.

For clock-skew-based identification, one of the open problems is the behaviour of mobile devices. Chapter 5 notes that Apple devices exhibit an abnormal clock skew in TCP timestamps. Why? We did not focus on Android as we did not have access to a representative number of devices. However, our experiments show bigger instability of clock skew compared to other Linux distributions.

This PhD research has focused on identification based on passive short-term clock skew estimation. Beverly and Berger [6] combine clock skew fingerprinting and TCP stack fingerprinting. A question for possible future research is if clock skew fingerprinting can be combined with another fingerprinting method so that a passive short-term remote fingerprinting is possible.

A privacy-related research in clock skew (and hidden identifiers in general) can focus on measures to remove the hidden identifiers. Can the time of the computer be completely decoupled from timestamps sent outside the computer? Can a computer display accurate time to the user and mimic an arbitrary clock skew to the network?

Identity graphs incorporate inaccuracy. However, this thesis does not specify inaccuracy that applies to more inaccurate partial identity detectors simultaneously. Future research can provide a unified methodology that evaluates the inaccuracy of a partial identity detector and inaccuracy of a specific linkage between two partial identities.

The FIDIS reported that “virtually any commonly used protocol reveals identifying and linkable information usable for profiling” [32]. The privacy-related question is if a user can prevent such linking.

# Bibliography


[1] 6Lab. Web statistics, 2015. Available online at http://6lab.cz/live-statistics/web/, last visit: 2015-08-12.

[2] Limbesh B. Aal, Jignesh N. Parmar, Vishvesh R. Patel, and Dhrubo Jyoti Sen. Whatsapp, Skype, Wickr, Viber, Twitter and Blog are Ready to Asymptote Globally from All Corners during Communications in Latest Fast Life. *Research Journal of Science and Technology*, 6(2):101–116, 2014. ISSN 0975-4393.

[3] AQSACOM. Lawful Interception for IP Network, 2012. White Paper.

[4] Tuomas Aura and Alf Zugenmaier. Privacy, control and internet mobility. In *Security Protocols*, Lecture Notes in Computer Science, pages 133–145. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-40925-0. LNCS 3957.

[5] Abdullah Azfar, Kim-Kwang Raymond Choo, and Lin Liu. A study of ten popular android mobile VoIP applications: Are the communications encrypted? In *Proceedings of the 2014 47th Hawaii International Conference on System Sciences*, HICSS '14, pages 4858–4867, Washington, DC, USA, 2014. IEEE Computer Society. ISBN 978-1-4799-2504-9.

[6] Robert Beverly and Arthur Berger. Server siblings: Identifying shared IPv4/IPv6 infrastructure via active fingerprinting. In *Passive and Active Measurement*, Lecture Notes in Computer Science 8995, pages 149–161. Springer International Publishing, 2015. ISBN 978-3-319-15508-1.

[7] Karthikeyan Bhargavan and Carl A. Gunter. Network event recognition. *Formal Methods in System Design*, 27:213–251, 2005. ISSN 0925-9856.

[8] Jim Bound, Bernie Volz, Ted Lemon, Charles E. Perkins, and Mike Carney. *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*. IETF, 2003. RFC3315 (Proposed Standard).

[9] Eoghan Casey. *Digital Evidence and Computer Crime: Forensic Science, Computers and the Internet*. Academic Press, Elsevier Inc., USA, 2011. ISBN 978-0-12-374268-1. Third Edition.

[10] Kiran K. Chittimaneni, Merike Kaeo, and Eric Vyncke. *Operational Security Considerations for IPv6 Networks*. Internet Engineering Task Force, 2017. Internet Draft version 11 (Work in progress).

[11] Morten Jagd Christensen, Karen Kimball, and Frank Solensky. *Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches*. IETF, 2006. RFC 4541 (Informational).

[12] Lorenzo Colitti, Vint Cerf, Stuart Cheshire, and David Schinazi. *Host address availability recommendations*. IETF, 2016. RFC 7934 (Best Current Practice 204).

[13] Constitutional Order of the Czech Republic. Constitutional Act No. 2/1993 Coll. Article 1 – *Listina základních práv a svobod* (Charter of Fundamental Rights and Freedoms).

