Biswas, Arnab Kumar and Ghosal, Dipak and Nagaraja, Shishir (2017) A survey of timing channels and countermeasures. ACM Computing Surveys, 50 (1). ISSN 1557-7341 , http://dx.doi.org/10.1145/3023872

A Survey of Timing Channels and Countermeasures

Arnab Kumar Biswas, Indian Institute of Science
Diapk Ghosal, University of California Davis
Shishir Nagaraja, Lancaster University

A timing channel is a communication channel that can transfer information to a receiver/decoder by modulating the timing behavior of an agent. Examples of this agent include the inter-packet delays of a packet stream, re-ordering packets in a packet stream or resource access time of a cryptographic module. The advances in information theory and the availability of high performance computing systems interconnected by high speed networks, have spurred interest and development of various types of timing channels. With the emergence of complex timing channels, novel detection and prevention techniques are also being developed to counter them. In this paper we provide a detailed survey of timing channels broadly categorized into network timing channel in which communicating entities are connected by a network and in-system timing channel in which the communicating entities are within a computing system. This survey builds upon the last comprehensive survey by [Zander et al. 2007] and considers all the three canonical applications of timing channels namely, covert communication, timing side-channel, and network flow watermarking. We survey the theoretical foundations, the implementation, and the various detection and prevention techniques that have been reported in the literature. Based on the analysis of the current literature we articulate potential future research directions both in the design and applications of timing channels and their detection and prevention techniques.

Categories and Subject Descriptors: A.1 [**General Literature**]: INTRODUCTORY AND SURVEY; C.2.0 [**COMPUTER-COMMUNICATION NETWORKS**]: General—*Security and protection*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Unauthorized access*

General Terms: Security, Documentation

Additional Key Words and Phrases: Timing Channel, Covert Timing Channel Communication, Timing Side Channel, Network Flow Watermarking, Timing Channel Categories, Timing Channel Countermeasures

ACM Reference Format:
Arnab Kumar Biswas, Diapk Ghosal, and Shishir Nagaraja, 2016. A Survey of Timing Channels and Countermeasures. *ACM Comput. Surv.* 0, 0, Article 0 (February 2016), 43 pages.
DOI:http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

According to [Department of Defense Standard 1985], a covert channel is defined as a "communication channel that allows a process to transfer information in a manner that violates the system's security policy." Here the term system may refer to single computer system or a distributed system connected by network. The concept of covert channels was first published in [Lampson 1973] with a characterization of the confinement problem and methods to block some subtle information leakage paths. While the work was primarily conceptual in nature, the underlying concept has since grown in importance. Significant developments in information theory, coding theory, and the emergence of high performance systems interconnected by high-speed networks have

This work is supported by the PhD scholarship from MHRD, Government of India.
Author's addresses: Arnab Kumar Biswas, Department of Electronic Systems Engineering, Indian Institute of Science; Diapk Ghosal, Department of Computer Science, University of California Davis; Shishir Nagaraja, School of Computing and Communications, Lancaster University.

enabled covert channels in general, and timing channels in particular, to evolve from a conceptual idea into a potentially useful and practical tool.

In this paper, we present a survey of timing channels and their countermeasures. Timing channel is defined as a channel that can transfer information based on the modulation of some timing behavior of an agent. An example of this agent is the inter-packet delay (IPD) of a packet stream generated by a distributed application. Specifically, the transfer of information is achieved by modulating the IPDs to carry the information bits. Other examples of timing channel agents include the packet ordering in network, and the resource access time of a cryptographic module. In literature, the term covert timing channel (CTC) is used to refer to a timing channel that transfers covert messages over a network. In this paper, we use the term covert timing channel communication (CTCC) in place of CTC to emphasize its use as a communication channel to transfer messages. The term network flow watermarking (NFW) is used to refer to a timing channel that can mark network flows, while the term timing side channel (TSC) is used to refer to a timing channel that leaks information due to faulty/vulnerable operations. In this survey, we consider NFW and TSC along with CTCC as types of timing channels.

Timing channels have both legitimate and malicious applications. As an example of malicious application, CTCCs are used by criminals to ex-filtrate secret informations from an enterprise. Whereas, as an example of legitimate application, a network administrator can use CTCCs to hide network management related communication or to transfer authentication data. Further details of various CTCC applications (both malicious and legitimate) are given in [Zander et al. 2007; Archibald 2013]. An important new application of NFW is to detect stepping stones. Stepping stones refer to compromised nodes that are used as intermediate nodes in a network to launch the final attack. NFW can also be used to deanonymize an anonymous network [Dingledine et al. 2004]. These applications can be considered malicious or legitimate depending on the actual application scenario. TSCs are mainly used in malicious applications e.g. to extract secret cryptographic keys.

An earlier very comprehensive survey on network covert channels including CTCCs is given in [Zander et al. 2007]. The survey covered the literature during the period from 1987 through 2006 and reviewed all types of available implementations of network covert channels and their mitigation techniques. [Jaskolka et al. 2012] have surveyed the conditions for existence of various CTCCs available in literature. Another related survey on various CTCC detection techniques is presented in [Goher et al. 2012]. Both of these surveys ([Jaskolka et al. 2012; Goher et al. 2012]) are short and their coverages are also limited to existence conditions, and detection techniques respectively.

In this paper we primarily build on [Zander et al. 2007] and review different CTCC design, implementations, and detection/mitigation techniques since 2006. It is to be noted that CTCC is only a type of timing channel and other types of timing channel are also surveyed in this paper. We first categorize different timing channels based upon location i.e, network timing channel and in-system timing channel. Each category is then again categorized into implementations, and detection/prevention works against those implementations. In literature, different detection/prevention techniques against different timing channels are considered as threats to those timing channels. Extent of fulfillment of different timing channel requirements are considered during discussion of different implementation works. Even though NFW is a type of network timing channel, it is treated separately in a section because of its importance. It is also to be noted that TSCs are a type of timing channel and they are present not only inside systems but in network too. The counterpart of TSC in network is known as remote timing side channel (RTSC) and different RTSC techniques are surveyed in same section as other network timing channel techniques. Inclusion of TSC in the survey helps to complete the understanding of different timing channels that is started by CTCC and NFW. We do not cover various programming language and OS related or Virtual

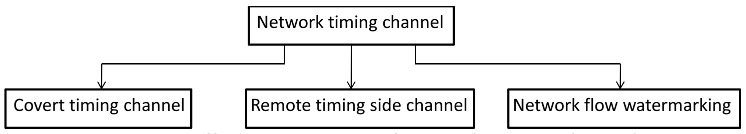


Fig. 1. Different categories of network timing channels.

machine and cloud related timing channels in this paper due to space constraint. To the best of our knowledge, this paper is the first attempt to survey both the NFW and the TSC along with the CTCC.

We provide preliminaries in Appendix A with definitions and pictorial models of different types of timing channels. We also discuss the requirements of timing channels.

1.1. Outline

The rest of the paper is organized as follows. In Section 2, we discuss the various network timing channel implementations and detection/mitigation techniques. Though NFW is a type of network timing channel, the various NFW techniques and threat models are discussed in Section 3. The various in-system timing channel implementation techniques and detection/prevention methods are discussed in Section 4. We conclude and suggest future research directions in Section 5.

2. NETWORK TIMING CHANNELS

Network timing channel can be of three types as shown in Figure 1 : Covert timing channel, Remote timing side channel, and Network flow watermarking (NFW). Among these three types covert timing channel communication (CTCC) based literatures are more in number. In this section, we discuss various CTCC related theoretical works, implementation techniques, and detection and prevention methods. We also discuss various studies related to remote timing side channel. NFW is discussed in Section 3.

2.1. Covert timing channel communication

In some publications the term steganographic timing channel is used in place of covert timing channel. From our perspective they are same because of their identical implementation techniques, characteristics, and goal. In this section steganographic timing channel and CTCC are used interchangeably.

2.1.1. Theory. Different types of theoretical works related to CTCC exist in literature. Some considers theoretical limits of maximum or achievable information transmission rate and channel capacity. Others consider a general model to represent all CTCCs.

In [Moskowitz and Miller 1992] authors have investigated the effect of noise on a CTCC. They have used Shannons information theory to quantify the information flow through the CTCC. It is noted in [Anantharam and Verdu 1996] that the channel capacity of a queue having exponentially distributed service time is $e^{-1}\mu$ nats/sec and it is also the minimum channel capacity considering all servers having service rate μ. Various aspects of Exponential Service Timing Channels (ESTC) are discussed and studied in detail in [Anantharam and Verdu 1996; Sundaresan and Verdu 2000a; 2000b; Sundaresan and Verdu 2006; Giles and Hajek 2002; Wagner and Anantharam 2005; Momcilovic 2006] with the discrete-time counterpart present in [Bedekar and Azizoglu 1998; Thomas 1997]. Single-server timing channel capacity's lower bounds are estimated in [Sellke et al. 2006] with the service time distributions of uniform, Gaussian, and truncated Gaussian.

Sellke et al. have given two lower bounds and an upper bound on the capacity of Bounded Service Timing Channels (BSTC) [Sellke et al. 2007]. BSTC has bounded support service time distributions. The notion of bounded support means that the service times follow $P(a < S_k < a + \Delta) = 1$ where $a, \Delta > 0$ are some constants and S_k is i.i.d service time [Sellke et al. 2007]. The authors have also shown that the uniform BSTC has the lowest capacity amongst different kinds of BSTCs for small support in-

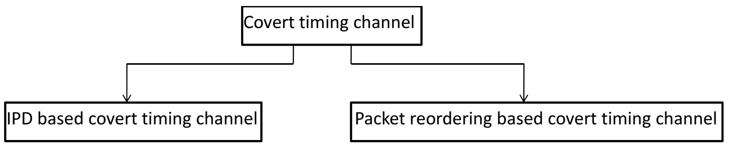

Covert timing channel
IPD based covert timing channel
Packet reordering based covert timing channel

Fig. 2. Two types of covert timing channels.

terval. Gorantla et al. have given an upper bound of the CTCC capacity from a high security level user (HU) to a low security level user (LU) through the full buffer NRL pump [Gorantla et al. 2010; Gorantla et al. 2012].

Yao et al. have presented the maximum information transmission rate in an on/off CTCC [Yao et al. 2009]. In an on/off CTCC information is transfered by transmitting or not transmitting a packet within a predetermined time period. The authors have divided the on/off CTCC into two types based on transmission IPD: deterministic CTCC and non-deterministic CTCC. They have mainly analyzed the non-deterministic on/off CTCC considering stable or slowly varying network delay characteristic. Zi et al. have proposed a procedure to determine the maximum transmission rate of an IPD based CTCC without actually implementing it [Zi et al. 2011]. For the method, they need to know the interval jitter characteristic in the network which they describe by a group of probability functions. They also need the packet loss probability, and the minimum sending interval without any congestion or packet loss. In their scheme the overt and covert senders are same i.e. the scheme is applicable to active CTCCs only.

Ezzeddine et al. have proposed codes to implement CTCC using queue-based codes and Shannon's encoding functions [Ezzeddine and Moulin 2009]. Sparse graph coset codes over non-binary finite fields can be used to approach capacity for queuing timing channels [Coleman and Kiyavash 2008a; 2008b]. Wang et al. have studied the theoretical limits of steganographic channel capacity and the required coding structure to achieve those limits [Wang and Moulin 2008].

Wang et al. have also noted that the perfect security requirement of steganographic timing channel is that the probability distribution of stegotext must be same as legitimate traffic distribution [Wang and Moulin 2008]. Works on achievable secrecy rate, and coding scheme to achieve maximum information rate are available. Dunn et al. have presented achievable secrecy rate of IPD based CTCC involving parallel queues [Dunn et al. 2009]. They have also proposed the necessary and sufficient condition to achieve some secrecy for a deterministic encoder.

2.1.2. Implementations. CTCC implementations are mainly of two types as shown in Figure 2 : IPD based and packet reordering based. IPD based CTCC is implemented by modulating the inter-packet delay of overt traffic based upon a CTCC encoding mechanism. Active and passive both type of CTCC can be implemented using IPD modulation. On the other hand, packet reordering based CTCC is implemented by encoding the covert messages in the order of packets in a flow or multiple flows. In case of multiple flows, the set of flows themselves also can be used to encode messages. Only active CTCC can be implemented using this type because passive CTCC will require a lot of buffer space to hold one or multiple traffic flows.

A) IPD based covert timing channel. Probably the first published work on CTCC in networking domain (specifically Local area network) is [Girling 1987]. The author has analyzed the capacity of CTCC in LAN and has provided some measures to reduce the bandwidth of CTCC. Different requirements of an efficient CTCC i.e. non-detectability, non-disclosure, robustness, and high capacity are not considered in this early paper. A TCP-based CTCC called TCPScript is proposed in [Luo et al. 2008] by embedding covert messages into the TCP burst size (i.e. number of packets in a burst). In this manner they have tried to keep the TCP's normal burstiness patterns. Though authors have given different results for capacity as well for robustness against packet loss, packet reordering, and traffic shaping, no solution is given to improve them. They have also not considered non-detectability and non-disclosure requirements. Another IPD based CTCC over TCP/IP connection is proposed in [Sellke et al. 2009]. Sellke et al. have given separately non-detectable solution with low capacity, and high capacity solution without non-detectability attribute. Non-disclosure, and robustness requirements are not considered in their work. Kiyavash et al. have proposed an IPD based CTCC targeting interactive SSH traffic [Kiyavash and Coleman 2009]. They have modeled the traffic using two-state Markov Modulated Poisson Process (MMPP). Though authors have considered non-detectability requirement, no other requirements are considered in their work. Their solution is non-detectable only to Markov-modulated Poisson process testing.

A time-replay CTCC is proposed in [Cabuk 2006] where covert messages are transmitted by replaying a previously recorded sequence of timing intervals. They have partitioned the prerecorded sequence and the number of partitions are equal to the size of the message alphabet. Each partition is then associated with a symbol. A timing interval from a partition is randomly chosen to send the associated symbol. An example of time-replay CTCC with discussion is given in Appendix C. Non-detectability, and non-disclosure requirements are considered in the work but robustness, and high capacity requirements are not considered.

A number of steganographic timing channels are implemented by modulating IPD [Adam Aviv and Blaze 2011; Liu et al. 2010]. Adam et al. have considered non-detectability, and non-disclosure requirements but robustness, and high capacity requirements are not considered. Liu et al. have proposed an IPD based CTCC using spreading codes [Liu et al. 2010]. The use of spreading code makes the CTCC robust but with low capacity. The proposal is only applicable to independent and identically distributed (i.i.d.) IPDs. They have also shown that their CTCC is provably non-detectable.

Houmansadr et al. have proposed an IPD based CTCC called CoCo [Houmansadr and Borisov 2011a]. They have used different error correcting codes to increase robustness of their scheme. Apart from robustness, other requirements i.e. non-detectability, non-disclosure, and high capacity are also considered in the work. Convolutional codes, Block codes (Reed-Solomon and Golay codes), Turbo codes, and Low density party-check codes (LDPC) are used to show comparative results. In their scheme, both covert sender and receiver need to generate IPDs using a shared secret key. Huffman coding to compress the CTCC data is proposed in [Wu et al. 2012]. Though experimental results of non-detectability, and capacity are given, non-disclosure and robustness are not considered in their work. Stillman et al. have proposed to use a concatenated code (convolutional outer code over an inner copy code) for a CTCC implementation through Mix-firewalls [Stillman 2008]. They have not considered any requirement of an effective CTCC. Liu et al. have proposed an IPD based CTCC using spreading codes [Liu et al. 2009]. It is shown that the use of spreading code can ensure balance between CTCC non-detectability, robustness, and capacity. They have proposed to emulate the overt IPD distribution in covert traffic.

Gianvecchio et al. have proposed model-based CTCC to mimic the statistical properties of legitimate traffic [Gianvecchio et al. 2008]. They have proposed a framework consisting of filter, analyzer, encoder, and transmitter. The framework is discussed in more detail in Appendix D. Though they have considered non-detectability, and capacity requirements, other requirements i.e. robustness and non-disclosure are not considered.

Liu et al. also have proposed a CTCC with distribution matching [Liu et al. 2009; Liu et al. 2012]. Their method divides the overt traffic into fixed-length fragments and all the packet delays in a fragment are used to get histogram. This histogram distribution is matched after the covert messages are encoded into delays. Authors have considered all requirements of an efficient CTCC. Model-based CTCC with polynomial-time non-detectability and high capacity is proposed in [Ahmadzadeh and Agnew 2013]. This means that a polynomial-time statistical test cannot detect the CTCC. They have used trellis structure at the modulation stage of the transmitter and at the iterative demodulation stage of the receiver. They have also used an adaptive modulation scheme to improve the channel robustness without any loss of non-detectability. Archibald et al. have proposed a model based CTCC using Fountain codes as error correcting codes [Archibald and Ghosal 2012; Archibald 2013]. Model based CTCC helps to achieve non-detectability requirement and use of Fountain codes causes continuous generation of encoded symbols until receiver informs successful reception (ensuring robustness). That means a separate opposite channel is required from receiver to sender. They have also proposed to use IPD guard band (at sender side or receiver side) to reduce bit error rate (BER). The use of guard band causes degradation in channel covertness. Authors have also considered capacity and non-disclosure requirements in their work. Generally independent identically-distributed (iid) IPDs are used in model based CTCCs. It is shown that these CTCCs can be detected if the real IPDs are not iid [Zander et al. 2011]. Zander et al. have proposed a non-detectable CTCC using a portion of each IPD to embed covert data. This causes their CTCC to lack robustness and highly sensitive to channel noise. They have also proposed to use packet contents to generate random numbers that can help to establish synchronization between the covert parties.

Chen et al. have proposed a Phase Lock Loop (PLL) based synchronization scheme for CTCC implementation [Chen et al. 2011]. They have shown that their scheme's decoding rate remains high even after packet loss and out-of-order packet arrival. Apart from robustness, no other requirements are considered in their work. Lee et al. have proposed a CTCC in the physical layer using sub-microsecond modulation [Lee et al. 2014]. They have shown that their CTCC named Chupja is non-detectable, and robust with high capacity.

Various detection resistant CTCCs are also proposed in literature. Active CTCCs are regular in nature because they are generated by computer programs. To avoid detection because of regularity, Kothari et al. have proposed an active irregular CTCC called Mimic [Kothari and Wright 2013]. They have used two modules called "shape modeler" and "regularity modeler" to learn about shape and regularity properties of legitimate traffic and to generate Mimic traffic. Walls et al. have proposed a CTCC called Liquid that is resistant to entropy detection tests [Walls et al. 2011]. They have proposed to use a number of IPDs to smooth out the shape distortions apart from using some IPDs to embed covert messages. Both detection resistant works consider only non-detectability and capacity requirements; robustness and non-disclosure requirements are not considered. Different CTCC detection and prevention techniques are discussed later.

CTCC implementation targeting a particular application is also available in literature. Sun et al. have proposed a CTCC based identity authentication mechanism [Sun et al. 2012]. They have used overt IPDs to encode the authentication tags: long IPD as bit 1 and short IPD as bit 0. They have only tried to achieve the non-detectability requirement of the CTCC.

Zi et al. have proposed a passive CTCC where the covert sender dwells in an intermediate node (e.g. router, gateway) and modulates incoming IPDs to embed covert messages [Zi et al. 2010]. They have also proposed to use synchronization and error correction techniques based on frames for covert communication. They have only considered the robustness requirement in their work.

The main notable attributes of different IPD based CTCC works can be represented by the Table I.

B) Packet reordering based covert timing channel:. Ahsan et al. have proposed a CTCC based on packet ordering within the IPSec framework [Ahsan and Kundur 2002]. The different requirements of an efficient CTCC are not considered in this early paper. Another packet reordering based CTCC is proposed in [El-Atawy and Al-Shaer 2009]. They have proposed to increase reliability (robustness) of CTCC by embedding

Table I. Main notable attributes of IPD based CTCC works.