[14] Alissa Cooper, Fernando Gont, and Dave Thaler. *Security and Privacy Considerations for IPv6 Address Generation Mechanisms*. IETF, 2016. RFC 7721 (Informational).

[15] Council of Europe. Convention on Cybercrime, 2001. ETS No. 185, Budapest Convention.

[16] Matt Crawford. *Transmission of IPv6 Packets over Ethernet Networks*. IETF, 1998. RFC 2464 (Proposed Standard).

[17] Marius Cristea and Bogdan Groza. Fingerprinting smartphones remotely via ICMP timestamps. *IEEE Communications Letters*, 17(6):1081–1083, 2013. ISSN 1089-7798.

[18] Eric Cronin, Micah Sherr, and Matt Blaze. On the (un)reliability of eavesdropping. *International Journal of Security and Networks*, 3:103–113, 2008. ISSN 1747-8405.

[19] Stephen Deering and Robert Hinden. *Internet Protocol, Version 6 (IPv6) Specification*. IETF, 1998. RFC 2460 (Draft Standard).

[20] Amogh Dhamdhere, Matthew Luckie, Bradley Huffaker, kc claffy, Ahmed Elmokashfi, and Emile Aben. Measuring the deployment of IPv6: topology, routing and performance. In *Proceedings of the 2012 ACM conference on Internet measurement conference*, IMC '12, pages 537–550, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1705-4. Boston, Massachusetts, USA.

[21] Ralph Droms. *Dynamic Host Configuration Protocol*. IETF, 1997. RFC 2131 (Draft Standard).

[22] ETSI. *ETSI TR 101 943: Telecommunications security; Lawful Interception (LI); Concepts of Interception in a generic Network Architecture*. European Telecommunications Standards Institute, 2001. Version 1.1.1.

[23] ETSI. *ETSI TR 101 944: Telecommunications security; Lawful Interception (LI); Issues on IP Interception*. European Telecommunications Standards Institute, 2001. Version 1.1.2.

[24] ETSI. *ETSI ES 201 158: Telecommunications security; Lawful Interception (LI); Requirements for network functions*. European Telecommunications Standards Institute, 2002. Version 1.2.1.

[25] ETSI. *ETSI TR 102 528: Lawful Interception (LI); Interception domain Architecture for IP networks*. European Telecommunications Standards Institute, 2006. Version 1.1.1.

[26] ETSI. *ETSI TR 101 331: Lawful Interception (LI); Requirements of Law Enforcement Agencies*. European Telecommunications Standards Institute, 2009. Version 1.3.1.

[27] ETSI. *ETSI TR 102 232-3: Lawful Interception (LI); Handover Interface and Service-Specific Details (SSD) for IP delivery; Part 3: Service-specific details for internet access services*. European Telecommunications Standards Institute, 2009. Version 2.2.1.

[28] ETSI. *ETSI TR 101 671: Lawful Interception (LI); Handover interface for the lawful interception of telecommunications traffic*. European Telecommunications Standards Institute, 2010. Version 3.6.1.

[29] ETSI. *ETSI TS 102 232-2: Lawful Interception (LI); Handover Interface and Service-Specific Details (SSD) for IP delivery; Part 2: Service-specific details for E-mail services*. European Telecommunications Standards Institute, 2010. Version 2.5.1.

[30] ETSI. *ETSI TS 102-232-5: Lawful Interception (LI); Handover Interface and Service-Specific Details (SSD) for IP delivery; Part 5: Service-specific details for IP Multimedia Services*. European Telecommunications Standards Institute, 2010. Version 2.5.1.

[31] Cyrus Farivar. Cisco attributes part of lowered earnings to China's anger toward NSA. Ars Technica, 2013. Available online at http://arstechnica.com/business/2013/11/cisco-attributes-part-of-lowered-earnings-to-chinas-anger-towards-nsa/, last visit: 2015-01-16.

[32] FIDIS project. D3.8: Study on protocols with respect to identity and identification — an insight on network protocols and privacy-aware communication, 2008. Marit Hansen and Ammar Alkassar (ed.), Version 0.8. Available online at http://www.fidis.net/resources/fidis-deliverables/hightechid/#c2216, last visit: 2017-05-22.

[33] Ondřej Filip. Za rok přibylo v ČR více než 100 tisíc uživatelů IPv6, 2013. Blog of CZ.NIC staff. Available online at https://blog.nic.cz/2013/10/23/za-rok-pribylo-v-cr-vice-nez-100-tisic-uzivatelu-ipv6/ (in Czech), last visit: 2015-08-12.

[34] Barbora Franková. Computer identity based on its internal clock skew, 2013. Bachelor's thesis, Brno University of Technology, CZ (in Czech). Supervisor Libor Polčák.

[35] Barbora Franková. Lawful interception in software defined networks, 2015. Master's thesis, Brno University of Technology, CZ (in Czech). Supervisor Libor Polčák.

[36] Fernando Gont and Tim Chown. *Network Reconnaissance in IPv6 Networks*. IETF, 2016. RFC 7707 (Informational).

[37] Stephen Groat, Matthew Dunlop, Randy Marchany, and Joseph Tront. The privacy implications of stateless IPv6 addressing. In *Cyber Security and Information Intelligence Research*, pages 52:1–52:4, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0017-9. Oak Ridge, Tennessee.

[38] Stephen Groat, Matthew Dunlop, Randy Marchany, and Joseph Tront. What DHCPv6 says about you. In *2011 World Congress on Internet Security*, pages 146–151, London, UK, 2011. ISBN 978-1-4244-8879-7.

[39] Jonathan L. Gross and Jay Yellen. *Graph Theory and Its Applications*. Chapman & Hall/CRC, Taylor & Francis Group, Boca Raton, USA, 2006. ISBN 978-1-58488-505-4. Second Edition.

[40] Matěj Grégr. *User accounting in next generation networks*. PhD thesis, Brno University of Technology, Faculty of Information Technology, 2016. URL http://www.fit.vutbr.cz/study/DP/PD.php?id=448.

[41] Matěj Grégr, Petr Matoušek, Tomáš Podermański, and Miroslav Švéda. Practical IPv6 Monitoring - Challenges and Techniques. In *Symposium on Integrated Network Management*, pages 660–663, Dublin, Ireland, 2011. IEEE CS. ISBN 978-1-4244-9220-6.

[42] Uwe Hansmann, Lothar Merk, Martin S. Nicklous, and Thomas Stober. *Pervasive Computing*. Springer-Verlag Berlin Heidelberg, Germany, 2003. ISBN 3-540-00218-9. Second Edition.

[43] Clive Harfield and Karen Harfield. *Covert Investigation*. Oxford University Press Inc, New York, NY, USA, 2008. ISBN 978-0-19-954962-7. Second Edition.

[44] Robert M. Hinden and Stephen E. Deering. *IP Version 6 Addressing Architecture*. IETF, 2006. RFC 4291 (Draft Standard).

[45] Paul Hoffman and Kornel Terplan. *Intelligence Support Systems: Technologies for Lawful Intercepts*. Auerbach Publications, U.S., 2006. ISBN 978-0-8493-2855-8.

[46] Martin Holkovič. Identity Detection in TCP/IP Architecture, 2013. Bachelor's thesis, Brno University of Technology, CZ (in Slovak). Supervisor Libor Polčák.

[47] Martin Holkovič. SDN Controlled According to User Identity, 2015. Master's thesis, Brno University of Technology, CZ. Supervisor Libor Polčák.

[48] Martin Holkovič and Libor Polčák. Neighbor discovery tracker – ndtrack, 2013. https://www.fit.vutbr.cz/~ipolcak/prods.php?id=308.

[49] Radek Hranický. Additions to lawful interception system, 2014. Master's thesis, Brno University of Technology, CZ (in Czech). Supervisor Libor Polčák).

[50] Ding-Jie Huang, Kai-Ting Yang, Chien-Chun Ni, Wei-Chung Teng, Tien-Ruey Hsiang, and Yuh-Jye Lee. Clock skew based client device identification in cloud environments. In *Advanced Information Networking and Applications*, pages 526–533, 2012.