Main attributes		**IPD based CTCC works**
CTCC general model		[Shrestha et al. 2013; Shrestha et al. 2014]
Achievable secrecy rate		[Dunn et al. 2009]
Channel capacity	Coding scheme	[Coleman and Kiyavash 2008a; 2008b; Wu et al. 2012] [Wang and Moulin 2008; Ezzeddine and Moulin 2009]
	Theoretical bounds	[Sellke et al. 2007] [Gorantla et al. 2010; Gorantla et al. 2012]
	Maximum rate	[Yao et al. 2009; Zi et al. 2011]
Protocol specific (e.g. TCP, SSH) CTCC		[Girling 1987; Luo et al. 2008] [Sellke et al. 2009; Kiyavash and Coleman 2009]
Non-detectability	General	[Adam Aviv and Blaze 2011; Liu et al. 2010]
	Time replay	[Cabuk 2006]
	Model based	[Liu et al. 2009; Gianvecchio et al. 2008] [Liu et al. 2009; Liu et al. 2012] [Ahmadzadeh and Agnew 2013; Zander et al. 2011] [Archibald and Ghosal 2012; Archibald 2013]
	Physical layer	[Lee et al. 2014]
	Resistant to Regularity test	[Kothari and Wright 2013]
	Resistant to Entropy test	[Walls et al. 2011]
Robustness (use of ECC)		[Houmansadr and Borisov 2011a] [Stillman 2008; Liu et al. 2009]
PLL based synchronization		[Chen et al. 2011]
Identity authentication		[Sun et al. 2012]
Passive CTCC		[Zi et al. 2010]

covert messages using specific permutations of consecutive packets and to increase non-detectability by imitating real traffic distribution. They have also considered the high capacity requirement of CTCC. Another similar work is reported in [Chakinala et al. 2007]. Authors have proposed polynomial time optimal encoding and decoding algorithms to achieve maximum channel capacity. They have proposed formal models for packet re-ordering channels. They have proved the existence of Nash equilibrium after taking game theoretic approach to analyze the channel. In this theoretical work no solution is proposed to improve different requirements (except capacity) of a practical CTCC.

Till now packet reordering in a single flow is reported to build a CTCC but another type of CTCC is also present where packet reordering among multiple flows is used to build a CTCC. In [Luo et al. 2007; Luo et al. 2012a], authors have proposed a CTCC called Cloak by sending N packets in every round over X flows to transfer a message. They have proposed 10 encoding and decoding schemes to use different combinations of N and X. Their proposal is to exploit TCPs reliable transmission mechanism to provide robustness (against packet losses) to Cloak. They have also considered non-detectability and high capacity requirements of CTCC.

Inter-socket packet arrival order is also used to build a CTCC. In [Khan et al. 2009], authors have proposed a high capacity CTCC based on inter-socket packet arrival order using multiple active connections. Their proposed CTCC is shown to be non-detectable by regularity based tests. The reason is that the covert message is embedded in the order and sequence of different connections and not in the IPDs of a connection. Robustness and non-disclosure requirements are not considered in the work.

The main notable attributes of different packet reordering based CTCC works can be represented by the Table II.

C) CTCC implementations in wireless network:. CTCCs are implemented not only in wired network but in other types of networks also. For example, CTCC can be implemented in wireless network [Kiyavash et al. 2013; Radhakrishnan et al. 2013; Yue et al. 2014]. Kiyavash et al. have implemented a CTCC in carrier sense multiple access with collision avoidance (CSMA/CA) protocol with a triggering mechanism [Kiyavash et al. 2013]. Radhakrishnan et al. have implemented a CTCC on 802.11 network exploiting

Table II. Main notable attributes of packet reordering based CTCC works.

Main attributes	Packet reordering based CTCC works
Packets reorder in single flow	[Ahsan and Kundur 2002; El-Atawy and Al-Shaer 2009] [Chakinala et al. 2007]
Packets in a flow and multiple flows reorder	[Luo et al. 2007; Luo et al. 2012a]
Inter-socket packet arrival order	[Khan et al. 2009]

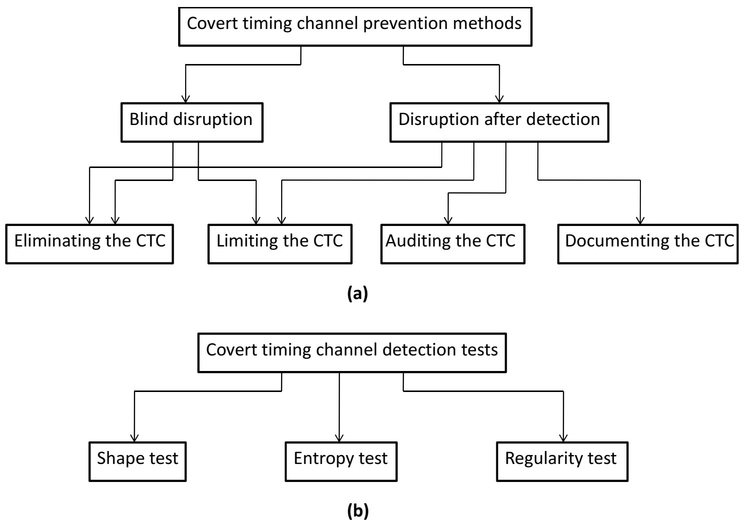


Fig. 3. Different types of (a) CTCC prevention methods, and (b) CTCC detection tests.

the random back-off in distributed coordination function (DCF) [Radhakrishnan et al. 2013]. Wireless CTCC is also implemented by varying CPU speed in mobile devices exploiting Android OS's performance governor [Yue et al. 2014]. There is a legitimate application using wireless CTCC reported in [Edwards et al. 2012]. They have proposed to use CTCC to detect wormhole attacks in Mobile ad hoc networks (MANETs). It is shown that the CTCC capacity is reduced under wormhole attack. They have used error correcting codes and have used the number of errors corrected to detect the presence of wormhole attack.

2.1.3. Detection and Prevention. There are two types of CTCC prevention methods - blind disruption and disruption after detection [Archibald 2013; Archibald and Ghosal 2014]. Blind disruption means disrupting the possible presence of covert timing channel without detecting it. According to [Zander et al. 2007] there are mainly four ways to counter CTCC after detecting it - 1) Eliminating the covert timing channel, 2) Limiting the covert timing channel, 3) Auditing the covert timing channel, and 4) Documenting the covert timing channel. In fact, elimination and limiting methods can also be launched blindly. Different CTCC prevention methods are shown in Figure 3(a). Authors in [Archibald 2013; Archibald and Ghosal 2014] have categorized the CTCC detection tests into three categories as shown in Figure 3(b) : 1) shape test like Kolmogorov Smirnov (KS) test, 2) entropy test like Kullbacke Leibler (KL) divergence test, and simple entropy test, and 3) regularity test like auto-correlation test. A test may belong to more than one category like corrected conditional entropy (CCE) test measures entropy and regularity both.

Archibald et al. have compared three CTCC detection test performances against three types of CTCCs : 1) IPD based CTCC, 2) model-based CTCC, and 3) time-replay CTCC [Archibald 2013; Archibald and Ghosal 2014]. They have also proposed a shape test called Welch's t-test and multi-feature Support Vector Machine (SVM) classifier for increasing the classification rate. Mou et al. have also proposed a CTCC detection method using SVM and wavelet transform [Mou et al. 2012]. They have also proposed a detection and classification method using a sliding window to detect different CTCCs in a single traffic.

Gianvecchio et al. have suggested that the change of entropy in a flow due to CTCC can be used to detect a CTCC [Gianvecchio and Wang 2007; 2011]. They have proposed an entropy test to detect CTCC by observing abnormal shape.

Mason et al. have presented a CTCC detection mechanism using entropy measurements on a Massively Parallel Processing Architecture (MPPA) called Ambric [Mason et al. 2010]. They have used this architecture to detect malware, CTCC, and to perform symmetric encryption.

Cabuk et al. have shown that IPDs in CTCC are more regular than legitimate channel [Cabuk 2006; Cabuk et al. 2009]. They have proposed to use compressibility to detect CTCC by detecting regularity in traffic. Details of compressibility metric are given in Appendix E.

There are few CTCC blind disruption techniques reported in literature. Wang et al. have proposed a traffic controller to control IPD based CTCC by randomly delaying traffic [Wang et al. 2009]. Their method does not detect the presence of any CTCC. Insertion of delays causes performance degradation of all traffic including legitimate ones. Another timing mitigator to limit the CTCC capacity is proposed in [Askarov et al. 2010]. They have also shown the trade off between the system performance and the achievable CTCC limit.

IPD based CTCC and packet reordering based CTCC can exist in a virtual private network (VPN) [Sadeghi et al. 2012]. To mitigate IPD based CTCC, Sadeghi et al. have proposed to implement traffic reshaping inside the Linux kernel [Sadeghi et al. 2012]. They have implemented IPsec anti-replay window as a packet buffer and packets are sorted using their Encapsulated Security Payload (ESP) sequence numbers. This helps to mitigate packet reordering based CTCC. Another work on CTCC mitigation in case of VPN is [Herzberg and Shulman 2013]. Here authors have assumed that man-in-the-end (MitE) malware runs within both ends of protected network and a man-in-the-middle (MitM) attacker is located on the path of communication. They have suggested to use traffic shaper similar to [Sadeghi et al. 2012] to modify the IPDs of sender's packets. They have also suggested to use error correcting codes in the sender to limit the packet reordering based CTCC from MitM to MitE. The receiver can recover original packets removing most if not all MitM covert messages.

Luo et al. have proposed two methods and one combined method to detect a TCP based CTCC called TCPScript [Luo et al. 2008]. This CTCC is also proposed by them in the same work. The first method is based on the TCP burst size because in TCPScript the burst size is dependent on the covert message. The second method is based on the delay between an arriving ACK packet and the next departing packet. This is because a TCPScript encoder sends two TCP bursts with a separation that is much higher than a normal inter-TCP burst separation. They have also proposed a third method that is a combination of the two earlier methods.

According to [Stillman 2008], any correlation between memory content and IPDs suggests a CTCC. The reason is that the memory must contain a portion of covert message before the covert encoder encodes them in IPDs. They have proposed to decode the possible covert bit strings from IPDs and then they have matched these bits against the memory content. They have shown that their detection method can detect IPD based CTCC even in the presence of noise.

The main notable attributes of different CTCC detection and prevention works can be represented by the Table III.

2.2. Remote timing side channel

Remote timing side channel (RTSC) is a type of timing side channel (TSC) attack launched in a network. Here the adversary tries to exploit a vulnerability of a cryptographic implementation remotely.

Table III. Main notable attributes of CTCC detection and prevention works.

Main attributes	**CTCC detection and prevention works**
Comparison between detection tests	[Archibald 2013; Archibald and Ghosal 2014]
Use of wavelet transform and SVM	[Mou et al. 2012]
Entropy test	[Gianvecchio and Wang 2007; 2011; Mason et al. 2010]
Regularity test	[Cabuk 2006; Cabuk et al. 2009]
Blind disruption	[Wang et al. 2009; Askarov et al. 2010]
Targeting VPN based CTCC	[Sadeghi et al. 2012; Herzberg and Shulman 2013]
Targeting TCP based CTCC	[Luo et al. 2008]
Memory correlation based detection	[Stillman 2008]

Song et al. have shown two weaknesses in SSH through which an adversary can extract passwords remotely during an SSH session [Song et al. 2001]. First, the original data size can be approximately revealed by the eight-byte boundary of transmitted packets. Second, the inter-keystroke timing information is leaked due to every keystroke information is sent to the remote machine during the interactive mode. They have shown that significant information about users' typing can be obtained by using some advanced statistical techniques. Key sequences can be predicted from the inter-keystroke timings by Hidden Markov Model and their key sequence prediction algorithm. They have also proposed an attacker system called Herbivore that can speed up exhaustive search for passwords by monitoring SSH sessions by a factor of 50. They have also proposed some countermeasures.

There are some works that have launched remote timing attacks against OpenSSL. In OpenSSL, square-and-multiply algorithm is optimized using Sliding Window Exponentiation technique. In this technique a block of bits (window) of the private decryption exponent are considered in every iteration compared to one bit in normal square-and-multiply algorithm. A multiplication table is usually pre-computed for sliding window technique. Brumley et al. have shown that the extraction of the private keys is possible from a local network web server (based on OpenSSL) [Brumley and Boneh 2003; 2005]. Authors in [Aciiçmez et al. 2005] have proposed an attack that improves 10 times the efficiency of the attack in [Brumley and Boneh 2003]. They have exploited the Montgomery multiplications' timing behavior during the initialization of table. This allows them to get one of the prime factors of RSA moduli by increasing the number of multiplications. Another remote timing attack against OpenSSL is shown in [Brumley and Tuveri 2011; Billy Bob Brumley 2011]. They have used a vulnerability in OpenSSL to recover the private key of a TLS server. The vulnerability is in the Montgomerys ladder implementation in elliptic curve cryptosystem (ECC). They have used messages, signatures, and timing of those messages to launch the attack.

A remote timing attack on AES based on cache use, is reported in [Acimez et al. 2007] to get the remote cryptosystem's secret keys. The requirement is that the victim server must be a multitasking or simultaneous multithreading system running large workload.

Crosby et al. have discussed the limits of remote timing attacks by measuring network response times and jitter. They have conducted their research both on the Internet and a local network [Crosby et al. 2009]. They have proposed filters to minimize the effects of jitter and increase the timing measurement accuracy across the network.

Gong et al. have shown a RTSC attack based on a scheduler between two users [Gong and Kiyavash 2013]. Utilizing Shannon equivocation as a privacy metric, they have proved that one user can learn the complete traffic pattern of the other user if the scheduler employs a first come first serve (FCFS) policy. Moreover, they have shown the feasibility of a remote timing attack exploiting the TSC inside a home digital subscriber line (DSL) router. Using the attack an attacker can get victim's passwords, and voice over IP (VoIP) conversations.

The main notable attributes of different RTSC works can be represented by the Table IV.

Table IV. Main notable attributes of remote timing side channel works.

Main attributes	**Remote timing side channel works**
SSH based attack	[Song et al. 2001]
OpenSSL based attack	[Brumley and Boneh 2003; 2005; Aciiçmez et al. 2005] [Brumley and Tuveri 2011; Billy Bob Brumley 2011]
Cache based attack on AES	[Acimez et al. 2007]
Limits of attack	[Crosby et al. 2009]
Scheduler based attack	[Gong and Kiyavash 2013]

2.3. Summary

In this subsection we summarize the whole Section 2 and provide some possible future work directions. Most of the CTCC theoretical works address IPD based CTCC and most works modulate IPDs to implement CTCC. Packet reordering based CTCC works are less in number. Similar statement can be made for RTSC also.

Maximum transmission rate via Non-deterministic on/off timing channel has been reported in [Yao et al. 2009]. But the reported result is obtained under ideal condition i.e. without taking the security of channel or the synchronization cost taking into account. So it is apparent that the actual maximum rate is smaller than the reported one. There is a scope of further work which will report a maximum rate that is closer to actual one after taking the actual factors into account. A method to calculate the inter-packet covert timing channel transmission rate is reported in [Zi et al. 2011] in case of active CTCC. Further work is possible in case of passive CTCC. Many factors like the passing-by traffic fluctuation have to be taken into account to provide effective transmission rate for passive CTCC. A simple error correction approach is proposed in [Zi et al. 2010] to provide a reliable passive CTCC. Further research is required to explore the possibilities of more reliable communication over passive CTCC. An encoding scheme for IPD based CTCC is proposed in [Zander et al. 2011] to increase the CTCC robustness. Future work is required to analyze the effects of different network conditions including packet loss over CTCC.

Mathematical model to represent CTCC operation is reported in [Shrestha et al. 2013; Shrestha et al. 2014] and in this paper. Further work is possible that can provide mathematical models for detecting covert channels. Packet reordering based CTCC is proposed in [Ahsan and Kundur 2002]. Future work is possible to develop formal theories for packet reordering based CTCC using various coding strategies taking into account the practical network behavior. Possible future research can be to adjust CTCC parameters dynamically to handle the network changes properly. The initialization process can be equipped with learning capabilities to know the host connection properties.

Spyware communication circuits exploiting the CSMA/CA MAC protocol properties is proposed in [Kiyavash et al. 2013] to enable covert communication. Start of covert communication or the trigger time detection can be a possible future work. A wireless network is associated with many timing aspects such as rate adaptation or sleeping mode. Detailed research is required to examine if spyware can be implemented exploiting all or some of them. A covert timing channel for 802.11 networks is reported in [Radhakrishnan et al. 2013]. Impact on the Bit Error Rate due to change in surrounding wireless user number and sending rate is a possible future work. Detailed research is required to study the CTCC performance in different sender-receiver scenarios like multiple-sender single-receiver, and multiple-sender multiple-receiver.

Model-based covert timing channel framework is presented in [Gianvecchio et al. 2008]. Detection of model-based CTCCs using different goodness-of-fit tests like the Cramer-Von Mises and Anderson-Darling tests, is a possible future research direction. These tests may more effectively measure certain types of traffic though they are less general than the Kolmogorov-Smirnov test.

All of the works in RTSC are used to launch various attacks i.e. used for malicious applications. Future work is needed to see if the technology can be used for any legitimate application also.

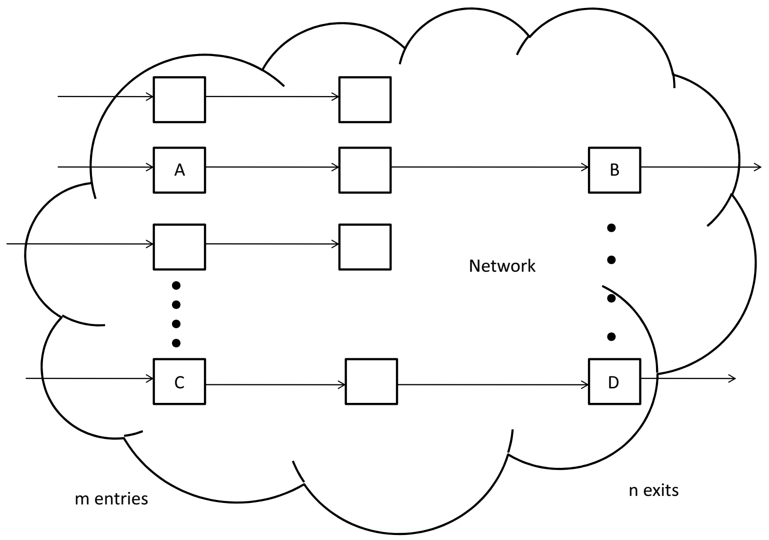


Fig. 4. Target network with m incoming and n outgoing flows. There are multiple nodes in the network including some stepping stones. A and C are the entry nodes whose traffic are relayed to the exit nodes B and D through stepping stones.

3. NETWORK FLOW WATERMARKING

In this section we provide various classes of network flow watermarking (NFW) techniques, and different threat models against those techniques.

Though NFW is a type of network timing channel, various NFW techniques, and threat models are given in this separate section to manage the sections properly. NFW is an emerging field where secret information (i.e. watermark) is transferred by modulating some timing characteristics of the flows. This aspect is very similar to covert timing channel but there are some differences. Main difference is in the application scenarios. NFW is mainly used to detect stepping stones in a target network and to deanonymize an anonymous network [Gong et al. 2013]. Stepping stones in a network refers to compromised nodes that work as midway steps to launch an ultimate attack. Figure 4 shows a target network with m incoming and n outgoing flows. In that target network, there are two incoming flows that are relayed to the exit nodes by two stepping stones. So if all the flows are watermarked, node B and D can easily correlate the incoming and outgoing flows and can detect the presence of stepping stones. In a different situation, NFW can be used to deanonymize an anonymous network like Tor [Dingledine et al. 2004]. VoIP calls using an anonymous network can also be tracked using NFW. The purpose of an anonymous network is to map incoming flows randomly to outgoing flows. NFW can relate an outgoing flow with it's corresponding incoming flow (and vice versa) and attack an anonymous network. So, depending upon the application scenario, the role of an entity changes (from protector to attacker).

There are mainly two parts in an NFW system - embedder and detector. Embedder embeds the watermark into a flow by modulating certain timing characteristics and detector detects the watermark from a flow using some statistical measures. If the application scenario is only to discover the presence of a watermark in a network flow, the output of detector can give only 1 bit of information (i.e. presence or absence). Though the watermark itself may be of multiple bits long but the output information will be only 1 bit. There are applications where multiple flows and their sources have to be detected distinctly. In those cases the output information has to be multiple bits. Recently this multi-bit output information scenario is termed separately as "flow fingerprinting" [Houmansadr and Borisov 2013b; Elices and Perez-Gonzalez 2013]. We agree with the separate terminology because it can help to identify different problems and to obtain their solutions in different cases. Flow fingerprinting can be considered as a special case of NFW. Because watermarks are used in both the cases and the fin-

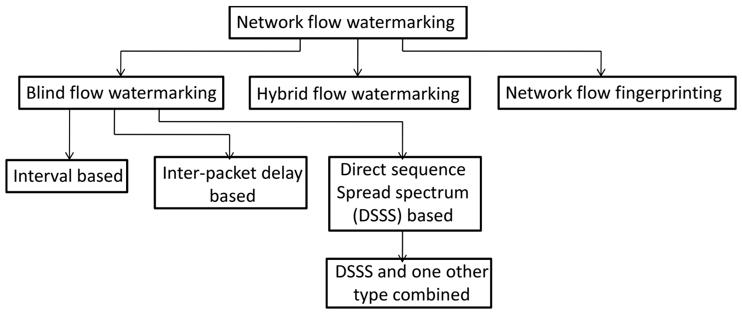

Network flow watermarking
Blind flow watermarking
Hybrid flow watermarking
Network flow fingerprinting
Interval based
Inter-packet delay based
Direct sequence Spread spectrum (DSSS) based
DSSS and one other type combined

Fig. 5. Different categories of network flow watermarking techniques.

gerprinting technique is also very similar to the watermarking technique with some differences.