[51] *Internet Protocol*. Information Sciences Institute University of Southern California, IETF, 1981. RFC 791 (Internet Standard).

[52] Suman Jana and Sneha K. Kasera. On fast and accurate detection of unauthorized wireless access points using clock skews. *IEEE Transactions on Mobile Computing*, 9(3):449–462, 2010. ISSN 1536-1233.

[53] Jakub Jeleň. HTTP-request-based identification, 2015. Bachelor's thesis, Brno University of Technology, CZ (in Slovak). Supervisor Libor Polčák.

[54] Jakub Jirásek. Computer identification using time information, 2012. Master's thesis, Brno University of Technology, CZ (in Czech). Supervisor Libor Polčák.

[55] Christophe Kalt. *Internet Relay Chat: Architecture*. IETF, 2000. RFC 2810 (Informational).

[56] Christophe Kalt. *Internet Relay Chat: Client Protocol*. IETF, 2000. RFC 2812 (Informational).

[57] Juhoon Kim, Nadi Sarrar, and Anja Feldmann. Watching the IPv6 takeoff from an IXP's viewpoint. Technical report, 2014. Forschungsberichte der Fakultät IV Elektrotechnik und Informatik, Technische Universität Berlin, Bericht-Nummer: 2014-01, Berlin, DE.

[58] John C. Klensin. *Simple Mail Transfer Protocol*. IETF, 2008. RFC 5321 (Draft Standard).

[59] Tadayoshi Kohno, Andre Broido, and Kimberly C. Claffy. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2): 93–108, 2005. ISSN 1545-5971.

[60] Julius Kriukas. addrwatch: A tool similar to arpwatch for IPv4/IPv6 and ethernet address pairing monitoring, 2012. https://github.com/fln/addrwatch.

[61] Susan Landau. *Surveillance or security? : the risks posed by new wiretapping technologies*. The MIT Press, Cambridge, MA, USA, 2010. ISBN 978-0-262-01530-1.

[62] Fabian Lanze, Andriy Panchenko, Benjamin Braatz, and Andreas Zinnen. Clock skew based remote device fingerprinting demystified. In *Global Communications Conference*, pages 813–819, 2012.

[63] Law Order of the Czech Republic. Act No. 127/2005 Coll. and its subsequent amendments and additions *Zákon o elektronických komunikacích* accompanied by the Decree No. 336/2005 Coll. (In Czech).

[64] Louis Mamakos, Kurt Lidl, Jeff Evarts, David Carrel, Dan Simone, and Ross Wheeler. *A Method for Transmitting PPP Over Ethernet (PPPoE)*. IETF, 1999. RFC 2516 (Informational).

[65] David L. Mills, Jim Martin, Jack Burbank, and William Kasch. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. IETF, 2010. RFC 5905 (Proposed Standard).

[66] Nick 'Sharkey' Moore. *Optimistic Duplicate Address Detection (DAD) for IPv6*. IETF, 2006. RFC 4429 (Proposed Standard).

[67] Steven J. Murdoch. Hot or not: Revealing hidden services by their clock skew. In *Computer and Communications Security*, pages 27–36, New York, NY, USA, 2006. ACM. ISBN 1-59593-518-5. Alexandria, Virginia, USA.

[68] Thomas Narten, Richard Draves, and Suresh Krishnan. *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*. IETF, 2007. RFC 4941 (Draft Standard).

[69] Thomas Narten, Erik Nordmark, William Allen Simpson, and Hesham Soliman. *Neighbor Discovery for IP version 6 (IPv6)*. IETF, 2007. RFC 4861 (Draft Standard).

[70] Konstantina Papagiannaki, Sue Moon, Chuck Fraleigh, Patrick Thiran, Fouad Tobagi, and Christophe Diot. Analysis of measured single-hop delay from an operational backbone network. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 535–544, 2002.

[71] Andreas Pfitzmann and Marit Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. Technical report, 2010. Version 0.34, Available online at https://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf.

[72] Libor Polčák. Challenges in identification in future computer networks. In *ICETE 2014 Doctoral Consortium*, pages 15–24. SciTePress - Science and Technology Publications, 2014.

[73] Libor Polčák and Barbora Franková. On reliability of clock-skew-based remote computer identification. In *International Conference on Security and Cryptography*. SciTePress - Science and Technology Publications, 2014. Vienna, AT.

[74] Libor Polčák and Barbora Franková. Clock-skew-based computer identification: Traps and pitfalls. *Journal of Universal Computer Science*, 21(9):1210–1233, 2015. ISSN 0948-6968.

[75] Libor Polčák and Martin Holkovič. Behaviour of various operating systems during SLAAC, DAD, and ND, 2013. http://6lab.cz/?p=1691.

[76] Libor Polčák, Tomáš Martínek, Radek Hranický, Stanislav Bárta, Martin Holkovič, Barbora Franková, and Petr Kramoliš. Sec6Net Identity Management System, 2011–2014. https://www.fit.vutbr.cz/~ipolcak/prods.php.en?id=399&notitle=1.

[77] Libor Polčák, Tomáš Martínek, Radek Hranický, Stanislav Bárta, Martin Holkovič, Barbora Franková, and Petr Kramoliš. Sec6Net Lawful Interception System, 2011-2014. https://www.fit.vutbr.cz/~ipolcak/prods.php.en?id=397&notitle=1.

[78] Libor Polčák, Martin Holkovič, and Petr Matoušek. A New Approach for Detection of Host Identity in IPv6 Networks. In *Data Communication Networking*, pages 57–63. SciTePress - Science and Technology Publications, 2013. ISBN 978-989-8565-72-3. Reykjavík, IS.

[79] Libor Polčák, Martin Holkovič, and Petr Matoušek. Host Identity Detection in IPv6 Networks. In *Communications in Computer and Information Science*. Springer Berlin Heidelberg, DE, 2014.

[80] Libor Polčák, Radek Hranický, and Tomáš Martínek. On identities in modern networks. *The Journal of Digital Forensics, Security and Law*, 2014(2):9–22, 2014. ISSN 1558-7215.

[81] Libor Polčák, Jakub Jirásek, and Petr Matoušek. Comments on "Remote physical device fingerprinting". *IEEE Transactions on Dependable and Secure Computing*, 11 (5):494–496, 2014. ISSN 1545-5971. Los Alamitos, CA, USA, US.

[82] Libor Polčák, Tomáš Martínek, Radek Hranický, Stanislav Bárta, Martin Holkovič, Barbora Franková, and Petr Kramoliš. Sec6net: Lawful interception group summary. Technical report, 2014. Faculty of Information Technology Brno University of Technology, FIT-TR-2014-07, Brno, CZ (in Czech).

[83] Libor Polčák, Leo Caldarola, Davide Cuda, Marco Dondero, Domenico Ficara, Barbora Franková, Martin Holkovič, Amine Choukir, Roberto Muccifora, and Antonio Trifilo. High level policies in SDN. In *E-Business and Telecommunications*, volume 2016, pages 39–57. Springer International Publishing, 2016. ISBN 978-3-642-35754-1.

[84] Kai Rannenberg, Denis Royer, and André Deuker. Introduction. In *The Future of Identity in the Information Society*, pages 1–11. Springer Berlin Heidelberg, 2009. ISBN 978-3-540-88480-4.

[85] Yakov Rekhter, Robert G Moskowitz, Daniel Karrenberg, Geert Jan de Groot, and Eliot Lear. *Address Allocation for Private Internets*. IETF, 1996. RFC 1918 (Best Current Practice 5).

[86] Carl Rigney, Allan C. Rubens, William Allen Simpson, and Steve Willens. *Remote Authentication Dial In User Service (RADIUS)*. IETF, 2000. RFC 2865 (Draft Standard).

[87] Réseaux IP Européens Network Coordination Centre (RIPE NCC). IPv4 Exhaustion, 2012. Available online at http://www.ripe.net/internet-coordination/ipv4-exhaustion, last visit: 2015-04-10.