As per our knowledge, the first work on NFW was published in 2003 [Wang and Reeves 2003]. After that a lot of work has been done in this field and a lot is still going on. Based upon available literature, various NFW techniques can be divided into two categories: blind and hybrid. Figure 5 shows the different categories. We have also included flow fingerprinting as a third category in the figure. There are some non-blind or passive flow analysis techniques available to correlate traffic flows. These techniques may use different flow timing characteristics like inter-packet delay (IPD) to analyze flows but they do not transfer any information (like watermark) secretly. So, we will not discuss about these techniques any further. The reason of calling these passive techniques non-blind and the second watermarking category as hybrid will be clear when we will discuss about hybrid flow watermarking. In this paper we use the terms flow watermarking and network flow watermarking (NFW) interchangeably.

3.1. Blind flow watermarking

Unlike passive or non-blind flow analysis techniques, blind flow watermarking detector does not require the original incoming flow to correlate with an outgoing flow. All the required information are embedded in the flow itself to be used by detector.

Mainly three types of blind flow watermarking techniques are reported in literature - A) interval based, B) IPD based, and C) Direct Sequence Spread Spectrum (DSSS) based.

3.1.1. Interval based flow watermarking. In this type of flow watermarking, a packet flow is divided into equal length intervals. Then some transformations are done to each selected interval to embed the watermark bits.

Wang et al. have presented an Interval Centroid Based Watermarking (ICBW) scheme [Wang et al. 2007]. Their scheme is shown to be resilient to various flow transformations like flow merging/splitting and packet dropping etc. Common flow transformation techniques are discussed in Appendix F and centroid of an interval is defined in Appendix G.

In the field of NFW, various detection mechanisms are known as attacks or threats to watermarking. So non-detectable watermarks are called attack resistant watermarks. One such attack is Multi-flow Attack (MFA) and few MFA resistant interval based watermarking schemes are available in literature. Houmansadr et al. have presented a Multi-flow Attack Resistant Interval Centroid Based Watermarking (MAR-ICBW) method [Houmansadr et al. 2009a]. MAR-ICBW removes correlations between flows by embedding the watermarks on arbitrary locations over multiple flows. Another MFA resistant scheme called Scalable Watermark Invisible and Resilient to packet Losses (SWIRL) is proposed in [Houmansadr and Borisov 2011b]. In this scheme each flow is marked with a different pattern to resist multi-flow attack. Later we will discuss more about Multi-flow Attack.

Pyun et al. have proposed a watermarking method called Interval Based Watermarking (IBW) [Pyun et al. 2007; Pyun et al. 2012]. This method utilizes the passed time of the flows to trace the traffic during repacketization, timing perturbation, chaff packets, and flow splitting. Every flow is divided in small intervals and packet timing is modified to control the packet number in particular intervals. Clock synchronization among the watermark decoder and encoder is not required.

Houmansadr et al. have proposed a variation of IBW called BotMosaic to counter Internet Relay Chat (IRC)-based botnets [Houmansadr and Borisov 2013a]. The watermark encoder places a specific pattern in the flow that will be identified by client organizations. The watermark is collaboratively inserted into multiple flows at once to enable easy detection.

A sequential watermark detection model (SWDM) is proposed in [Wang et al. 2012; Wang et al. 2013]. Three sequential detectors are also proposed to traceback network attack flows using the proposed model SWDM. The authors have proposed the "single-interval-based optimum watermark detector" (SOWD) and the "paired-intervals-based optimum watermark detector" (POWD) assuming known parameters of the perceived flow. They have also proposed the sequential sign watermark detector (SSWD) for non-parametric watermark detection.

3.1.2. IPD based flow watermarking. In this type of flow watermarking, IPDs are modified to embed watermark bits. IPD values are first quantized because they are continuous in nature. Then depending upon the specific scheme, some transformation is done to embed watermark bits.

Wang et al. have presented an IPD based watermarking method that adjusts the timing of selected packets [Wang and Reeves 2003; 2011]. They have shown quantitative trade-offs among the attainable correlation and the timing perturbation's attributes. They have also shown the packet number's upper bound to get an effective correlation. A watermarking-based host correlation detection method is proposed in [Pan et al. 2009]. Authors have divided the IPDs into two different groups randomly, and have used the average value of IPDs in different groups to embed the watermark. An IPD based watermarking technique is also used to correlate encrypted, peer-to-peer VoIP calls passing through low latency anonymizing networks [Wang et al. 2005]. In this scheme timing of selected packets are adjusted to embed a watermark into the encrypted VoIP flow. Neither robustness nor non-detectability of watermark is shown to be satisfied in their work.

Gong et al. have proposed a hidden NFW method that inserts watermarks in IPDs using quantization-index modulation [Gong et al. 2012; 2013]. They have proposed to use error-correction coding and a maximum likelihood decoding (ML) method to deal with the watermark desynchronization and substitution errors.

3.1.3. DSSS based flow watermarking. In this type of flow watermarking, a Pseudo-Noise (PN) code is used to implement the Direct Sequence Spread Spectrum (DSSS). Secret spread spectrum signal is embedded in the sender's traffic following some method based upon the specific scheme.

Yu et al. have proposed a DSSS based watermarking technique where the traffic rate is varied to embed a hidden spread spectrum signal in the sender's traffic [Yu et al. 2007]. Detailed description of the proposed system is given in Appendix H. This system does not require any offline training for detection. However, there is a limitation in their technique. The target flow must have a fixed traffic rate though it may not be true in practical network. They have not considered the requirements of non-detectability and robustness of watermark. Mere secrecy of PN codes is not sufficient to satisfy the non-detectability of watermark.

Zhang et al. have improved the scheme proposed in [Yu et al. 2007] by modulating the statistical characteristics of packets' arrival time instead of traffic rate [Zhang et al. 2009]. They have also not considered the non-detectability and robustness requirements. Another DSSS based watermarking technique by varying sender's traffic

Table V. Main notable attributes of blind flow watermarking works.

Type	Main attributes	Blind flow watermarking works
Interval based	Interval Centroid Based	[Wang et al. 2007; Houmansadr et al. 2009a] [Houmansadr and Borisov 2011b]
	Interval Based	[Pyun et al. 2007; Pyun et al. 2012]
	sequential detection	[Wang et al. 2012; Wang et al. 2013]
	Countering IRC-based botnets	[Houmansadr and Borisov 2013a]
IPD based	Generic	[Wang and Reeves 2003; 2011] [Pan et al. 2009; Wang et al. 2005]
	Quantization-index modulation	[Gong et al. 2012; 2013]
DSSS based	Varying traffic rate	[Yu et al. 2007; Huang et al. 2011]
	Modulating packet arrival time	[Zhang et al. 2009]
	Resistant to MSAC and Multi-flow Attacks	[Zhang et al. 2010b]
DSSS and other type combined	Combining with ICBW	[Wang et al. 2009; Luo et al. 2012b] [Wang et al. 2010]
	Combining with IBW	[Zhang et al. 2010a]

rate is proposed in [Huang et al. 2011]. Their scheme uses long PN code to track anonymous flows over an anonymous network. They have used a short portion of a long PN code to modulate each signal bit to provide resistance to Mean-Square Autocorrelation (MSAC) based detection.

"MSAC and Multi-flow Attacks Resistant Spread Spectrum Watermarking" (MMAR-SSW) technique is proposed in [Zhang et al. 2010b] for multiple network flow tracing. The technique embeds watermark bits in randomly selected interval positions using multiple orthogonal PN codes. Later we will discuss more about MSAC attack.

A) DSSS and other type combined. Apart from the above mentioned DSSS techniques, there are some hybrid techniques that use DSSS in combination with some other technique to embed watermarks into flows.

Interval Centroid Based Watermarking (ICBW) technique is combined with Spread Spectrum (SS) technique to obtain the "Interval Centroid Based Spread Spectrum Watermarking" scheme (ICBSSW) [Wang et al. 2009; Luo et al. 2012b]. Authors have used this technique to track multiple flows simultaneously. They have used many PN codes to randomize the inserted watermark location over many flows. A "Double Interval Centroid-Based Watermark" (DICBW) scheme is proposed in [Wang et al. 2010] for network flow traceback. DICBW considers every pair of adjacent intervals and collaboratively modulates their packet timing. They have also formed a hybrid watermarking scheme combining DICBW with SS scheme.

Zhang et al. have proposed an "Interval-Based Spread Spectrum Watermarking" (IBSSW) scheme by joining Interval-Based Watermarking (IBW) and SS schemes to trace multiple network flows [Zhang et al. 2010a]. Their scheme modulates the packet arrival time and uses multiple orthogonal PN codes to randomize the inserted watermark locations over many traffic.

The main notable attributes of different blind flow watermarking works can be represented by the Table V.

3.2. Hybrid flow watermarking

Earlier we have called the passive flow analysis technique as non-blind. Because this technique does not have any similarity with blind watermarking. There are few works available that are called as non-blind by the authors themselves. These so-called non-blind techniques actually consists of various characteristics of blind watermarking (like active flow watermarking) and passive flow analysis. That is why these techniques should be called hybrid flow watermarking. They require the original input flow characteristics to be shared with/sent to all the detectors similar to passive or non-blind case.

Peng et al. have proposed a method using packet matching and timing-based active watermarking to correlate interactive stepping stone connections [Peng et al. 2005].

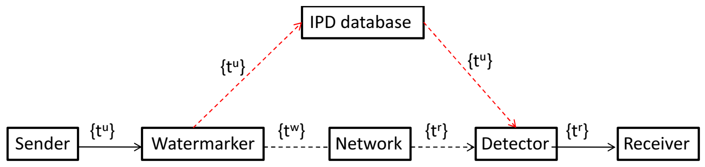


Fig. 6. RAINBOW network flow watermarking system model [Houmansadr et al. 2009b].

Table VI. Main notable attributes of hybrid flow watermarking works.

Main attributes	**Hybrid flow watermarking works**
Generic	[Peng et al. 2005]
Use of IPD database	[Houmansadr et al. 2009b; 2014; Houmansadr and Borisov 2011c]

Table VII. Main notable attributes of flow fingerprinting works.

Main attributes	**Flow fingerprinting works**
Flow fingerprinting problem	[Houmansadr and Borisov 2013b]
Game-theoretic framework	[Elices and Perez-Gonzalez 2013]

They have proposed different algorithms with various computation cost, false positive rate, and detection rate. It is clear that the detector needs all the input flow informations and that is why it is a hybrid scheme. Although it is not clear how the detectors are going to get those informations. Authors have not considered robustness and non-detectability requirements of the watermark.

A hybrid watermarking scheme called RAINBOW is proposed in [Houmansadr et al. 2009b; 2014]. RAINBOW uses smaller delays compared to blind techniques by eliminating the interference caused by the flow. Their extended scheme is shown to be robust against packet drops and repacketization using selective correlation and longer observation periods. Apart from the active watermarking like blind techniques their scheme requires an IPD database to hold all unwatermarked input flow IPD informations. This input information transmission/sharing property is similar to passive/non-blind flow analysis techniques. In RAINBOW, all detectors must have access to all the entries in the database to make correct decision. Figure 6 shows the model of the RAINBOW watermarking system. Here t^u, t^w, and t^r represent unwatermarked flow, watermarked flow, and received flow respectively. The authors have also proposed to use the Repeat-Accumulate codes to improve the detection performance of RAINBOW [Houmansadr and Borisov 2011c].

The main notable attributes of different hybrid flow watermarking works can be represented by the Table VI.

3.3. Flow fingerprinting

Houmansadr et al. have studied about flow fingerprinting for the first time [Houmansadr and Borisov 2013b]. They have introduced the flow fingerprinting problem. Differences between flow watermarking and flow fingerprinting are also explained. They have proposed a hybrid fingerprinting technique called Fancy based upon RAINBOW watermark and coding theory.

A game-theoretic framework for network flow linking problem is proposed in [Elices and Perez-Gonzalez 2013] and authors have used it to derive the Nash Equilibrium. They have compared the optimal strategies with different fingerprinting schemes to study the limits of flow correlation based on packet timings against an active attacker.

The main notable attributes of different flow fingerprinting works can be represented by the Table VII.

Table VIII shows which NFW methods satisfy non-disruptability, and/or non-detectability requirements. It can be observed that many methods do not consider the non-detectability requirement. Houmansadr et al. have not considered the non-disclosure requirement of flow fingerprinting in [Houmansadr and Borisov 2013b].

Table VIII. Non-disruptability, and/or non-detectability requirements following NFW methods.

Requirements	NFW methods
Non-disruptability	[Houmansadr et al. 2009a; 2009b; 2014] [Pyun et al. 2007; Pyun et al. 2012; Wang and Reeves 2003; 2011] [Luo et al. 2012b; Pan et al. 2009; Gong et al. 2012; 2013] [Wang et al. 2007; Wang et al. 2009; 2010] [Houmansadr and Borisov 2011b; 2011c; 2013a; 2013b]
Non-detectability	[Houmansadr et al. 2009a; Houmansadr and Borisov 2011b] [Huang et al. 2011; Zhang et al. 2010a; Zhang et al. 2010b] [Luo et al. 2012b; Wang et al. 2009; Gong et al. 2012; 2013]

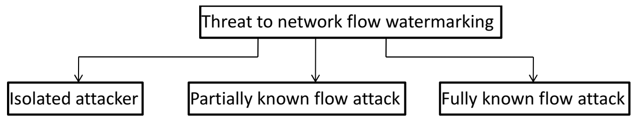

Threat to network flow watermarking
Isolated attacker
Partially known flow attack
Fully known flow attack

Fig. 7. Different threat types towards network flow watermarking techniques.

3.4. Cost analysis of different categories

If multiple detectors are used to detect relayed flows, continuous communication among the detectors is required for passive traffic analysis schemes to transmit flow statistics ($\bigcirc(n)$ communication overhead for one detector) [Houmansadr and Borisov 2011b]. Again, a detector has to correlate each outgoing flow against all the original input flow candidates ($\bigcirc(mn)$ computation overhead). The hybrid technique like RAINBOW also requires similar communication and computation overhead like passive or non-blind schemes [Houmansadr et al. 2014]. There is no communication cost other than the shared key ($\bigcirc(1)$ communication overhead) for blind schemes. Blind detectors process each outgoing flow individually and can detect watermarks ($\bigcirc(n)$ computation overhead) [Houmansadr and Borisov 2011b].

3.5. Threat models

Three types of threats are reported against flow watermarking schemes based upon the strength of the attacker - A) isolated attacker, B) partially known flow attack, and C) fully known flow attack [Gong et al. 2013]. Figure 7 shows the different threat types.

3.5.1. Isolated attacker. In this type of attack an attacker only observes a watermarked flow. The attacker may also have some knowledge of original flow like distribution of IPDs.

Kiyavash et al. have shown that interval based watermarking approach creates time dependent correlations that can be exploited using a multiple flow attack (MFA) [Kiyavash et al. 2008]. An attacker can detect the watermark, extract the secret parameters, and even delete the watermark pattern using MFA. The attack can not be defeated by using different watermarks in separate flows to carry different messages. They have assumed a Markov-modulated Poisson process (MMPP) model of interactive traffic to analyze their attack model. They have also proposed to use multiple watermark positions to defeat MFA.

A scheme to detect homogeneous PN code based watermarks in single flow is proposed in [Jia et al. 2009]. They have used traffic rate time series of a single modulated flow to get the mean-square autocorrelation (MSAC). This MSAC is then used to detect DSSS watermarks without knowing the applied PN code. Another spread-spectrum flow watermark detection mechanism using PN codes is proposed in [Luo et al. 2010]. They have also used TCPs flow-control mechanism to remove spread-spectrum flow watermarks.

3.5.2. Partially known flow attack. In this type of attack, an attacker has a distorted version of the original flow and observes a watermarked flow for detection. The imperfect version of original flow is more informative than the information available to an isolated attacker.

Table IX. Main notable attributes of different threats towards network flow watermarking techniques.

Type of threat	Main attributes	Flow watermarking threat works
Isolated attacker	Multi-flow based	[Kiyavash et al. 2008]
	Single-flow based (targeting DSSS based flow watermarking)	[Jia et al. 2009; Luo et al. 2010]
Partially known flow attack	Generic	[Peng et al. 2006; Luo et al. 2011]
Fully known flow attack	Generic	[Lin and Hopper 2012]

Peng et al. have proposed an attack method by studying the packet delays among two close stepping stones [Peng et al. 2006]. They have proposed to send packets with controlled timing. Then a detector can detect uncommon extra delays by statistical methods. This was the first work that analyzed threat to a watermark scheme. An attacker can extract secret watermark parameters, and can duplicate the watermarks using their attack. Some trade-offs of watermarking becomes apparent from their work (like trade-off between detection rate and invisibility). Another partially known flow attack called BACKLIT is proposed in [Luo et al. 2011]. It is based on the fact that any timing-based flow watermark causes noticeable alterations of a TCP flow's intrinsic timing features. They have shown successful attack against IBW, ICBW, RAINBOW and SWIRL watermarking schemes.

3.5.3. Fully known flow attack. In this type of attack, an attacker observes both the unwatermarked original flow and the watermarked flow. So the process of detection is easier than the other threat models. It must be noted that the assumptions involved in this specific attack may not be true for all practical cases and the attack may not be applicable also [Gong et al. 2013]. In [Lin and Hopper 2012], authors have introduced a known/chosen flow attack model and a copy attack. They have shown successful attacks against IBW, ICBW, RAINBOW and SWIRL.

The main notable attributes of different threats towards network flow watermarking techniques can be represented by the Table IX.

3.6. Summary

Flow watermarking techniques are quite new compared to the field of digital watermarking or multimedia watermarking. Many of the network flow watermarks are inspired by the multimedia watermarking techniques [Kiyavash et al. 2008]. Still there is a need to model various effects of network traffic on watermarks. More work can be done towards threat analysis of different watermarking schemes considering different practical limitations or situations of network. More practical application scenarios need to be examined for different watermarking techniques as well for different threats towards those techniques.

An active watermark-based correlation scheme to tackle arbitrary timing perturbations is proposed in [Wang and Reeves 2011]. Future research is possible to develop a flow watermarking technique that is optimal from coding theoretic perspective. Development of robust flow watermarking technique with few packets is another future research direction. Effect of various flow transformations (like flow merging and splitting, packet drop, packet reordering, chaff packet addition etc) on flow watermarking technique can be analyzed more elaborately in future research.

A blind detection method for malicious DSSS traceback is proposed in [Jia et al. 2009]. Future work is possible in the design of detection mechanism when various bits of same signal are spread using many PN codes. Same sequence of PN codes can be used to despread the signal. The mean square autocorrelation may not be helpful for detection if the PN codes are chosen to be orthogonal. Detailed investigation is also necessary to see if design of flow watermark elimination system is possible such that the normal traffic characteristics (like throughput) do not get affected.

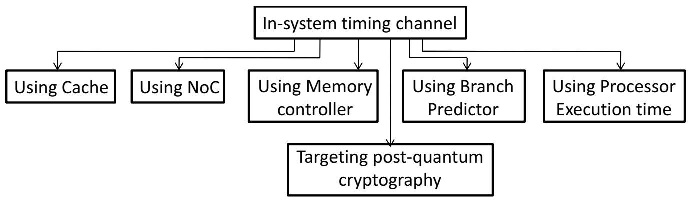


Fig. 8. In-system timing channel categories.

4. IN-SYSTEM TIMING CHANNELS

In this section we provide various classes of in-system timing channel techniques. Various types of detection and prevention techniques against different in-system timing channels are also provided.

In-system timing channel refers to those channels that reside inside a system. Here, the system refers to a computer system or a chip (like SoC, MP-SoC). In-system timing channel can either be a TSC or a covert timing channel. The difference between the two types is that covert timing channel always has a covert source like a Trojan process. But the TSC is unintentional information leakage by timing behavior. Figure 13 in Appendix A.1 shows a TSC pictorial model with different key entities. In this paper we consider only the hardware architecture based in-system timing channels. We will not discuss about programming language and OS related or Virtual machine and cloud related channels. Though insider attack is most prominent for hardware architecture related channels, both insider and external attacks are possible using in-system channels. A Trojan can be installed by an external party to obtain the required timing information.

4.1. Architecture based in-system timing channel techniques

An architecture based in-system timing channel can be implemented using cache, network-on-chip (NoC), memory controller, branch predictor, or processor execution time of different cryptographic algorithms. Even some timing channel based attacks are reported against post-quantum cryptographic algorithms like McEliece public key cryptography (PKC). Among all these types of timing channels, cache based timing channels are most common type of attacks reported in literature. Figure 8 shows different in-system timing channel categories.

4.1.1. Cache based timing channel. Cache based attacks can be classified among three types: access-driven, trace-driven, and timing-channel attack [Savaş 2013]. In this paper we discuss about cache based timing channels only. An attacker can observe the variations in execution times of cryptographic operations and can infer the approximate number of cache hits or misses. Even in some attacks, an attacker can locate the exact cache line that is accessed by the victim. There are many variations of cache timing attacks like chosen plaintext, chosen ciphertext or attack without any knowledge of plaintext or ciphertext.

Bernstein et al. has shown that full AES key retrieval is possible from a network server using the timings of known-plaintext [Bernstein 2005]. He has pointed out that the vulnerability is in the AES design itself due to difficulty in writing efficient AES code having fixed runtime. Several other timing attacks against the table-based software implementation of the AES cipher is proposed in [Bonneau and Mironov 2006]. They have predicted timing variation due to cache-collisions during encryption process. They have also proposed a patch that can reduce the vulnerability to their attacks. Xinjie et al. have also proposed a cache timing attack against 128-bit AES using first two rounds access [Xinjie et al. 2008]. They have used a malicious process to get the 128-bit full AES key by obtaining table lookup indices during encryption. Rebeiro et al. have proved that a similar third round cache based timing attack does not work [Rebeiro et al. 2010]. A differential cache-collision timing attack on AES software implementation is proposed in [Bogdanov et al. 2010]. They have treated pairs of AES executions differentially and have demonstrated the attack on ARM9. Aly et al. have reproduced Bernstein's attack towards AES on Pentium Dual-Core and Core 2 Duo processors [Aly and ElGayyar 2013]. They have targeted OpenSSL's AES implementation on Windows and Linux. They have used two-way measurements to improve Bernstein's first round attack and have also used the above minimum timing information to improve the results.

Percival has shown that memory caches with shared access, allows threads to create high bandwidth covert channel [Percival 2005]. The shared access allows a malicious thread to observe another thread's execution. This may allow stealing of cryptographic keys. Some recommendations to mitigate or eliminate this attack entirely are also provided in [Percival 2005]. Authors in [Osvik et al. 2006; Tromer et al. 2010] have described inter-process leakages revealing memory access patterns through cache memories. They have also described an attack that does not require any knowledge of specific plaintexts or ciphertexts. They have targeted Linux's encrypted partition by dm-crypt, and OpenSSL. They have also presented some countermeasures to mitigate the attacks.

First micro-architectural attack using Instruction Cache (I-Cache) is shown in [Aciiçmez 2007]. Chen et al. have proposed a trace-driven timing attack towards RSA algorithm by observing the whole I-Cache [Chen et al. 2013b]. They have also proposed an error detection mechanism to detect erroneous decisions to reduce the number of erroneous recovered bits. Vector quantization and hidden Markov models can be used to improve I-Cache data analysis techniques as proposed in [Acimez et al. 2010; Billy Bob Brumley 2011]. They have recovered keys after attacking OpenSSL's DSA implementation. They have also proposed kernel or compiler level software countermeasures. Vector quantization and hidden Markov model cryptanalysis to recover secret key is also shown in [Brumley and Hakala 2009; Billy Bob Brumley 2011]. They have targeted the timing attack against the ECC in OpenSSL.

Hund et al. have presented the limitations of kernel space Address Space Layout Randomization (ASLR) against a local attacker with restricted privileges [Hund et al. 2013]. They have shown that an adversary can implement a generic side channel attack against the memory management system to deduce information about the privileged address space layout.

4.1.2. NoC based timing channel. Network-on-chip (NoC) is a shared resource utilized by different applications running on a chip-multiprocessor (CMP) or a multiprocessor-system-on-chip (MP-SoC). One application can affect another application's timing characteristics through interference in some NoC resources like links and buffers. A malicious application can observe another application's data dependent timing characteristics and can obtain secret information resulting in TSC attack. Again two applications can communicate covertly using intentional timing characteristics modifications resulting in covert timing channel attack [Wang and Suh 2012].

4.1.3. Memory controller based timing channel. A memory-based timing channel attack is launched by exploiting the fact that the memory latencies of one program depend on memory accesses by other programs sharing the same memory [Wang et al. 2014]. In case of a shared memory controller a memory timing channel exists between software modules in different security domains. The reason is that one memory request scheduling time depends on other competing requests. The sources of interference in a memory controller can be categorized into three groups: queuing structure interference, scheduler arbitration interference, and DRAM device resource contention. Wang et al. have demonstrated that shared memory controllers are vulnerable to both side channel and covert channel attacks that exploit memory interference as timing channels [Wang et al. 2014].

4.1.4. Branch predictor based timing channel. First branch prediction based software side-channel attack is proposed in [Acimez et al. 2006]. It is shown that cryptanalysis can be conducted on a cryptographic primitive employing a data-dependent program flow. Cryptanalysis uses the extra clock cycle requirement for a mis-predicted branch. The attack is independent of memory protection mechanisms, sandboxing or virtualization. They have targeted the attack against RSA. Some countermeasures are also suggested to mitigate branch prediction attacks. Aciicmez et al. have proposed a Simple Branch Prediction Analysis (SBPA) attack [Aciiçmez et al. 2007]. Simply by observing a quasi-parallel task, the states of the CPU's Branch Predictor can be analyzed. They have attacked an OpenSSL RSA implementation by SBPA attack.

A side-channel timing attack against the MSP430 serial bootstrap loader (BSL) is proposed in [Goodspeed 2008]. Version 2.12 is the only version reported to be vulnerable. The attack is based on an unbalanced branch that takes two cycles longer to execute than other branch.

4.1.5. Execution time based timing channel. An attacker can obtain secret information used in a cryptographic algorithm by observing the time taken to execute different operations of that algorithm.

Kocher has presented an attack that can factor RSA keys and get Diffie-Hellman exponents by computing private key execution times [Kocher 1996]. He has proposed some techniques to prevent the attack for RSA and Diffie-Hellman. As per our understanding, this is the first published work that considers execution time based timing attack to break cryptographic algorithms. A basic description of the cryptanalysis method is given next. Both Diffie-Hellman and RSA private-key operations require to compute $R = y^x$ mod n, where n is public and y can be obtained by eavesdropping. The attack's main aim is to find the secret key x. For a successful attack, the victim is required to compute y^x mod n for different values of y. Here it is assumed that the attacker knows the values of y, n, and the computation time. In fact attacker can record all the messages received by the target and can calculate the response time to each y. A timing attack to recover secret keys from a smart card using Kocher's proposed ideas is reported in [Dhem et al. 2000].

Kelsey et al. have launched timing attack towards a product block cipher called IDEA, Hamming weight attack towards DES, and processor-flag attack towards RC5 [Kelsey et al. 1998]. Two DES implementations are analyzed in [Hevia and Kiwi 1999] against timing attack and authors have obtained the Hamming weight of the key used in both implementations. They have also shown that necessary design characteristics for the attack can be obtained from timing measurements. A timing attack against Rijndael is proposed in [Koeune and Quisquater 1999] using few thousand measurements per key byte.

A timing attack against RSA is proposed in [Schindler 2000]. The condition is that Chinese Remainder Theorem (CRT) and Montgomery's algorithm must be used for exponentiation with the secret exponent. Walter et al. have shown that conditional subtractions at the end of Montgomery modular multiplications can enable an attack on RSA without knowing the input plaintext [Walter and Thompson 2001]. The attack proposed by [Walter and Thompson 2001] is generalized in [Schindler 2002]. [Schindler 2002] has shown that the original attack is not applicable for 4-bit tables. He has also shown that his optimized attack can be launched against other table based methods, other multiplication algorithms and in-exact timings. Chen et al. have proposed a timing attack scheme on RSA-CRT using t-test [Chen et al. 2013a]. They have used an error detection and correction mechanism to detect and correct the erroneous decision.

4.1.6. Targeting post-quantum cryptography. Security of modern public-key cryptographic algorithms depends on some mathematical problems that cannot be solved by modern computers in acceptable times. But theses problems are assumed to be solvable by a quantum computer with sufficient qubits. So, cryptographic algorithms are required that can provide system and network security in the presence of practical quantum

Table X. Main notable attributes of different architecture based in-system timing channel works.

Type	Main attributes	Architecture based in-system timing channel works
Cache based	Targeting AES	[Bernstein 2005; Bonneau and Mironov 2006] [Xinjie et al. 2008; Rebeiro et al. 2010] [Bogdanov et al. 2010; Aly and ElGayyar 2013]
	Inter-process leakage	[Percival 2005; Osvik et al. 2006] [Tromer et al. 2010]
	Using Instruction Cache	[Aciiçmez 2007; Chen et al. 2013b] [Acimez et al. 2010; Billy Bob Brumley 2011]
	Use of vector quantization and hidden Markov model	[Acimez et al. 2010] [Brumley and Hakala 2009] [Billy Bob Brumley 2011]
	Targeting ASLR	[Hund et al. 2013]
NoC based	Inter-application interference	[Wang and Suh 2012]
Memory controller based	Memory interference	[Wang et al. 2014]
Branch predictor based	Generic	[Acimez et al. 2006; Aciiçmez et al. 2007]
	Against serial bootstrap loader	[Goodspeed 2008]
Execution time based	Generic	[Kocher 1996; Dhem et al. 2000]
	Towards a block cipher	[Kelsey et al. 1998; Hevia and Kiwi 1999] [Koeune and Quisquater 1999]
	Against RSA CRT and Montgomery's algorithm	[Schindler 2000; Walter and Thompson 2001] [Schindler 2002; Chen et al. 2013a]
Targeting post-quantum cryptography	Against public-key cryptosystems (like McEliece)	[Strenzke et al. 2008; Avanzi et al. 2011]
	Against Patterson Algorithm	[Shoufan et al. 2010; Strenzke 2010]

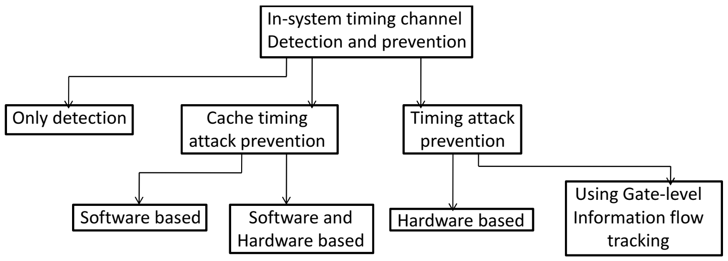

In-system timing channel Detection and prevention
Only detection
Cache timing attack prevention
Timing attack prevention
Software based
Software and Hardware based
Hardware based
Using Gate-level Information flow tracking

Fig. 9. In-system timing channel detection and prevention categories.

computers. McEliece PKC is one example of such cryptography. Strenzke et al. have proposed a timing attack towards a McEliece PKC code [Strenzke et al. 2008]. They have also suggested some countermeasures to prevent these attacks. Avanzi et al. have also presented timing side-channel attacks against the Niederreiter and McEliece PKCs and have proposed countermeasures against such attacks [Avanzi et al. 2011].

The McEliece PKC uses Patterson Algorithm during decryption operation for error correction. A timing attack towards the Patterson Algorithm is proposed in [Shoufan et al. 2010]. An attacker can extract the secret error vector by launching the attack. A similar attack is proposed in [Strenzke 2010] to gather secret permutation information. This information is useful in a brute force attack against the secret key.

The main notable attributes of different architecture based in-system timing channel works can be represented by the Table X.

4.2. Detection and Prevention

Figure 9 shows different categories of in-system timing channel detection and prevention works. There are works available that propose to prevent cache timing attack or execution time based timing attack. Some works only try to detect timing channels.

Table XI. Main notable attributes of in-system timing channel detection works.

Main attributes	In-system timing channel detection works
Generic	[Kemmerer 1982; 1983; 2002; Wray 1991; Chen and Venkataramani 2014]

4.2.1. Detection. Kemmerer has outlined a methodology for discovering storage and timing channels [Kemmerer 1982; 1983; 2002]. The methodology known as the Shared Resource Matrix Methodology can be used through all phases of the software life cycle to increase the assurance that all channels have been identified. Another method to identify all timing channels in a computer system is presented in [Wray 1991]. Chen et al. have also proposed an algorithm to identify shared processor timing channels [Chen and Venkataramani 2014]. They have tested their algorithm on memory-bus/Quick-Path-Interconnect (QPI) based, and integer divider based timing channels.

The main notable attributes of different in-system timing channel detection works can be represented by the Table XI.

4.2.2. Cache timing attack prevention. Cache timing attack can be prevented by software-only or software-hardware combined approach.

A) Software solution. Software solution provides a timing attack resistant cryptographic algorithm implementation or employ randomization technique like masking. Masking is the use of random values to mask the input to a cryptographic algorithm to uncorrelate the intermediate results against the input. This process can stop useful information leakage from the timing side-channel.

An AES randomization technique (masking) is proposed in [Blomer et al. 2005] and authors have formally proven its security against side-channel attacks.

Ksper et al. have presented a timing attack protected bit-sliced implementation of AES encryption [Ksper and Schwabe 2009]. The implementation is a counter mode AES on 64-bit Intel processor. They have also proposed a constant-time implementation of AES-GCM that is resistant to timing attack. Another timing attack protected bit-sliced implementation of AES-128 is presented in [Konighofer 2008]. The implementation is immune to cache-timing attacks as it does not need table-lookups.

B) Software-hardware combined solution. To counter cache based timing channel, different cache architectures with their software control interfaces are proposed. There is a proposal to limit the timekeeping granularity to reduce the timing channel information bandwidth. Instruction sets supporting AES operations in hardware are also proposed.

Wang et al. have proposed two security-alert caches called “Random Permutation Cache” (RPcache) and “Partition-Locked cache” (PLcache) [Wang and Lee 2006; 2007]. They have shown that the partition-based PLcache can eliminate cache interference and the randomization based RPcache can randomize cache interference to minimize information leakage. Kong et al. have analyzed both PLcache and RP-cache and have shown vulnerabilities of those designs [Kong et al. 2008]. They have proposed some modifications of those cache designs to overcome the vulnerabilities. Pre-loading is also proposed to secure the PLcache [Kong et al. 2009]. They have leveraged informing loads to protect the RPcache. They have also replaced the random permutation hardware in the RPcache with a software permutation technique.

Another cache architecture is proposed in [Wang and Lee 2008] to reduce cache miss rate, to overcome access-time overhead and to thwart information leakage. They have used a security-aware cache replacement algorithm to thwart information leakage. Liu et al. have tested the security of a security enhancing cache called Newcache using cache side channel attacks for conventional set-associative caches [Liu and Lee 2013]. They have shown that Newcache can thwart most of the attacks. They have also modified the vulnerable replacement algorithm in Newcache design to secure against the remaining possible attacks.

Martin et al. have proposed techniques to curb the adherence of performance counters and accurate timekeeping [Martin et al. 2012]. They have shown that their pro-

Table XII. Main notable attributes of different cache timing attack prevention works.

Type	Main attributes	Cache timing attack prevention works
Software based	Masking	[Blomer et al. 2005]
	Immune implementation	[Ksper and Schwabe 2009; Konighofer 2008]
Software-hardware combined	Cache design	[Wang and Lee 2006; 2007; 2008] [Kong et al. 2008; 2009; Liu and Lee 2013]
	Limiting fidelity of timekeeping	[Martin et al. 2012]
	Hardware support	[Mowery et al. 2012; Gueron 2012]

posed methods can thwart timing attacks by confusing the attacker in distinguishing different micro-architectural events.

Mowery et al. have shown that data-cache side-channel attacks on AES has become more difficult because of five new developments in software and hardware [Mowery et al. 2012]. These developments are: physically tagged caches, sophisticated software, new prefetchers, multi-core processors with caches, and the AES-NI instructions. The Intel AES New Instructions (NI) are presented in [Gueron 2012]. The proposed Intel architecture consists of 6 instructions with hardware support for AES. The table-less and data-independent runtime AES implementation can prevent timing attacks.

The main notable attributes of different cache timing attack prevention works can be represented by the Table XII.

4.2.3. Timing attack prevention. There are mainly two different solutions to prevent timing attacks : hardware based solution and using gate level information flow tracking (GLIFT). Both of these solutions are described next separately.

A) Hardware based solution. Different hardware implementations of cryptographic algorithms are proposed that can prevent execution time based timing attack. There is a proposal to make all clocks available to a process noisy (fuzzy time) and thus limiting timing channel bandwidth [Hu 1991].

Ciet et al. have proposed a parallel architecture to counteract the timing attack [Ciet et al. 2003]. They have used Montgomery Multiplication derived from Residue Number Systems (RNS) to design the architecture. It can perform an RSA signature in parallel on a set of identical and independent co-processors. A timing attack resistant elliptic curve processor is proposed in [Hodjat et al. 2005]. They have modified the point multiplication algorithm (double-add-subtract) to prevent timing attack. The processor is based on the Galois Field of GF(2n) where n is configurable. Ghosh et al. have proposed another programmable GF(p) arithmetic unit to perform modular addition, subtraction, multiplication, inversion, and division [Ghosh et al. 2011]. They have also designed an elliptic curve scalar multiplication hardware to perform point doubling and point addition in each iteration concurrently on two cores of programmable GF(p) arithmetic unit.

Ghosh et al. have proposed a 128-bit CCA2-secure McEliece cryptoprocessor incorporating BLAKE-512 module into the architecture [Ghosh and Verbauwhede 2014]. They have shown that their design is resistant against existing timing attacks. They have also introduced a binary-XGCD algorithm for Goppa field.

“Dual-spacer dual-rail delay-insensitive Logic” (D^3L) is proposed in [Cilio et al. 2010; Cilio et al. 2013] to mitigate timing attacks. They have inserted random delays to mitigate the timing-data correlation.

Different hardware based solutions are available in literature to prevent timing channel attack in Network-on-chip (NoC) based MP-SoC. Wang et al. have proposed a technique called “Reversed Priority with Static Limits” (RPSL) that requires changes to the router hardware [Wang and Suh 2012]. In RPSL scheme high priority is given to the low-security traffic. So low-security application cannot imply any information about high-security application using congestion in NoC and hence timing channel is mitigated. An on-chip network called SurfNoC is proposed in [Wassel et al. 2013] to reduce the latency due to temporal partitioning. They have introduced a scheduling

Table XIII. Main notable attributes of different timing attack prevention works.

Type	Main attributes	Timing attack prevention works
Hardware based	Fuzzy time	[Hu 1991]
	Modified architecture	[Ciet et al. 2003; Hodjat et al. 2005] [Ghosh et al. 2011; Ghosh and Verbauwhede 2014]
	Random delay	[Cilio et al. 2010; Cilio et al. 2013]
	NoC architecture	[Wang and Suh 2012; Wassel et al. 2013]
	Memory controller	[Wang et al. 2014]
Using GLIFT	Implementation	[Tiwari et al. 2009; Tiwari et al. 2010] [Tiwari et al. 2011; Oberg et al. 2011]
	Theoretical analysis	[Oberg et al. 2010; Hu et al. 2012]

policy and router micro-architecture to allow data from different domains to flow in a strictly non-interfering manner.

Wang et al. have proposed a memory controller that allows mutually mis-trusting parties to share main memory securely by eliminating memory timing channels [Wang et al. 2014]. They have proposed two changes to eliminate the interference across security domains. The changes are security domain specific queuing structure, and time slots statically allocated in the scheduling algorithm.

B) Using gate level information flow tracking (GLIFT). GLIFT is used to track all implicit, explicit and timing information movements within a system. The proposed hardware solutions are costly (in terms of area, time etc.) compared to the original implementations.

Tiwari et al. have proposed a GLIFT based architecture implementing Shadow Logic capable of tracking all explicit and implicit information flows within a machine [Tiwari et al. 2009; Tiwari et al. 2010]. They have called the logic as Shadow Logic because it co-exists with normal logic and helps to understand the information flow propagation throughout a system. Their proposed implementation is not general-purpose programmable but supports some programming to handle public-key encryption and authentication. Another configurable architectural skeleton is proposed in [Tiwari et al. 2011] that combines a microkernel with a hardware realization. They have statically verified whole structure's information flow attributes at the gate-level implementation. Oberg et al. have proposed a methodology for testing information flows in I2C and USB using GLIFT [Oberg et al. 2011]. They have shown that unintended information flows are present in those two protocols. They have also proposed a method to isolate devices on the bus using time division multiple access (TDMA).

Theoretical analysis of GLIFT is presented in [Oberg et al. 2010] and authors have explained the Shadow Logic theory for GLIFT implementing systems. They have shown the exponential increase of minterms with the increase of inputs for the Shadow Logic. Hu et al. have given a formal basis for deriving GLIFT logic [Hu et al. 2012]. They have shown that generation of precise GLIFT logic is NP-complete.

The main notable attributes of different timing attack prevention works can be represented by the Table XIII.