[88] Peter Saint-Andre. *Extensible Messaging and Presence Protocol (XMPP): Core*. IETF, 2011. RFC 6120 (Proposed Standard).

[89] Surasak Sanguanpong and Kasom Koht-Arsa. A design and implementation of dual-stack aware authentication system for enterprise captive portal. In *9th International Conference on Network and Service Management (CNSM)*, pages 118–121, Zürich, Switzerland, 2013. ISBN 978-3-901882-53-1.

[90] Quirin Scheitle, Oliver Gasser, Minoo Rouhi, and Georg Carle. Large-scale classification of ipv4-ipv6 siblings with nonlinear clock skew. *Computing Research Repository*, abs/1610.07251, 2016. URL http://arxiv.org/abs/1610.07251.

[91] Sebastian Schrittwieser, Peter Frühwirt, Peter Kieseberg, Manuel Leithner, Martin Mulazzani, Markus Huber, and Edgar Weippl. Guess who's texting you? evaluating the security of smartphone messaging applications. In *Proceedings of the Network and Distributed System Security Symposium*. The Internet Society, 2012.

[92] Michal Šeptun. Identities in tunelled networks and during network address translation, 2015. Master's thesis, Brno University of Technology, CZ (in Czech). Supervisor Libor Polčák.

[93] Swati Sharma, Alefiya Hussain, and Huzur Saran. Experience with heterogenous clock-skew based device fingerprinting. In *Workshop on Learning from Authoritative Security Experiment Results*, pages 9–18. ACM, 2012. ISBN 978-1-4503-1195-3. Arlington, Virginia.

[94] The Council of the European Union. COUNCIL RESOLUTION of 17 January 1995 on the lawful interception of telecommunications (96/C 329/01), 1996.

[95] The Internet Assigned Numbers Authority. IANA IPv4 address space registry, 2014. Available online at http://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml, last visit: 2015-04-10.

[96] Susan Thomson, Thomas Narten, and Tatuya Jinmei. *IPv6 Stateless Address Autoconfiguration*. IETF, 2007. RFC 4862 (Draft Standard).

[97] Jenny Torres, Michele Nogueira, and Guy Pujolle. A survey on identity management for the future network. *IEEE Communications Surveys & Tutorials*, 15 (2):787–802, 2013. ISSN 1553-877X.

[98] Utimaco TS GmbH. Lawful interception in the digital age: Vital elements of an effective solution, 2014. White Paper.

[99] Vodafone Group Plc. Vodafone Group Plc Sustainability Report 2013/14, 2014. Available online at http://www.vodafone.com/content/dam/sustainability/2014/pdf/vodafone_full_report_2014.pdf, last visit: 2015-01-16.

[100] Eric Vyncke, Pascal Thubert, Eric Levy-Abegnoli, and Andrew Yourtchenko. *Why Network-Layer Multicast is Not Always Efficient At Datalink Layer*. Internet Engineering Task Force, 2014. Internet Draft version 01 (Expired Work in progress).

[101] Jason Weil, Victor Kuarsingh, Chris Donley, Christopher Liljenstolpe, and Marla Azinger. *IANA-Reserved IPv4 Prefix for Shared Address Space*. IETF, 2012. RFC 6598 (Best Current Practice 153).

[102] Dan Wing and Andrew Yourtchenko. *Happy Eyeballs: Success with Dual-Stack Hosts*. IETF, 2012. RFC 6555 (Proposed Standard).

[103] Menghui Yang and Hua Liu. Implementation and performance of VoIP interception based on SIP session border controller. *Telecommunication Systems*, 55(3):345–361, 2014. ISSN 1018-4864.

[104] Andrew Yourtchenko and Erik Nordmark. *A survey of issues related to IPv6 Duplicate Address Detection*. Internet Engineering Task Force, 2015. Internet Draft version 01 (Expired Work in progress).

[105] Sebastian Zander and Steven J. Murdoch. An improved clock-skew measurement technique for revealing hidden services. In *Proceedings of the 17th Conference on Security Symposium*, pages 211–225, Berkeley, CA, USA, 2008. USENIX Association.