4.3. Summary

Most works consider only malicious applications for in-system timing channels. More elaborate studies are required on possible legitimate application aspects of in-system channels. Most of the timing attack related works target cache but there are other resources in a system. Detailed studies about possible timing attack vulnerabilities related to other resources are required. It is true that specific modification in an algorithm is required if a particular algorithm is found vulnerable. But general purpose preventive measures that is application- and architecture-independent can be developed to protect from some if not all timing attacks. The mitigation techniques can be implemented solely on software, solely on hardware, or on a mixture of both software and hardware.

Various attack scenarios targeting RSA is reported in [Acimez et al. 2006] exploiting CPU's branch prediction unit. Exploration of efficient mitigation techniques against such attack scenarios is a viable future research direction. Further investigation is required to see if such attacks are possible against symmetric ciphers like DES. The branch prediction based attacks depend on Simultaneous Multi Threading (SMT) ability of a CPU but it will be of great importance to see if the attacks can be effective against non-SMT capable CPUs. Two cache architectures i.e. the RPcache and the PLcache are proposed in [Wang and Lee 2007] to prevent cache based TSC attacks by eliminating or randomizing cache interference. Future research is required to explore the design space of security-aware cache and computer architectures without sacrificing performance, cost, and energy consumption.

A formal notion of security for randomized maskings is presented in [Blomer et al. 2005] for cryptographic algorithms. Future research can be conducted to find solutions where the requirement of randomness is minimized without any adverse effect on security. The reason is that true randomness generation is very costly. A timing-attack resistant crypto-processor for computing McEliece post-quantum PKS is proposed in [Ghosh and Verbauwhede 2014]. Future research can be directed to develop efficient crypto-processors for different code-based crypto-systems like Niederreiter PKS.

5. CONCLUSIONS

In this paper we have provided a detailed survey of timing channels. Both network and in-system timing channels have been considered. We have given models for both covert timing channel and TSC. Both network and in-system timing channels are categorized and each category is discussed in detail. We have also categorized and provided detailed discussions of different detection and prevention techniques of different timing channel types.

We have also indicated the various possibilities of future research directions. For example, innovative legitimate and malicious applications of timing channels are a possible future research direction. Till now most of the timing channel categories are considered as a threat to information security but it must be noted that no technology is bad or good by itself. The applications of that technology make it good or bad. This is more relevant in current scenarios of high speed network and massively parallel multiprocessor chips. Malicious application scenarios are also needed to be examined so that effective detection and prevention techniques can be devised for them.

REFERENCES

Onur Aciiçmez. 2007. Yet Another MicroArchitectural Attack:: Exploiting I-Cache. In *Proceedings of the 2007 ACM Workshop on Computer Security Architecture (CSAW '07)*. ACM, New York, NY, USA, 11–18. DOI:http://dx.doi.org/10.1145/1314466.1314469

Onur Aciiçmez, Çetin Kaya Koç, and Jean-Pierre Seifert. 2007. On the Power of Simple Branch Prediction Analysis. In *Proceedings of the 2Nd ACM Symposium on Information, Computer and Communications Security (ASIACCS '07)*. ACM, New York, NY, USA, 312–320. DOI:http://dx.doi.org/10.1145/1229285.1266999

Onur Aciiçmez, Werner Schindler, and Çetin K. Koç. 2005. Improving Brumley and Boneh Timing Attack on Unprotected SSL Implementations. In *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS '05)*. ACM, New York, NY, USA, 139–146. DOI:http://dx.doi.org/10.1145/1102120.1102140

Onur Acimez, BillyBob Brumley, and Philipp Grabher. 2010. New Results on Instruction Cache Attacks. In *Cryptographic Hardware and Embedded Systems, CHES 2010*, Stefan Mangard and Franois-Xavier Standaert (Eds.). Lecture Notes in Computer Science, Vol. 6225. Springer Berlin Heidelberg, 110–124. DOI:http://dx.doi.org/10.1007/978-3-642-15031-9_8

Onur Acimez, etinKaya Ko, and Jean-Pierre Seifert. 2006. Predicting Secret Keys Via Branch Prediction. In *Topics in Cryptology CT-RSA 2007*, Masayuki Abe (Ed.). Lecture Notes in Computer Science, Vol. 4377. Springer Berlin Heidelberg, 225–242. DOI:http://dx.doi.org/10.1007/11967668_15

Onur Acimez, Werner Schindler, and etinK. Ko. 2007. Cache Based Remote Timing Attack on the AES. In *Topics in Cryptology CT-RSA 2007*, Masayuki Abe (Ed.). Lecture Notes in Computer Science, Vol. 4377. Springer Berlin Heidelberg, 271–286. DOI:http://dx.doi.org/10.1007/11967668_18

Guarav Shah Adam Aviv and Matt Blaze. 2011. *Steganographic Timing Channels*. Technical Report. Department of Computer and Information Science, University of Pennsylvania.

S.A. Ahmadzadeh and G. Agnew. 2013. Turbo covert channel: An iterative framework for covert communication over data networks. In *INFOCOM, 2013 Proceedings IEEE*. 2031–2039. DOI: http://dx.doi.org/10.1109/INFCOM.2013.6567004

K. Ahsan and D. Kundur. 2002. Practical Data Hiding in TCP/IP. In *Proc. Workshop on Multimedia Security at ACM Multimedia '02* (2002-12-01). Juan Les Pins, France. http://www.comm.utoronto.ca/~dkundur/pub_pdfs/AhsKunMMSec02.pdf

Hassan Aly and Mohammed ElGayyar. 2013. Attacking AES Using Bernsteins Attack on Modern Processors. In *Progress in Cryptology AFRICACRYPT 2013*, Amr Youssef, Abderrahmane Nitaj, and AboulElla Hassanien (Eds.). Lecture Notes in Computer Science, Vol. 7918. Springer Berlin Heidelberg, 127–139. DOI: http://dx.doi.org/10.1007/978-3-642-38553-7_7

V. Anantharam and S. Verdu. 1996. Bits through queues. *Information Theory, IEEE Transactions on* 42, 1 (jan 1996), 4 –18. DOI: http://dx.doi.org/10.1109/18.481773

R. Archibald. 2013. *Design and Detection of Covert Communiation: Timing Channels and Application Tunneling*. Ph.D. Dissertation. Davis, CA., USA.

R. Archibald and D. Ghosal. 2012. A Covert Timing Channel Based on Fountain Codes. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*. 970–977. DOI: http://dx.doi.org/10.1109/TrustCom.2012.21

Rennie Archibald and Dipak Ghosal. 2014. A comparative analysis of detection metrics for covert timing channels. *Computers & Security* 45, 0 (2014), 284 – 292. DOI: http://dx.doi.org/10.1016/j.cose.2014.03.007

Aslan Askarov, Danfeng Zhang, and Andrew C. Myers. 2010. Predictive Black-box Mitigation of Timing Channels. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS '10)*. ACM, New York, NY, USA, 297–307. DOI: http://dx.doi.org/10.1145/1866307.1866341

Roberto Avanzi, Simon Hoerder, Dan Page, and Michael Tunstall. 2011. Side-channel attacks on the McEliece and Niederreiter public-key cryptosystems. *Journal of Cryptographic Engineering* 1, 4 (2011), 271–281. DOI: http://dx.doi.org/10.1007/s13389-011-0024-9

A.S. Bedekar and M. Azizoglu. 1998. The information-theoretic capacity of discrete-time queues. *Information Theory, IEEE Transactions on* 44, 2 (Mar 1998), 446–461. DOI: http://dx.doi.org/10.1109/18.661496

Do E. Bell and Lo J. La Padula. 1976. *Secure Computer System: Unified Exposition and MULTICS Interpretation*. Technical Report 522B. Deputy for Command and Management System, Hanscom Air Force Base, Bedford, Massachusetts.

Daniel J. Bernstein. 2005. *Cache-timing attacks on AES*. Technical Report. Department of Mathematics, Statistics, and Computer Science, The University of Illinois at Chicago, Chicago, IL 606077045.

Billy Bob Brumley. 2011. *Covert Timing Channels, Caching, and Cryptography*. Ph.D. Dissertation. Espoo, Finland.

Johannes Blomer, Jorge Guajardo, and Volker Krummel. 2005. Provably Secure Masking of AES. In *Selected Areas in Cryptography*, Helena Handschuh and M.Anwar Hasan (Eds.). Lecture Notes in Computer Science, Vol. 3357. Springer Berlin Heidelberg, 69–83. DOI: http://dx.doi.org/10.1007/978-3-540-30564-4_5

Andrey Bogdanov, Thomas Eisenbarth, Christof Paar, and Malte Wienecke. 2010. Differential Cache-Collision Timing Attacks on AES with Applications to Embedded CPUs. In *Topics in Cryptology - CT-RSA 2010*, Josef Pieprzyk (Ed.). Lecture Notes in Computer Science, Vol. 5985. Springer Berlin Heidelberg, 235–251. DOI: http://dx.doi.org/10.1007/978-3-642-11925-5_17

Joseph Bonneau and Ilya Mironov. 2006. Cache-Collision Timing Attacks Against AES. In *Cryptographic Hardware and Embedded Systems - CHES 2006*, Louis Goubin and Mitsuru Matsui (Eds.). Lecture Notes in Computer Science, Vol. 4249. Springer Berlin Heidelberg, 201–215. DOI: http://dx.doi.org/10.1007/11894063_16

BillyBob Brumley and RistoM. Hakala. 2009. Cache-Timing Template Attacks. In *Advances in Cryptology ASIACRYPT 2009*, Mitsuru Matsui (Ed.). Lecture Notes in Computer Science, Vol. 5912. Springer Berlin Heidelberg, 667–684. DOI: http://dx.doi.org/10.1007/978-3-642-10366-7_39

BillyBob Brumley and Nicola Tuveri. 2011. Remote Timing Attacks Are Still Practical. In *Computer Security ESORICS 2011*, Vijay Atluri and Claudia Diaz (Eds.). Lecture Notes in Computer Science, Vol. 6879. Springer Berlin Heidelberg, 355–371. DOI: http://dx.doi.org/10.1007/978-3-642-23822-2_20

David Brumley and Dan Boneh. 2003. Remote Timing Attacks Are Practical. In *Proceedings of the 12th Conference on USENIX Security Symposium - Volume 12 (SSYM'03)*. USENIX Association, Berkeley, CA, USA, 1–1. http://dl.acm.org/citation.cfm?id=1251353.1251354

David Brumley and Dan Boneh. 2005. Remote timing attacks are practical. *Computer Networks* 48, 5 (2005), 701 – 716. DOI: http://dx.doi.org/10.1016/j.comnet.2005.01.010 Web Security.

Serdar Cabuk. 2006. *Network Covert Channels: Design, Analysis, Detection, and Elimination*. Ph.D. Dissertation. West Lafayette, IN, USA.

Serdar Cabuk, Carla E. Brodley, and Clay Shields. 2009. IP Covert Channel Detection. *ACM Trans. Inf. Syst. Secur.* 12, 4, Article 22 (April 2009), 29 pages. DOI: http://dx.doi.org/10.1145/1513601.1513604

Christian Cachin. 2004. An information-theoretic model for steganography. *Information and Computation* 192, 1 (2004), 41 – 56. DOI:http://dx.doi.org/10.1016/j.ic.2004.02.003

R.C. Chakinala, A. Kumarasubramanian, R. Manokaran, G. Noubir, C.Pandu Rangan, and R. Sundaram. 2007. Steganographic Communication in Ordered Channels. In *Information Hiding*, JanL. Camenisch, ChristianS. Collberg, NeilF. Johnson, and Phil Sallee (Eds.). Lecture Notes in Computer Science, Vol. 4437. Springer Berlin Heidelberg, 42–57. DOI:http://dx.doi.org/10.1007/978-3-540-74124-4_4

Changlong Chen, Min Song, G. Hsieh, and Chunsheng Xin. 2011. A PLL Based Approach to Building an Effective Covert Timing Channel. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*. 1–5. DOI:http://dx.doi.org/10.1109/GLOCOM.2011.6134373

CaiSen Chen, Tao Wang, YingZhan Kou, XiaoCen Chen, and Xiong Li. 2013b. Improvement of trace-driven I-Cache timing attack on the {RSA} algorithm. *Journal of Systems and Software* 86, 1 (2013), 100 – 107. DOI:http://dx.doi.org/10.1016/j.jss.2012.07.020

CaiSen Chen, Tao Wang, and Junjian Tian. 2013a. Improving timing attack on RSA-CRT via error detection and correction strategy. *Information Sciences* 232, 0 (2013), 464 – 474. DOI:http://dx.doi.org/10.1016/j.ins.2012.01.027

Jie Chen and Guru Venkataramani. 2014. An Algorithm for Detecting Contention-based Covert Timing Channels on Shared Hardware. In *Proceedings of the Third Workshop on Hardware and Architectural Support for Security and Privacy (HASP '14)*. ACM, New York, NY, USA, Article 1, 8 pages. DOI:http://dx.doi.org/10.1145/2611765.2611766

M. Ciet, M. Neve, E. Peeters, and J-J Quisquater. 2003. Parallel FPGA implementation of RSA with residue number systems - can side-channel threats be avoided?. In *Circuits and Systems, 2003 IEEE 46th Midwest Symposium on*, Vol. 2. 806–810 Vol. 2. DOI:http://dx.doi.org/10.1109/MWSCAS.2003.1562409

W. Cilio, M. Linder, C. Porter, Jia Di, S. Smith, and D. Thompson. 2010. Side-channel attack mitigation using dual-spacer Dual-rail Delay-insensitive Logic (D3L). In *IEEE SoutheastCon 2010 (SoutheastCon), Proceedings of the*. 471–474. DOI:http://dx.doi.org/10.1109/SECON.2010.5453826

Washington Cilio, Michael Linder, Chris Porter, Jia Di, Dale R. Thompson, and Scott C. Smith. 2013. Mitigating power- and timing-based side-channel attacks using dual-spacer dual-rail delay-insensitive asynchronous logic. *Microelectronics Journal* 44, 3 (2013), 258 – 269. DOI:http://dx.doi.org/10.1016/j.mejo.2012.12.001

T.P. Coleman and N. Kiyavash. 2008a. Practical codes for queueing channels: An algebraic, state-space, message-passing approach. In *Information Theory Workshop, 2008. ITW '08. IEEE*. 318–322. DOI:http://dx.doi.org/10.1109/ITW.2008.4578677

T.P. Coleman and N. Kiyavash. 2008b. Sparse graph codes and practical decoding algorithms for communicating over packet timings in networks. In *Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on*. 447–452. DOI:http://dx.doi.org/10.1109/CISS.2008.4558568

Scott A. Crosby, Dan S. Wallach, and Rudolf H. Riedi. 2009. Opportunities and Limits of Remote Timing Attacks. *ACM Trans. Inf. Syst. Secur.* 12, 3, Article 17 (Jan. 2009), 29 pages. DOI:http://dx.doi.org/10.1145/1455526.1455530

Department of Defense Standard. 1985. Trusted Computer System Evaluation Criteria. *Tech. Rep. DOD 5200.28-STD* (1985).

Jean-Franois Dhem, Franois Koeune, Philippe-Alexandre Leroux, Patrick Mestr, Jean-Jacques Quisquater, and Jean-Louis Willems. 2000. A Practical Implementation of the Timing Attack. In *Smart Card Research and Applications*, Jean-Jacques Quisquater and Bruce Schneier (Eds.). Lecture Notes in Computer Science, Vol. 1820. Springer Berlin Heidelberg, 167–182. DOI:http://dx.doi.org/10.1007/10721064_15

Roger Dingledine, Nick Mathewson, and Paul Syverson. 2004. Tor: The Second-generation Onion Router. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13 (SSYM'04)*. USENIX Association, Berkeley, CA, USA, 21–21. http://dl.acm.org/citation.cfm?id=1251375.1251396

B.P. Dunn, M. Bloch, and J.N. Laneman. 2009. Secure bits through queues. In *Networking and Information Theory, 2009. ITW 2009. IEEE Information Theory Workshop on*. 37–41. DOI:http://dx.doi.org/10.1109/ITWNIT.2009.5158537

J.J. Edwards, J.D. Brown, and P.C. Mason. 2012. Using covert timing channels for attack detection in MANETs. In *MILITARY COMMUNICATIONS CONFERENCE, 2012 - MILCOM 2012*. 1–7. DOI:http://dx.doi.org/10.1109/MILCOM.2012.6415726

A El-Atawy and E. Al-Shaer. 2009. Building Covert Channels over the Packet Reordering Phenomenon. In *INFOCOM 2009, IEEE*. 2186–2194. DOI:http://dx.doi.org/10.1109/INFCOM.2009.5062143

J.A. Elices and F. Perez-Gonzalez. 2013. The flow fingerprinting game. In *Information Forensics and Security (WIFS), 2013 IEEE International Workshop on*. 97–102. DOI:http://dx.doi.org/10.1109/WIFS.2013.6707801

I. Ezzeddine and P. Moulin. 2009. Achievable rates for queue-based timing stegocodes. In *Information Theory Workshop, 2009. ITW 2009. IEEE*. 379–383. DOI:http://dx.doi.org/10.1109/ITW.2009.5351197

S. Ghosh, D. Mukhopadhyay, and D. Roychowdhury. 2011. Petrel: Power and Timing Attack Resistant Elliptic Curve Scalar Multiplier Based on Programmable GF(p) Arithmetic Unit. *Cir-*

cuits and Systems I: Regular Papers, IEEE Transactions on 58, 8 (Aug 2011), 1798–1812. DOI:http://dx.doi.org/10.1109/TCSI.2010.2103190

S. Ghosh and I Verbauwhede. 2014. BLAKE-512-Based 128-Bit CCA2 Secure Timing Attack Resistant McEliece Cryptoprocessor. *Computers, IEEE Transactions on* 63, 5 (May 2014), 1124–1133. DOI:http://dx.doi.org/10.1109/TC.2012.271

S. Gianvecchio and H. Wang. 2007. Detecting covert timing channels: an entropy-based approach. In *CCS '07: Proceedings of the 14th ACM Conference on Computer and Communications Security*. 307–316.

S. Gianvecchio and H. Wang. 2011. An Entropy-Based Approach to Detecting Covert Timing Channels. *Dependable and Secure Computing, IEEE Transactions on* 8, 6 (nov.-dec. 2011), 785–797. DOI:http://dx.doi.org/10.1109/TDSC.2010.46

S. Gianvecchio, H. Wang, D. Wijesekera, and S. Jajodia. 2008. Model-Based Covert Timing Channels: Automated Modeling and Evasion. In *RAID '08: Proceedings of the 11th international symposium on Recent Advances in Intrusion Detection*. Springer-Verlag, Berlin, Heidelberg, 211–230.

J. Giles and B. Hajek. 2002. An information-theoretic and game-theoretic study of timing channels. *Information Theory, IEEE Transactions on* 48, 9 (sep 2002), 2455 – 2477. DOI:http://dx.doi.org/10.1109/TIT.2002.801405

C. G. Girling. 1987. Covert Channels in LAN's. *IEEE Transactions on Software Engineering* 13, 2 (1987), 292–296.

S.Z. Goher, B. Javed, and N.A. Saqib. 2012. Covert channel detection: A survey based analysis. In *High Capacity Optical Networks and Enabling Technologies (HONET), 2012 9th International Conference on*. 057–065. DOI:http://dx.doi.org/10.1109/HONET.2012.6421435

Xun Gong and N. Kiyavash. 2013. Timing side channels for traffic analysis. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. 8697–8701. DOI:http://dx.doi.org/10.1109/ICASSP.2013.6639364

Xun Gong, M. Rodrigues, and N. Kiyavash. 2012. Invisible flow watermarks for channels with dependent substitution and deletion errors. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. 1773–1776. DOI:http://dx.doi.org/10.1109/ICASSP.2012.6288243

Xun Gong, M. Rodrigues, and N. Kiyavash. 2013. Invisible Flow Watermarks for Channels With Dependent Substitution, Deletion, and Bursty Insertion Errors. *Information Forensics and Security, IEEE Transactions on* 8, 11 (Nov 2013), 1850–1859. DOI:http://dx.doi.org/10.1109/TIFS.2013.2279794

Travis Goodspeed. 2008. A Side-channel Timing Attack of the MSP430 BSL. In *Black Hat USA 2008*.

S.K. Gorantla, S. Kadloor, T.P. Coleman, N. Kiyavash, I.S. Moskowitz, and M.H. Kang. 2010. Directed information and the NRL Network Pump. In *Information Theory and its Applications (ISITA), 2010 International Symposium on*. 343–348. DOI:http://dx.doi.org/10.1109/ISITA.2010.5649143

S.K. Gorantla, S. Kadloor, N. Kiyavash, T.P. Coleman, I.S. Moskowitz, and M.H. Kang. 2012. Characterizing the Efficacy of the NRL Network Pump in Mitigating Covert Timing Channels. *Information Forensics and Security, IEEE Transactions on* 7, 1 (FEBRUARY 2012), 64–75. DOI:http://dx.doi.org/10.1109/TIFS.2011.2163398

Shay Gueron. 2012. *Intel Advanced Encryption Standard (AES) New Instructions Set*. Technical Report 323641-001, Revision 3.01. Intel Corporation, Intel Architecture Group, Israel Development Center.

A. Herzberg and H. Shulman. 2013. Limiting MitM to MitE Covert-Channels. In *Availability, Reliability and Security (ARES), 2013 Eighth International Conference on*. 236–241. DOI:http://dx.doi.org/10.1109/ARES.2013.138

Alejandro Hevia and Marcos Kiwi. 1999. Strength of Two Data Encryption Standard Implementations Under Timing Attacks. *ACM Trans. Inf. Syst. Secur.* 2, 4 (Nov. 1999), 416–437. DOI:http://dx.doi.org/10.1145/330382.330390

A Hodjat, D.D. Hwang, and I Verbauwhede. 2005. A scalable and high performance elliptic curve processor with resistance to timing attacks. In *Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on*, Vol. 1. 538–543 Vol. 1. DOI:http://dx.doi.org/10.1109/ITCC.2005.32

Amir Houmansadr and Nikita Borisov. 2011a. CoCo: Coding-Based Covert Timing Channels for Network Flows. In *Information Hiding (Lecture Notes in Computer Science)*, Toms Filler, Toms Pevn, Scott Craver, and Andrew D. Ker (Eds.), Vol. 6958. Springer, 314–328. http://dblp.uni-trier.de/db/conf/ih/ih2011.html#HoumansadrB11

Amir Houmansadr and Nikita Borisov. 2011b. SWIRL: A scalable watermark to detect correlated network flows. In *In Network and Distributed System Security Symposium. Internet Society*.

A. Houmansadr and N. Borisov. 2011c. Towards improving network flow watermarks using the repeat-accumulate codes. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. 1852–1855. DOI:http://dx.doi.org/10.1109/ICASSP.2011.5946866

Amir Houmansadr and Nikita Borisov. 2013a. BotMosaic: Collaborative network watermark for the detection of IRC-based botnets. *Journal of Systems and Software* 86, 3 (2013), 707 – 715. DOI:http://dx.doi.org/10.1016/j.jss.2012.11.005

Amir Houmansadr and Nikita Borisov. 2013b. The Need for Flow Fingerprints to Link Correlated Network Flows. In *Privacy Enhancing Technologies*, Emiliano De Cristofaro and Matthew Wright (Eds.). Lecture Notes in Computer Science, Vol. 7981. Springer Berlin Heidelberg, 205–224. DOI:http://dx.doi.org/10.1007/978-3-642-39077-7_11

A. Houmansadr, N. Kiyavash, and N. Borisov. 2009a. Multi-flow attack resistant watermarks for network flows. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. 1497–1500. DOI:http://dx.doi.org/10.1109/ICASSP.2009.4959879

Amir Houmansadr, Negar Kiyavash, and Nikita Borisov. 2009b. RAINBOW: A robust and invisible non-blind watermark for network flows. In *In: Proceedings of the Network and Distributed System Security Symposium (NDSS 2009). The INTERNET Society*.

A Houmansadr, N. Kiyavash, and N. Borisov. 2014. Non-Blind Watermarking of Network Flows. *Networking, IEEE/ACM Transactions on* 22, 4 (Aug 2014), 1232–1244. DOI:http://dx.doi.org/10.1109/TNET.2013.2272740

W.M. Hu. 1991. Reducing timing channels with fuzzy time. In *Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy*. 8–20.

Wei Hu, J. Oberg, A Irturk, M. Tiwari, T. Sherwood, Dejun Mu, and R. Kastner. 2012. On the Complexity of Generating Gate Level Information Flow Tracking Logic. *Information Forensics and Security, IEEE Transactions on* 7, 3 (June 2012), 1067–1080. DOI:http://dx.doi.org/10.1109/TIFS.2012.2189105

Junwei Huang, Xian Pan, Xinwen Fu, and Jie Wang. 2011. Long PN code based DSSS watermarking. In *INFOCOM, 2011 Proceedings IEEE*. 2426–2434. DOI:http://dx.doi.org/10.1109/INFCOM.2011.5935064

R. Hund, C. Willems, and T. Holz. 2013. Practical Timing Side Channel Attacks against Kernel Space ASLR. In *Security and Privacy (SP), 2013 IEEE Symposium on*. 191–205. DOI:http://dx.doi.org/10.1109/SP.2013.23

Jason Jaskolka, Ridha Khedri, and Qinglei Zhang. 2012. On the Necessary Conditions for Covert Channel Existence: A State-of-the-Art Survey. *Procedia Computer Science* 10, 0 (2012), 458 – 465. DOI:http://dx.doi.org/10.1016/j.procs.2012.06.059 {ANT} 2012 and MobiWIS 2012.

Weijia Jia, Fung Po Tso, Zhen Ling, Xinwen Fu, Dong Xuan, and Wei Yu. 2009. Blind Detection of Spread Spectrum Flow Watermarks. In *INFOCOM 2009, IEEE*. 2195–2203. DOI:http://dx.doi.org/10.1109/INFCOM.2009.5062144

John Kelsey, Bruce Schneier, David Wagner, and Chris Hall. 1998. Side channel cryptanalysis of product ciphers. In *Computer Security ESORICS 98*, Jean-Jacques Quisquater, Yves Deswarte, Catherine Meadows, and Dieter Gollmann (Eds.). Lecture Notes in Computer Science, Vol. 1485. Springer Berlin Heidelberg, 97–110. DOI:http://dx.doi.org/10.1007/BFb0055858

Richard Kemmerer. 1982. A Practical Approach to Identifying Storage and Timing Channels. In *Security and Privacy, IEEE Symposium on*.

R.A. Kemmerer. 2002. A practical approach to identifying storage and timing channels: twenty years later. In *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*. 109–118. DOI:http://dx.doi.org/10.1109/CSAC.2002.1176284

Richard A. Kemmerer. 1983. Shared Resource Matrix Methodology: An Approach to Identifying Storage and Timing Channels. *ACM Trans. Comput. Syst.* 1, 3 (Aug. 1983), 256–277. DOI:http://dx.doi.org/10.1145/357369.357374

H. Khan, Y. Javed, F. Mirza, and S.A. Khayam. 2009. Embedding a Covert Channel in Active Network Connections. In *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*. 1–6. DOI:http://dx.doi.org/10.1109/GLOCOM.2009.5425348

N. Kiyavash and T. Coleman. 2009. Covert timing channels codes for communication over interactive traffic. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. Taipei, Taiwan, 1485–1488.

Negar Kiyavash, Amir Houmansadr, and Nikita Borisov. 2008. Multi-flow Attacks Against Network Flow Watermarking Schemes. In *17th USENIX Security Symposium*.

N. Kiyavash, F. Koushanfar, T.P. Coleman, and M. Rodrigues. 2013. A Timing Channel Spyware for the CSMA/CA Protocol. *Information Forensics and Security, IEEE Transactions on* 8, 3 (March 2013), 477–487. DOI:http://dx.doi.org/10.1109/TIFS.2013.2238930

PaulC. Kocher. 1996. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *Advances in Cryptology CRYPTO 96*, Neal Koblitz (Ed.). Lecture Notes in Computer Science, Vol. 1109. Springer Berlin Heidelberg, 104–113. DOI:http://dx.doi.org/10.1007/3-540-68697-5_9

Francois Koeune and Jean-Jacques Quisquater. 1999. *A timing attack against Rijndael*. Technical Report CG-1999/1. Universite catholique de Louvain, Departement d Electricite (DICE), Belgium.

Jingfei Kong, Onur Aciicmez, Jean-Pierre Seifert, and Huiyang Zhou. 2008. Deconstructing New Cache Designs for Thwarting Software Cache-based Side Channel Attacks. In *Proceedings of the 2Nd ACM Workshop on Computer Security Architectures (CSAW '08)*. ACM, New York, NY, USA, 25–34. DOI:http://dx.doi.org/10.1145/1456508.1456514

J. Kong, O. Aciicmez, J.-P. Seifert, and Huiyang Zhou. 2009. Hardware-software integrated approaches to defend against software cache-based side channel attacks. In *High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on*. 393–404. DOI:http://dx.doi.org/10.1109/HPCA.2009.4798277

Robert Konighofer. 2008. A Fast and Cache-Timing Resistant Implementation of the AES. In *Topics in Cryptology CT-RSA 2008*, Tal Malkin (Ed.). Lecture Notes in Computer Science, Vol. 4964. Springer Berlin Heidelberg, 187–202. DOI:http://dx.doi.org/10.1007/978-3-540-79263-5_12

Kush Kothari and Matthew Wright. 2013. Mimic: An active covert channel that evades regularity-based detection. *Computer Networks* 57, 3 (2013), 647 – 657. DOI:http://dx.doi.org/10.1016/j.comnet.2012.10.008

Emilia Ksper and Peter Schwabe. 2009. Faster and Timing-Attack Resistant AES-GCM. In *Cryptographic Hardware and Embedded Systems - CHES 2009*, Christophe Clavier and Kris Gaj (Eds.). Lecture Notes in Computer Science, Vol. 5747. Springer Berlin Heidelberg, 1–17. DOI:http://dx.doi.org/10.1007/978-3-642-04138-9_1

Butler W. Lampson. 1973. A Note on the Confinement Problem. *Commun. ACM* 16, 10 (Oct. 1973), 613–615. DOI:http://dx.doi.org/10.1145/362375.362389

Ki Suh Lee, Han Wang, and Hakim Weatherspoon. 2014. PHY Covert Channels: Can You See the Idles?. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation (NSDI'14)*. USENIX Association, Berkeley, CA, USA, 173–185. http://dl.acm.org/citation.cfm?id=2616448.2616465

Zi Lin and Nicholas Hopper. 2012. New Attacks on Timing-based Network Flow Watermarks. In *Proceedings of the 21st USENIX Conference on Security Symposium (Security'12)*. USENIX Association, Berkeley, CA, USA, 20–20. http://dl.acm.org/citation.cfm?id=2362793.2362813

Fangfei Liu and Ruby B. Lee. 2013. Security Testing of a Secure Cache Design. In *Proceedings of the 2Nd International Workshop on Hardware and Architectural Support for Security and Privacy (HASP '13)*. ACM, New York, NY, USA, Article 3, 8 pages. DOI:http://dx.doi.org/10.1145/2487726.2487729

Guangjie Liu, Jiangtao Zhai, and Yuewei Dai. 2012. Network covert timing channel with distribution matching. *Telecommunication Systems* 49, 2 (2012), 199–205. DOI:http://dx.doi.org/10.1007/s11235-010-9368-1

Guangjie Liu, Jiangtao Zhai, Yuewei Dai, and Zhiquan Wang. 2009. Covert Timing Channel with Distribution Matching. In *Proceedings of the 2009 International Conference on Multimedia Information Networking and Security - Volume 01 (MINES '09)*. IEEE Computer Society, Washington, DC, USA, 565–568. DOI:http://dx.doi.org/10.1109/MINES.2009.280

Y. Liu, F. Armknecht, D. Ghosal, S. Katzenbeisser, A. Sadeghi, and S. Schulz. 2009. Hide and Seek in Time - Robust Covert Timing Channels. In *14th European Symposium on Research in Computer Security (ESORICS)*.

Yali Liu, Dipak Ghosal, Frederik Armknecht, Ahmad-Reza Sadeghi, Steffen Schulz, and Stefan Katzenbeisser. 2010. Robust and Undetectable Steganographic Timing Channels for i.i.d. Traffic. In *Information Hiding*, Rainer Bhme, PhilipW.L. Fong, and Reihaneh Safavi-Naini (Eds.). Lecture Notes in Computer Science, Vol. 6387. Springer Berlin Heidelberg, 193–207. DOI:http://dx.doi.org/10.1007/978-3-642-16435-4_15

Junzhou Luo, Xiaogang Wang, and Ming Yang. 2012b. An interval centroid based spread spectrum watermarking scheme for multi-flow traceback. *Journal of Network and Computer Applications* 35, 1 (2012), 60 – 71. DOI:http://dx.doi.org/10.1016/j.jnca.2011.03.003 Collaborative Computing and Applications.

Xiapu Luo, EdmondW.W. Chan, and RockyK.C. Chang. 2007. Cloak: A Ten-Fold Way for Reliable Covert Communications. In *Computer Security ESORICS 2007*, Joachim Biskup and Javier Lpez (Eds.). Lecture Notes in Computer Science, Vol. 4734. Springer Berlin Heidelberg, 283–298. DOI:http://dx.doi.org/10.1007/978-3-540-74835-9_19

Xiapu Luo, E. Chan, and R. Chang. 2008. TCP covert timing channels: Design and detection. In *Dependable Systems and Networks With FTCS and DCC, 2008. DSN 2008. IEEE International Conference on*. 420 –429. DOI:http://dx.doi.org/10.1109/DSN.2008.4630112

Xiapu Luo, E.W.W. Chan, Peng Zhou, and R.K.C. Chang. 2012a. Robust Network Covert Communications Based on TCP and Enumerative Combinatorics. *Dependable and Secure Computing, IEEE Transactions on* 9, 6 (Nov 2012), 890–902. DOI:http://dx.doi.org/10.1109/TDSC.2012.64

Xiapu Luo, Junjie Zhang, Roberto Perdisci, and Wenke Lee. 2010. On the Secrecy of Spread-Spectrum Flow Watermarks. In *Computer Security ESORICS 2010*, Dimitris Gritzalis, Bart Preneel, and Marianthi Theoharidou (Eds.). Lecture Notes in Computer Science, Vol. 6345. Springer Berlin Heidelberg, 232–248. DOI:http://dx.doi.org/10.1007/978-3-642-15497-3_15

Xiapu Luo, Peng Zhou, Junjie Zhang, Roberto Perdisci, Wenke Lee, and Rocky K. C. Chang. 2011. Exposing Invisible Timing-based Traffic Watermarks with BACKLIT. In *Proceedings of the 27th Annual Computer Security Applications Conference (ACSAC '11)*. ACM, New York, NY, USA, 197–206. DOI:http://dx.doi.org/10.1145/2076732.2076760

R. Martin, J. Demme, and S. Sethumadhavan. 2012. TimeWarp: Rethinking timekeeping and performance monitoring mechanisms to mitigate side-channel attacks. In *Computer Architecture (ISCA), 2012 39th Annual International Symposium on*. 118–129. DOI:http://dx.doi.org/10.1109/ISCA.2012.6237011

B.C. Mason, D. Ghosal, and C. Corbett. 2010. Evaluation of a Massively Parallel Architecture for Network Security Applications. In *Parallel, Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on*. 85–91. DOI:http://dx.doi.org/10.1109/PDP.2010.20

Petar Momcilovic. 2006. Mismatch Decoding of a Compound Timing Channel. In *Forty-Fourth Annual Allerton Conference on Communication, Control, and Computing*.

I.S. Moskowitz and A.R. Miller. 1992. The channel capacity of a certain noisy timing channel. *IEEE Transactions on Information Theory* 38, 4 (1992), 1339–1344.

S. Mou, Z. Zhao, S. Jiang, Z. Wu, and J. Zhu. 2012. Feature extraction and classification algorithm for detecting complex covert timing channel. *Computers and Security* 31, 1 (2012), 70–82. http://dblp.uni-trier.de/db/journals/compsec/compsec31.html#MouZJWZ12

Keaton Mowery, Sriram Keelveedhi, and Hovav Shacham. 2012. Are AES x86 Cache Timing Attacks Still Feasible?. In *Proceedings of the 2012 ACM Workshop on Cloud Computing Security Workshop (CCSW '12)*. ACM, New York, NY, USA, 19–24. DOI:http://dx.doi.org/10.1145/2381913.2381917

Jason Oberg, Wei Hu, Ali Irturk, Mohit Tiwari, Timothy Sherwood, and Ryan Kastner. 2010. Theoretical Analysis of Gate Level Information Flow Tracking. In *Proceedings of the 47th Design Automation Conference (DAC '10)*. ACM, New York, NY, USA, 244–247. DOI:http://dx.doi.org/10.1145/1837274.1837337

Jason Oberg, Wei Hu, Ali Irturk, Mohit Tiwari, Timothy Sherwood, and Ryan Kastner. 2011. Information Flow Isolation in I2C and USB. In *Proceedings of the 48th Design Automation Conference (DAC '11)*. ACM, New York, NY, USA, 254–259. DOI:http://dx.doi.org/10.1145/2024724.2024782

DagArne Osvik, Adi Shamir, and Eran Tromer. 2006. Cache Attacks and Countermeasures: The Case of AES. In *Topics in Cryptology CT-RSA 2006*, David Pointcheval (Ed.). Lecture Notes in Computer Science, Vol. 3860. Springer Berlin Heidelberg, 1–20. DOI:http://dx.doi.org/10.1007/11605805_1

Zheng Pan, Hong Peng, Xianzhong Long, Changle Zhang, and Ying Wu. 2009. A Watermarking-Based Host Correlation Detection Scheme. In *Management of e-Commerce and e-Government, 2009. ICMECG '09. International Conference on*. 493–497. DOI:http://dx.doi.org/10.1109/ICMeCG.2009.29

P. Peng, P. Ning, and D.S. Reeves. 2006. On the Secrecy of Timing-Based Active Watermarking Trace-Back Techniques. In *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*. Washington, DC, 334–349.

Pai Peng, Peng Ning, D.S. Reeves, and Xinyuan Wang. 2005. Active timing-based correlation of perturbed traffic flows with chaff packets. In *Distributed Computing Systems Workshops, 2005. 25th IEEE International Conference on*. 107–113. DOI:http://dx.doi.org/10.1109/ICDCSW.2005.30

Colin Percival. 2005. Cache missing for fun and profit. In *Proc. of BSDCan 2005*.

Young June Pyun, Younghee Park, Douglas S. Reeves, Xinyuan Wang, and Peng Ning. 2012. Interval-based flow watermarking for tracing interactive traffic. *Computer Networks* 56, 5 (2012), 1646 – 1665. DOI:http://dx.doi.org/10.1016/j.comnet.2012.01.017

Young June Pyun, Young Hee Park, Xinyuan Wang, D.S. Reeves, and Peng Ning. 2007. Tracing Traffic through Intermediate Hosts that Repacketize Flows. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*. 634–642. DOI:http://dx.doi.org/10.1109/INFCOM.2007.80

Sakthi V. Radhakrishnan, A. Selcuk Uluagac, and Raheem Beyah. 2013. Realizing an 802.11-based covert timing channel using off-the-shelf wireless cards. In *Global Communications Conference (GLOBECOM), 2013 IEEE*. 722–728. DOI:http://dx.doi.org/10.1109/GLOCOM.2013.6831158

C. Rebeiro, M. Mondal, and D. Mukhopadhyay. 2010. Pinpointing Cache Timing Attacks on AES. In *VLSI Design, 2010. VLSID '10. 23rd International Conference on*. 306–311. DOI:http://dx.doi.org/10.1109/VLSI.Design.2010.29

Ahmad-Reza Sadeghi, Steffen Schulz, and Vijay Varadharajan. 2012. The Silence of the LANs: Efficient Leakage Resilience for IPsec VPNs. In *Computer Security ESORICS 2012*, Sara Foresti, Moti Yung, and Fabio Martinelli (Eds.). Lecture Notes in Computer Science, Vol. 7459. Springer Berlin Heidelberg, 253–270. DOI:http://dx.doi.org/10.1007/978-3-642-33167-1_15

Erkay Savaş. 2013. Attacks on Implementations of Cryptographic Algorithms: Side-channel and Fault Attacks. In *Proceedings of the 6th International Conference on Security of Information and Networks (SIN '13)*. ACM, New York, NY, USA, 7–14. DOI:http://dx.doi.org/10.1145/2523514.2523593

Werner Schindler. 2000. A Timing Attack against RSA with the Chinese Remainder Theorem. In *Cryptographic Hardware and Embedded Systems CHES 2000*, etinK. Ko and Christof Paar (Eds.). Lecture Notes in Computer Science, Vol. 1965. Springer Berlin Heidelberg, 109–124. DOI:http://dx.doi.org/10.1007/3-540-44499-8_8

Werner Schindler. 2002. A Combined Timing and Power Attack. In *Public Key Cryptography*, David Naccache and Pascal Paillier (Eds.). Lecture Notes in Computer Science, Vol. 2274. Springer Berlin Heidelberg, 263–279. DOI:http://dx.doi.org/10.1007/3-540-45664-3_19

S.H. Sellke, C. Wang, S. Bagchi, and N. Shroff. 2009. TCP/IP Timing Channels: Theory to Implementation. In *INFOCOM 2009. The 28th Conference on Computer Communications. IEEE*. 2204–2212.

S. H. Sellke, N. B. Shroff, S. Bagchi, , and C. C. Wang. 2006. Timing Channel Capacity for Uniform and Gaussian Servers. In *Forty-Fourth Annual Allerton Conference on Communication, Control, and Computing*.

S. H. Sellke, C. Wang, N. Shroff, and S. Bagchi. 2007. Capacity Bounds on Timing Channels with Bounded Service Times. In *IEEE International Symposium on Information Theory*. 981–985.

Abdulhadi Shoufan, Falko Strenzke, H.Gregor Molter, and Marc Stttinger. 2010. A Timing Attack against Patterson Algorithm in the McEliece PKC. In *Information, Security and Cryptology ICISC 2009*, Donghoon Lee and Seokhie Hong (Eds.). Lecture Notes in Computer Science, Vol. 5984. Springer Berlin Heidelberg, 161–175. DOI:http://dx.doi.org/10.1007/978-3-642-14423-3_12

P.L. Shrestha, M. Hempel, M. Alahmad, and H. Sharif. 2013. Modeling packet rate covert timing channels. In *Innovations in Information Technology (IIT), 2013 9th International Conference on*. 54–59. DOI:http://dx.doi.org/10.1109/Innovations.2013.6544393

P.L. Shrestha, M. Hempel, H. Sharif, and H.-H. Chen. 2014. An Event-Based Unified System Model to Characterize and Evaluate Timing Covert Channels. (2014). DOI:http://dx.doi.org/10.1109/JSYST.2014.2328665

GustavusJ. Simmons. 1984. The Prisoners Problem and the Subliminal Channel. In *Advances in Cryptology*, David Chaum (Ed.). Springer US, 51–67. DOI:http://dx.doi.org/10.1007/978-1-4684-4730-9_5

D. X. Song, D. Wagner, and X. Tian. 2001. Timing analysis of keystrokes and timing attacks on SSH. In *SSYM'01: Proceedings of the 10th Conference on USENIX Security Symposium*. Berkeley, CA, USA, 25–25.

R.M. Stillman. 2008. Detecting IP covert timing channels by correlating packet timing with memory content. In *Southeastcon, 2008. IEEE*. 204–209. DOI:http://dx.doi.org/10.1109/SECON.2008.4494286

Falko Strenzke. 2010. A Timing Attack against the Secret Permutation in the McEliece PKC. In *Post-Quantum Cryptography*, Nicolas Sendrier (Ed.). Lecture Notes in Computer Science, Vol. 6061. Springer Berlin Heidelberg, 95–107. DOI:http://dx.doi.org/10.1007/978-3-642-12929-2_8

Falko Strenzke, Erik Tews, H.Gregor Molter, Raphael Overbeck, and Abdulhadi Shoufan. 2008. Side Channels in the McEliece PKC. In *Post-Quantum Cryptography*, Johannes Buchmann and Jintai Ding (Eds.). Lecture Notes in Computer Science, Vol. 5299. Springer Berlin Heidelberg, 216–229. DOI:http://dx.doi.org/10.1007/978-3-540-88403-3_15

Yanan Sun, Xiaohong Guan, Ting Liu, and Yu Qu. 2012. An Identity Authentication Mechanism Based on Timing Covert Channel. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*. 832–836. DOI:http://dx.doi.org/10.1109/TrustCom.2012.80

R. Sundaresan and S. Verdu. 2000a. Robust decoding for timing channels. *Information Theory, IEEE Transactions on* 46, 2 (Mar 2000), 405–419. DOI:http://dx.doi.org/10.1109/18.825800

R. Sundaresan and S. Verdu. 2000b. Sequential decoding for the exponential server timing channel. *Information Theory, IEEE Transactions on* 46, 2 (Mar 2000), 705–709. DOI:http://dx.doi.org/10.1109/18.825847

R. Sundaresan and S. Verdu. 2006. Capacity of queues via point-process channels. *Information Theory, IEEE Transactions on* 52, 6 (June 2006), 2697–2709. DOI:http://dx.doi.org/10.1109/TIT.2005.862079

J.A. Thomas. 1997. On the Shannon capacity of discrete time queues. In *Information Theory. 1997. Proceedings., 1997 IEEE International Symposium on*. 333–. DOI:http://dx.doi.org/10.1109/ISIT.1997.613261

M. Tiwari, Xun Li, H.M.G. Wassel, B. Mazloom, S. Mysore, F.T. Chong, and T. Sherwood. 2010. Gate-Level Information-Flow Tracking for Secure Architectures. *Micro, IEEE* 30, 1 (Jan 2010), 92–100. DOI:http://dx.doi.org/10.1109/MM.2010.17

M. Tiwari, J.K. Oberg, X. Li, J. Valamehr, T. Levin, B. Hardekopf, R. Kastner, F.T. Chong, and T. Sherwood. 2011. Crafting a usable microkernel, processor, and I/O system with strict and provable information flow security. In *Computer Architecture (ISCA), 2011 38th Annual International Symposium on*. 189–199.

Mohit Tiwari, Hassan M.G. Wassel, Bita Mazloom, Shashidhar Mysore, Frederic T. Chong, and Timothy Sherwood. 2009. Complete Information Flow Tracking from the Gates Up. In *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS XIV)*. ACM, New York, NY, USA, 109–120. DOI:http://dx.doi.org/10.1145/1508244.1508258

Eran Tromer, Dag Arne Osvik, and Adi Shamir. 2010. Efficient Cache Attacks on AES, and Countermeasures. *J. Cryptol.* 23, 2 (Jan. 2010), 37–71. DOI:http://dx.doi.org/10.1007/s00145-009-9049-y

A.B. Wagner and V. Anantharam. 2005. Zero-rate reliability of the exponential-server timing channel. *Information Theory, IEEE Transactions on* 51, 2 (Feb 2005), 447–465. DOI:http://dx.doi.org/10.1109/TIT.2004.840876

Robert J. Walls, Kush Kothari, and Matthew Wright. 2011. Liquid: A detection-resistant covert timing channel based on IPD shaping. *Computer Networks* 55, 6 (2011), 1217 – 1228. DOI:http://dx.doi.org/10.1016/j.comnet.2010.11.007

ColinD. Walter and Susan Thompson. 2001. Distinguishing Exponent Digits by Observing Modular Subtractions. In *Topics in Cryptology CT-RSA 2001*, David Naccache (Ed.). Lecture Notes in Computer Science, Vol. 2020. Springer Berlin Heidelberg, 192–207. DOI:http://dx.doi.org/10.1007/3-540-45353-9_15

Xinyuan Wang, Shiping Chen, and Sushil Jajodia. 2005. Tracking Anonymous Peer-to-peer VoIP Calls on the Internet. In *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS '05)*. ACM, New York, NY, USA, 81–91. DOI:http://dx.doi.org/10.1145/1102120.1102133

Xinyuan Wang, Shiping Chen, and S. Jajodia. 2007. Network Flow Watermarking Attack on Low-Latency Anonymous Communication Systems. In *Security and Privacy, 2007. SP '07. IEEE Symposium on*. 116–130. DOI:http://dx.doi.org/10.1109/SP.2007.30

Xiaogang Wang, Junzhou Luo, and Ming Yang. 2009. An interval centroid based spread spectrum watermark for tracing multiple network flows. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*. 4000–4006. DOI:http://dx.doi.org/10.1109/ICSMC.2009.5346664

Xiaogang Wang, Junzhou Luo, and Ming Yang. 2010. A Double Interval Centroid-Based Watermark for network flow traceback. In *Computer Supported Cooperative Work in Design (CSCWD), 2010 14th International Conference on*. 146–151. DOI:http://dx.doi.org/10.1109/CSCWD.2010.5471985

Xiaogang Wang, Junzhou Luo, and Ming Yang. 2012. An efficient sequential watermark detection model for tracing network attack flows. In *Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th International Conference on*. 236–243. DOI:http://dx.doi.org/10.1109/CSCWD.2012.6221824

Xinyuan Wang and D.S. Reeves. 2011. Robust Correlation of Encrypted Attack Traffic through Stepping Stones by Flow Watermarking. *Dependable and Secure Computing, IEEE Transactions on* 8, 3 (May 2011), 434–449. DOI:http://dx.doi.org/10.1109/TDSC.2010.35

Xinyuan Wang and Douglas S. Reeves. 2003. Robust Correlation of Encrypted Attack Traffic Through Stepping Stones by Manipulation of Interpacket Delays. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03)*. ACM, New York, NY, USA, 20–29. DOI:http://dx.doi.org/10.1145/948109.948115

Xiaogang Wang, Ming Yang, and Junzhou Luo. 2013. A novel sequential watermark detection model for efficient traceback of secret network attack flows. *Journal of Network and Computer Applications* 36, 6 (2013), 1660 – 1670. DOI:http://dx.doi.org/10.1016/j.jnca.2013.01.015

Yi Wang, Ping Chen, Yi Ge, Bing Mao, and Li Xie. 2009. Traffic Controller: A Practical Approach to Block Network Covert Timing Channel. In *Availability, Reliability and Security, 2009. ARES '09. International Conference on*. 349–354. DOI:http://dx.doi.org/10.1109/ARES.2009.141

Yao Wang, Andrew Ferraiuolo, and G.Edward Suh. 2014. Timing channel protection for a shared memory controller. In *High Performance Computer Architecture (HPCA), 2014 IEEE 20th International Symposium on*. 225–236. DOI:http://dx.doi.org/10.1109/HPCA.2014.6835934

Ying Wang and P. Moulin. 2008. Perfectly Secure Steganography: Capacity, Error Exponents, and Code Constructions. *Information Theory, IEEE Transactions on* 54, 6 (June 2008), 2706–2722. DOI:http://dx.doi.org/10.1109/TIT.2008.921684

Yao Wang and G.E. Suh. 2012. Efficient Timing Channel Protection for On-Chip Networks. In *Networks on Chip (NoCS), 2012 Sixth IEEE/ACM International Symposium on*. 142–151. DOI:http://dx.doi.org/10.1109/NOCS.2012.24

Zhenghong Wang and R.B. Lee. 2006. Covert and Side Channels Due to Processor Architecture. In *Computer Security Applications Conference, 2006. ACSAC '06. 22nd Annual*. 473–482. DOI:http://dx.doi.org/10.1109/ACSAC.2006.20

Zhenghong Wang and R.B. Lee. 2008. A novel cache architecture with enhanced performance and security. In *Microarchitecture, 2008. MICRO-41. 2008 41st IEEE/ACM International Symposium on*. 83–93. DOI:http://dx.doi.org/10.1109/MICRO.2008.4771781

Zhenghong Wang and Ruby B. Lee. 2007. New Cache Designs for Thwarting Software Cache-based Side Channel Attacks. In *Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA '07)*. ACM, New York, NY, USA, 494–505. DOI:http://dx.doi.org/10.1145/1250662.1250723

Hassan M. G. Wassel, Ying Gao, Jason K. Oberg, Ted Huffmire, Ryan Kastner, Frederic T. Chong, and Timothy Sherwood. 2013. SurfNoC: A Low Latency and Provably Non-interfering Approach to Secure Networks-on-chip. In *Proceedings of the 40th Annual International Symposium on Computer Architecture (ISCA '13)*. ACM, New York, NY, USA, 583–594. DOI:http://dx.doi.org/10.1145/2485922.2485972

J.C. Wray. 1991. An analysis of covert timing channels. In *Research in Security and Privacy, 1991. Proceedings., 1991 IEEE Computer Society Symposium on*. 2 –7. DOI:http://dx.doi.org/10.1109/RISP.1991.130767

J. Wu, Y. Wang, L. Ding, and X. Liao. 2012. Improving performance of network covert timing channel through Huffman coding. *Mathematical and Computer Modelling* 55, 1–2 (2012), 69–79. DOI:http://dx.doi.org/10.1016/j.mcm.2011.01.051 Advanced Theory and Practice for Cryptography and Future Security.

Zhao Xinjie, Wang Tao, Mi Dong, Zheng Yuanyuan, and Lun Zhaoyang. 2008. Robust First Two Rounds Access Driven Cache Timing Attack on AES. In *Computer Science and Software Engineering, 2008 International Conference on*, Vol. 3. 785–788. DOI:http://dx.doi.org/10.1109/CSSE.2008.633

L. Yao, X. Zi, L. Pan, and J. Li. 2009. A study of on/off timing channel based on packet delay distribution. In *Computers & Security*.

Wei Yu, Xinwen Fu, S. Graham, Dong Xuan, and Wei Zhao. 2007. DSSS-Based Flow Marking Technique for Invisible Traceback. In *Security and Privacy, 2007. SP '07. IEEE Symposium on*. 18–32. DOI:http://dx.doi.org/10.1109/SP.2007.14

Mengchao Yue, William H. Robinson, Lanier Watkins, and Cherita Corbett. 2014. Constructing Timing-based Covert Channels in Mobile Networks by Adjusting CPU Frequency. In *Proceedings of the Third Workshop on Hardware and Architectural Support for Security and Privacy (HASP '14)*. ACM, New York, NY, USA, Article 2, 8 pages. DOI:http://dx.doi.org/10.1145/2611765.2611768

S. Zander, G. Armitage, and P. Branch. 2007. A Survey of Covert Channels and Countermeasures in Computer Network Protocols. *Communications Surveys & Tutorials, IEEE* 9, 3 (2007), 44–57.

Sebastian Zander, Grenville Armitage, and Philip Branch. 2011. Stealthier Inter-packet Timing Covert Channels. In *NETWORKING 2011*, Jordi Domingo-Pascual, Pietro Manzoni, Sergio Palazzo, Ana Pont, and Caterina Scoglio (Eds.). Lecture Notes in Computer Science, Vol. 6640. Springer Berlin Heidelberg, 458–470. DOI:http://dx.doi.org/10.1007/978-3-642-20757-0_36

Lu Zhang, Junzhou Luo, and Ming Yang. 2009. An Improved DSSS-Based Flow Marking Technique for Anonymous Communication Traceback. In *Ubiquitous, Autonomic and Trusted Computing, 2009. UIC-ATC '09. Symposia and Workshops on*. 563–567. DOI:http://dx.doi.org/10.1109/UIC-ATC.2009.76

Liancheng Zhang, Zhenxing Wang, Qinglong Wang, and Fu Miao. 2010b. MSAC and Multi-flow Attacks Resistant Spread Spectrum Watermarks for network flows. In *Information and Financial Engineering (ICIFE), 2010 2nd IEEE International Conference on*. 438–441. DOI:http://dx.doi.org/10.1109/ICIFE.2010.5609393

Liancheng Zhang, Zhenxing Wang, Yu Wang, and Huisheng Liu. 2010a. Interval-Based Spread Spectrum Watermarks for tracing multiple network flows. In *Communication Technology (ICCT), 2010 12th IEEE International Conference on*. 393–396. DOI:http://dx.doi.org/10.1109/ICCT.2010.5688820

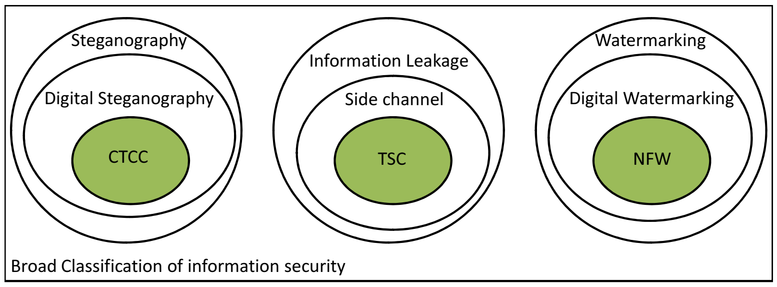
Xiaochao Zi, Lihong Yao, Xinghao Jiang, Li Pan, and Jianhua Li. 2011. Evaluating the transmission rate of covert timing channels in a network. *Computer Networks* 55, 12 (2011), 2760 – 2771. DOI:http://dx.doi.org/10.1016/j.comnet.2011.05.018

Xiaochao Zi, Lihong Yao, Li Pan, and Jianhua Li. 2010. Implementing a passive network covert timing channel. *Computers and Security* 29, 6 (2010), 686 – 696. DOI:http://dx.doi.org/10.1016/j.cose.2009.12.010

APPENDIX

A. PRELIMINARIES

According to [Department of Defense Standard 1985], covert timing channel is defined as "A covert channel in which one process signals information to another by modulating its own use of system resources (e.g., CPU time) in such a way that this manipulation affects the real response time observed by the second process." In this paper, we focus on timing channels. To the best of our knowledge, the first timing channel was reported in 1976 [Bell and Padula 1976]. It was an interval based timing channel where a long interval between two events indicated bit 1 and a short interval indicated bit 0. While the term timing channel was not used, it was described as an indirect communication path meant to disclose sensitive information.



CTCC: Covert timing channel communication, TSC: Timing side channel, NFW: Network flow watermarking

Fig. 10. Different applications of timing channels in the world of information security.

From a broad information security perspective, timing channels have been applied in steganography (information hiding), information leakage, and watermarking. As

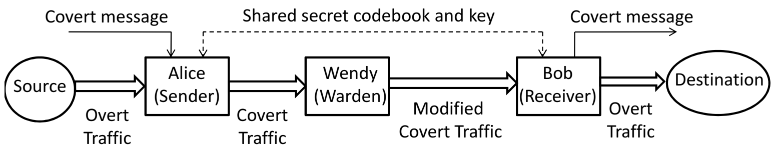


Fig. 11. A generic model of a covert timing channel showing the various entities. This figure is adapted from the figure in [Zander et al. 2007] where it is used to represent covert channels including CTCC.

shown in Figure 10, covert timing channel communication (CTCC) is a type of digital steganography where a sender sends covert messages using a timing channel. In the context of information leakage, timing side channel (TSC) is a type of side channel where information is leaked due to faulty or vulnerable operation of a system or by a timing attack. Finally, in the domain of digital watermarking, network flow watermarking (NFW) can be implemented using some underlying timing channel. Whereas for both CTCC and NFW there is a sender (embedder) and a receiver (decoder), in the case of TSC a sender may not be present.

In another orthogonal categorization, timing channels can be classified as network timing channel and in-system timing channel. Network timing channel is implemented over a local area network (LAN) or the Internet whereas in-system timing channel is implemented inside a computer system or a chip. Both CTCC and TSC have been studied and implemented for both in-system and networked systems. With regard to flow watermarking, its applicability is primarily in networked systems and hence referred to as network flow watermarking.

A.1. Key Entities and Definitions

CTCC was first proposed by Simmons as the prisoners' problem in 1984 [Simmons 1984]. The original prisoners' problem which was proposed to resemble the authentication without secrecy [Simmons 1984] can be described as follows. Alice and Bob are two prisoners who want to flee the prison. Wendy is a warden who suspects that the two prisoners are planning to flee but she does not have any proof. To get the proof, Wendy allows Alice and Bob to communicate with a requirement that the messages must be fully open to her and innocuous in nature. It is Alice and Bob's goal to communicate secret messages hidden in the innocuous communication in a way that Wendy cannot detect the hidden secret messages.

Figure 11 shows the key entities in a CTCC. Alice and Bob are two endpoints of the timing channel and they are referred to as the sender and the receiver, respectively. The endpoints of the overt communication are referred to as the source and the destination. The warden Wendy can be one of three types [Zander et al. 2007]: passive, active, or malicious. A passive warden simply monitors the messages that are exchanged by Alice and Bob, an active warden can modify the messages, and a malicious warden can change the messages without being detected. Capabilities of the warden can be understood more clearly with the help of different threat models. Without a systematic view of threats to timing channels, analyzing and comparing timing channels is hard. Detailed discussions on threat models are given in Appendix B. Although the model in Figure 11 shows a simplex unicast (one-to-one) channel, a CTCC can be duplex channel [Archibald and Ghosal 2012] or multicast (one-to-many) channel [Zander et al. 2007].

While Figure 11 shows different logical locations of the sender and the receiver with respect to the source and the destination, they may or may not correspond to a different physical locations. Logically separate parties maybe located on the same computer but in different OSI layers. On the other hand, the sender and the receiver may be located in routers between the source and the destination. Moreover, CTCC can be active and passive [Gianvecchio and Wang 2007]. An active CTCC refers to the configuration where the sender and the source are same entity and consequently the sender can generate overt traffic as required by the timing channel. Passive CTCC, on the other hand,

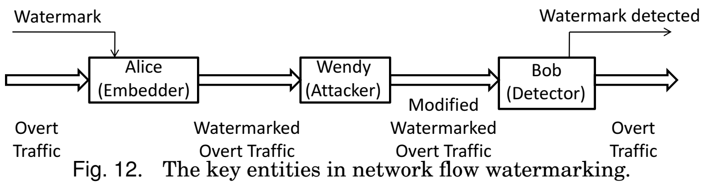


Fig. 12. The key entities in network flow watermarking.

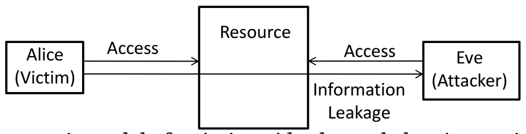


Fig. 13. A generic model of a timing side channel showing various entities.

refers to the scenario where the sender and the source are different entities and the source generates overt traffic independent of the requirements of the sender. An active CTCC is faster than a passive CTCC but passive CTCC is harder to detect. A warden can be located anywhere between the sender and the receiver i.e., on the same computer as source, on a router/gateway in the path between the sender and the receiver, or at the destination.

Figure 12 shows the various entities of NFW. Here the covert sender Alice represents the watermarker or the encoder that embeds watermarks into the overt traffic. The covert receiver Bob represents the watermark detector or the decoder that decodes the presence of watermarks from the network traffic. The attacker Wendy attempts to thwart the ability of Bob to accurately detect the watermark and trace it back to Alice. The specific roles of the source and the destination which are typically aligned with the attacker Wendy, depend on the specific use case of NFW.

While digital watermarking is a very well established field, NFW based on timing channels have recently received a lot of attention [Houmansadr et al. 2009a; Gong et al. 2013]. One of the main focus has been on using NFW to detect stepping stones [Gong et al. 2013]. Another application is to de-anonymize networks that attempt to anonymize communication such as those achieved using Tor [Dingledine et al. 2004]. The role of the adversary (Wendy) is similar to the case of CTCC with goal of detecting and/or disrupting the watermark.

Figure 13 shows the key entities of TSC. Here the victim Alice unintentionally leaks information to the observer Eve who is the attacker. Alice accesses the resource for her own operation depending upon some secret information. The same resource is also accessed by Eve. This results in contention at the resource due to access of the same resource by both Alice and Eve. Now Eve can interpret Alice's timing behavior by observing variation of her own timing characteristics. This leads to the leakage of the secret information of Alice with high probability. In some cases (like in chosen plaintext attack) an optional observer input is also applied to the victim. Some examples of the shared resource in Figure 13 are the cache in the memory system, the memory controller, and a scheduler. For in-system TSCs, both the victim and the observer are located in the same computer system. In a network based TSC, referred to as the RTSC, the observer is located outside the victim's computer system but on the same network. In most TSCs, knowledge of the victim's hardware/software architecture is necessary for the observer to interpret the secret information from victim's timing behavior. Generally, the attacker's aim is to infer secret cryptographic keys exploiting a vulnerability in cryptographic algorithm's actual implementation. Here the cryptographic algorithm implementation represents the victim.

A.2. Timing Channel Requirements

The basic common requirements of a timing channel are **non-detectability** and **non-disruptability**. Of these two, non-detectability has a common definition across different applications of timing channels, whereas the definition of non-disruptability depends on the specific application.

Non-detectability[1] implies that the warden cannot identify the existence of a timing channel. In the context of a CTCC, a timing channel is termed Polynomial not-detectable regarding a security variable δ provided that there exists a negligible function $\nu(\delta)$ such that $|T(d) - T(d_s)| \leq \nu(\delta)$ for some probabilistic polynomial-time statistical test T [Liu et al. 2010; Cachin 2004], where d and d_s are arbitrary N samples of the timing feature of the overt and covert traffic, respectively. We assume that N is a positive integer. Polynomial time statistical tests include Kullback-Leibler (KL) divergence test [Cachin 2004; Archibald and Ghosal 2014], Kolmogorov-Smirnov (KS) test [Cabuk et al. 2009] among others.

Another property that is closely related to non-detectability is **non-disclosure** which ensures that a warden cannot decode the secret information if the timing channel is detected. Non-detectability of a timing channel guarantees its non-disclosure[2]. In the context of CTCC, covert messages are generally cryptographically encoded to enhance non-disclosure of CTCC.

The timing channel is exposed to channel noise both purposefully injected and those that are inherent in the channel. **Non-disruptability** refers to property of correct reception of secret information in spite of the presence of noise in the channel. In literature, Robustness is also used to refer to the same term. According to [Liu et al. 2010], the ability to get a "bit error rate" (BER) $P_e \leq \varepsilon$ can be used to measure the robustness subject to a robustness condition $\varepsilon \in \mathbb{R}^+$. The "Signal-to-Noise Ratio" (SNR) and P_e are inversely proportional to each other.

A.2.1. Requirements of CTCC. Both non-detectability and non-disruptability are important requirements for a CTCC. Additionally, like any other information transmission channel, high capacity (secret information transfer rate) is a desirable property of CTCC [Archibald 2013]. Covert messages are generally cryptographically encoded to enhance non-disclosure of CTCC. Generally error correcting codes are used to detect and correct any bit error in CTCC to improve robustness. The main challenge lies in the fact that it is very difficult to achieve all desired characteristics simultaneously. For example, high CTCC capacity can be achieved by significant modification of the timing behavior of the overt channel. But this can be easily detected by the warden causing reduction of the non-detectability property.

A.2.2. Requirements of network flow watermarking. Non-disruptability is the most important requirement for NFW. It ensures that the watermark in a flow cannot be disrupted by an attacker. Non-detectability is also a required property for NFW. If the watermark is detected, it can be removed or replicated. It is to be noted that an attacker may try to disrupt watermark even without successfully detecting it in a flow. Capacity is not an important requirement as only a small watermark needs to be embedded. Non-disclosure is also not relevant for NFW because once the presence of watermark is detected, it can be removed or replicated. Disclosing of exact information (watermark) will not do any additional harm. Non-disclosure is relevant in case of flow fingerprinting where the information can be used to know exactly which flow is marked. In this case, just the knowledge of the presence of the watermark will not help to identify the exact flow that is marked.

A.2.3. Requirements of timing side channel. Similar to CTCC, non-detectability, non-disruptability, non-disclosure, and high capacity are desirable characteristics for an effective timing side channel technique. Non-detectability ensures that a warden cannot detect the presence of TSC and non-disclosure ensures that the warden cannot decode

[1]In literature the terms invisibility and undetectability are also used. In this paper we use the term non-detectability to mean the same.
[2]In literature the term non-decodability is also used. In this paper we use the term non-disclosure to mean the same.

the leaked secret information from TSC. The observer or the receiver of secret information cannot improve non-detectability, and non-disclosure properties as per requirement because the secret information is leaked by timing behavior of a faulty operation. Generally large number of observations are taken to reduce the error probability and improve the reliability of the extracted information (improved robustness). Sometimes experiments are repeated to cancel out the error. Different statistical techniques are also used to reduce error from the large number of observations. Though high capacity is a desirable characteristic but in reality the timing side channel capacity is very low because a large number of repeated observations are required to gain enough reliability of extracted secret information. Similar to CTCC, in case of timing side channel also high capacity and high robustness cannot be achieved simultaneously.

B. THREAT MODELS

We assume that the warden controls an intermediary location between the covert sender and the covert receiver. This is a realistic assumption for network timing channels and also for covert storage channels. Different types of warden models can be constructed on the basis of control they can exert on the different parts of the timing channel. In the following text, the use of the term *channel* by itself means a communication channel. To refer to the covert exchange of messages we use the term *timing channel*.

The objective of the warden is to systematically determine the presence of a timing channel, or even more drastically to deduce the hidden content being communicated. Inspired by attacks on encryption schemes, we consider the following.

(1) A passive *output-only* warden tries to detect a timing channel by only observing the channel output. This is the weakest threat model, therefore any timing channel implementation vulnerable to this type of attack is completely insecure. In this scenario, the warden is unable or unwilling to exert any other control. Such an assumption may be unrealistic as a warden may have access to additional information about the communication channel. It's worth noting that a significant number of papers from current literature discuss timing channel mechanisms with this very limited threat model. Various statistical techniques have been proposed on isolated channel outputs. These techniques incorporate tests such as entropy and distribution-based approaches for channel detection. Much more powerful wardens exist and this has seen very little attention so far. Multiflow attacks [Kiyavash et al. 2008] have been proposed where the warden aggregates channel output over a period of time before starting analysis.

(2) A *known-input* warden is aware of channel input and the corresponding channel output but does not control what goes into channel input. Such a warden is quite powerful, he/she observes arbitrary flows both before and after the encoding process and tries to detect the presence of timing information apart from expected or natural distortion such as reordering within the channel, congestion, and loss (which are common in network-based channels). A powerful variant of this warden is one who considers multiple input-output message sets, hence building up an apriori distribution of possible outputs against possible inputs, and then examines the impact of the encoding process on a fraction of the channel inputs. Such attacks are proposed in [Lin and Hopper 2012].

(3) A *chosen-input* warden chooses the channel input and observes the corresponding output. Subsequently, the warden uses any information deduced for detecting timing channels in previous outputs. This is a powerful attack wherein the warden is able to inject inputs with specific timing pattern optimal for studying the encoding function applied, whose output is also available to the warden. [Lin and Hopper 2012] first proposed these attacks in the context of watermarking.

(4) An *adaptive chosen-input* warden is a chosen-input warden wherin the choice of channel input may depend on the channel output received previously. This allows

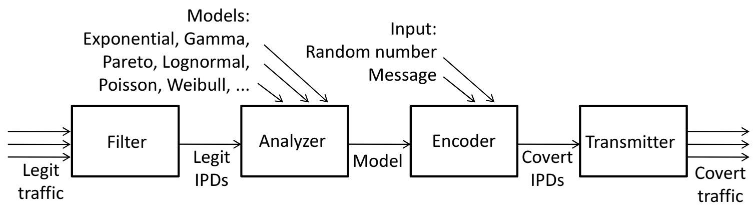


Fig. 14. Model-based CTCC building framework [Gianvecchio et al. 2008].

the warden to start with a simple timing pattern which reveals some information about the encoder and progressively fine-tunes the pattern to reveal the encoding function. Such attacks have not been proposed thus far.

(5) A *chosen-output* warden is given the channel input corresponding to an observed output. Such wardens are very powerful but not unrealistic. The warden gains access to the decoder (but not any keys) and is able to convince the decoder to decode a given encoder output, any number of times. Having this capability already means the warden can detect whether a channel output has covert messages. However, the objective is to then uncover any keys used within the timing channel implementation so that the warden can then encode arbitrary messages to confuse the recipient. Such an attack is particularly useful in the context of watermarking, where the warden can mark all messages as watermarked effectively jamming the channel. Thus far, no attacks have been proposed for this warden model.

(6) An *adaptive chosen-output* warden is a chosen-output warden where the choice of outputs the warden asks to be decoded are dependent on the results of previous requests to the decoder. Such wardens are amongst the most powerful, and no concrete attacks within the model have been proposed thus far in the literature.

All of the above mentioned threat models are applicable to flow watermarking techniques. A subset of models may be applicable to CTCCs also. In case of timing side channel (both network and in-system), there is no obvious sender and receiver like flow watermarking. The information is actually leaked unintentionally from timing behavior of victim. In fact some of the threat models are used by observer/attacker to extract the secret information from victim's timing behavior. Existing detection and prevention techniques of timing side channels try to either make the victim itself robust in terms of information leakage or try to detect all instances of interference from where an attacker may infer some information about the victim's timing characteristics. Knowledge of the threat models can help to devise more effective detection and prevention techniques.

C. AN EXAMPLE OF TIME-REPLAY CTCC

Let us assume that Alice and Eve communicates using a time-replay covert channel where Alice is the sender. Let us also assume that Alice transfers a code C that has symbols s_1 and s_2. Alice divides a prerecorded event sequence into two partitions : s_1 partition, and s_2 partition. Next, the s_1-partition is used to get a timing data τ_{s1} for sending symbol s_1. The symbol is embedded by delaying an event for τ_{s1} time. Similar procedure is followed to send symbol s_2 using a timing value from the s_2-partition. A specific timing value is not used more than once. At the receiver, Eve i) observes the time τ between events, ii) decides the exact partition that contains τ, and iii) decodes the symbol s_1 or s_2 based upon the result of second step.

D. MODEL BASED CTCC FRAMEWORK

The model based CTCC framework consists of filter, analyzer, encoder, and transmitter as shown in Figure 14. The filter first extracts specific type of traffic from legitimate traffic and then measures the IPDs from each flow. Next the filter results are forwarded to the analyzer so that the analyzer can decide the best traffic model. The analyzer tries to fit the legitimate traffic with the Weibull, Poisson, Lognormal, Pareto, Gamma, and Exponential distributions. Both offline and online execution is possible for the filter and analyzer stages. Both the online and the offline modes has their advantages and disadvantages. Fewer resources are required in the offline mode, but the current network traffic model may be different. The online mode requires more resources and a startup protocol to send the online model and parameters to the decoder, but the live network flow is closely approximated by the model. The encoder generates IPDs based on the model, a random number series, and a covert message. Then the transmitter sends packets according to the generated IPDs. The generated covert traffic approximates the legitimate traffic distribution to increase stealthiness.

E. COMPRESSIBILITY METRIC

Let $S \in \Sigma^s$ is a string to be compressed with Σ denoting the alphabet of symbols, and the length of S is s. Let $C \in \Sigma^c$ denotes the final compressed string. Cabuk et al. have defined the compressibility of S as

$$\kappa(S) = \frac{|S|}{|C|} \tag{1}$$

where $\kappa(.)$ denotes the compressibility operator, and $|.|$ denotes the length operator. It is reported that the compressibility generated by the legitimate channel IPDs is less than that generated by noiseless IP CTCC IPDs. It is to be noted that IPDs are numerical values and strings only can be compressed. So IPDs are converted to strings for compression. Cabuk et al. have considered the two significant non-zero digits of an IPD value for conversion and they have used a letter before the digits. This letter is decided by counting the zeros after the decimal point. Following the procedure 0.00247 becomes B25, and 0.0247 becomes A25.

The CTCC sender may insert noise to make the regularity metric comparable to legitimate channel. For this special case, Cabuk et al. have proposed two detection methods called “compressibility-walk” and “CosR-walk”. They have shown that the two methods are appropriate for offline measurement because of their high algorithmic complexity. In compressibility-walk a mixed data set is considered whose size is s and compressibility scores are calculated for every w size window. These windows are overlapped if s < w. If a window contains IPDs embedded with covert message, it generates higher compressibility scores compared to legitimate IPDs. In case of CosR-walk, CosR scores are calculated instead of compressibility scores for consecutive windows. Again, windows can be overlapped. CosR score is lower when a window contains legitimate IPDs and the other window contains IPDs with covert message compared to the score when both windows contain legitimate IPDs. CosR metric between two strings S and T can be defined as [Cabuk et al. 2009]

$$CosR(S,T) = 1 - \frac{\Im(S) + \Im(T) - \Im(S|T)}{\sqrt{\Im(S)\Im(T)}} \tag{2}$$

Here $\Im$ is a compressor and $|$ is a concatenation operator.

F. VARIOUS FLOW TRANSFORMATION TECHNIQUES

Figure 15 shows various intra-flow transformation techniques. Figure 15(a) shows addition of chaff packet or bogus packet addition to the flow. Figure 15(b) shows packet dropping. Figure 15(c) and (d) shows repacketization techniques i.e. packet merging, and fragmentation respectively. Figure 16 shows various inter-flow transformation techniques. Figure 16(a) shows flow mixing where a mixed flow f_0' is created by mixing a flow f_0 with separate flows: $f_1, ..., f_n$. Figure 16(b) shows flow splitting where many subflows: $f_0^1, ..., f_0^n$ are created by splitting a flow f_0. These split subflows can again be merged as in Figure 16(c).

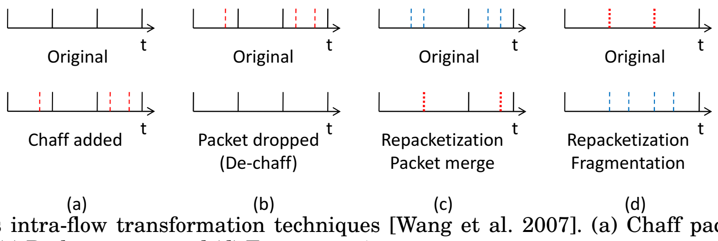


Fig. 15. Various intra-flow transformation techniques [Wang et al. 2007]. (a) Chaff packet addition, (b) Packet dropped, (c) Packet merge, and (d) Fragmentation.

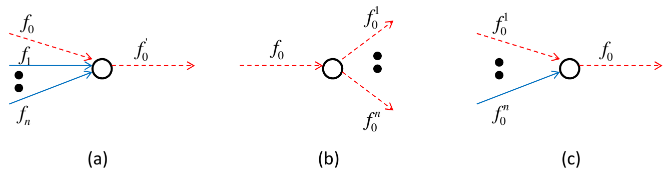


Fig. 16. Various inter-flow transformation techniques [Wang et al. 2007]. (a) Flow mixing, (b) Flow split, and (d) Flow merge.

G. CENTROID OF AN INTERVAL

Let us assume that there is a flow of duration $T_f > 0$ and l-bit watermark is embedded in the flow with redundancy $r > 0$. Also let us assume that a time period T_d can be divided among 2n (here $n = r \times l$) periods having duration $T(T > 0) : I_0, ..., I_{2n-1}$. Let us consider that the 2n periods contain packets $P_1, ..., P_{n_p}$ (total n_p packets) and $t_i(i = 1, ..., n_p)$ is the packet P_i's time instance. Next $t_i' = t_i - t_0$ represents P_i's relative time instance relative to the initial period t_0. Next the relative position of P_i within its period (i.e. the difference relative to beginning of its period) is calculated by Δt_i

$$\Delta t_i = t_i' \bmod T \tag{3}$$

Let us assume that period $I_i(i = 0, ..., 2n - 1)$ contains packets $P_{i_0}, ..., P_{i_{n_i-1}}$ (total n_i packets), then the "centroid of interval[3]" $I_i(i = 0, ..., 2n - 1)$ can be defined as [Wang et al. 2007]

$$Cent(I_i) = \frac{1}{n_i} \sum_{j=0}^{n_i-1} \Delta t_{i_j} \tag{4}$$

H. DSSS BASED FLOW WATERMARKING SYSTEM

Figure 17(a) shows the transmitter side DSSS based mark creation module and Figure 17(b) shows the receiver side DSSS based mark identification module as presented in [Yu et al. 2007]. The transmitter generates "+1" (logical bit 1) or "-1" (logical bit 0) of signal d_t. Next a PN code c_t having T_c chip duration (a chip is a PN code bit) is multiplied with the original signal d_t to obtain baseband signal t_b.

$$t_b = d_t c_t \tag{5}$$

The baseband signal t_b is then passed to the flow modulator for modulating the flow. For +1 chip, the flow traffic rate D is increased by applying weak interference against the flow for T_c seconds and obtain high rate (D + A). Similarly, for -1 chip, strong interference is applied and a low traffic rate of (D - A) is obtained for T_c seconds. Here A denotes the mark amplitude. The transmitted signal t_x from the flow modulator can be expressed as

$$t_x = Ad_t c_t + D \tag{6}$$

[3]The terms interval and period are used interchangeably.

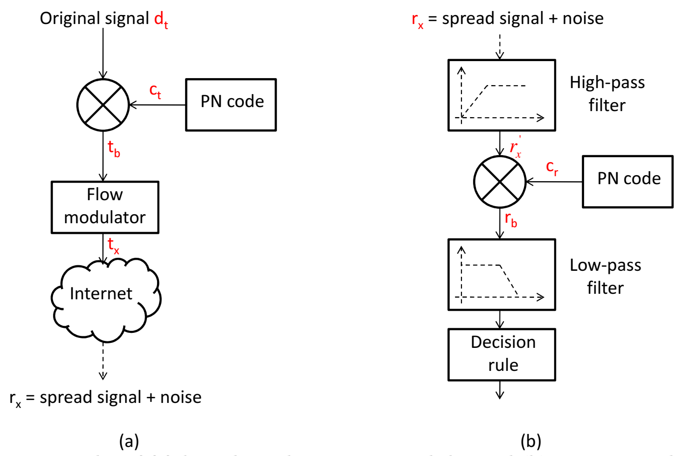


Fig. 17. (a) Transmitter side DSSS based mark creation module, and (b) Receiver side DSSS based mark identification module. [Yu et al. 2007]

The transmitted signal gets affected by noise while traveling through Internet before reaching the receiver. The noise is generated due to deliberate or intersecting traffic interference. If we denote noise by w and the signal after reception as r_x, then

$$r_x = Ad_tc_t + D + w \tag{7}$$

The received signal r_x is passed to a high-pass filter to get

$$r_x' \approx Ad_tc_t + w \tag{8}$$

Note that the DC component D is eliminated in the above filtered signal expression.

The filtered signal r_x' is multiplied with a PN code c_r for despreading and deriving the baseband signal r_b

$$r_b = Ad_tc_tc_r + wc_r \tag{9}$$

Note that the PN code c_r is identical to the PN code used in the transmitter.

The baseband signal r_b is then passed to a low-pass filter to get the low frequency signal. Next a decision rule helps to interpret the transmitted information from the filtered signal.