Chapter 12
On Creating Digital Evidence in IP Networks With NetTrack

Diana Berbecaru
Politecnico di Torino, Italy

ABSTRACT

Computer forensic is the practice of collecting, analyzing, and reporting digital evidence in a way that is legally admissible in open court. Network forensics, an offset of computer forensic, is mainly concerned with the monitoring and analysis of network traffic, both local and WAN/internet, in order to identify security incidents and to investigate fraud or network misuse. In this chapter, the authors discuss challenges in creating high-speed network forensic tools and propose NetTrack, a tamper-proof device aimed to produce evidences with probative value via digital signatures for the network traffic. Since digitally signing each IP packet is not efficient, the authors used a specific technique exploiting the Merkle trees to create digital signatures for flows and multicasts and implemented it by using an optimized algorithm for Merkle tree traversal to save space and time. Through experiments, the authors show NetTrack signing is fast as it can produce digital evidence within a short time.

INTRODUCTION

The Scientific Working Group on Digital Evidence (SWGDE, 2017) defines a "digital evidence" as "information of probative value that is stored or transmitted in binary form". Examples of "digital evidence" are: application files (like tests, images), system files (logs) or data ignored by the operating systems but stored on disks. This chapter addresses the problem of creating digital evidence for the network traffic, which is information with probative value about the network activity monitored at the interface of a network node (e.g. router, computer) or on a network link.

Such kind of digital evidence could be very useful in applications requiring both accurate tracing of data as well as the ability to proof its correctness in court. We give in this sense a simple example: two clients C1 and C2 connect almost simultaneously to a company server S providing time-sensitive services (e.g. stock option transactions). Even though the client C1 sends connection requests to S before C2, the IP packets are not guaranteed to arrive at S in this order: the network could actually delay

DOI: 10.4018/978-1-5225-4100-4.ch012

packets of C1 (with respect to the ones of C2) either involuntarily due to the processing of the packets in the intermediate routers, or deliberately due to attacks performed by malware users. In case a dispute is raised, the server S can exploit his internal logs to prove the quality of the service provided to his users. Such logs typically contain timestamped packets, which contain basically an association between the data arrived at the server S' network interface and the time indicated by the internal clock of the system S. In some situations, like for example in some banking applications, the information stored in logs could be accepted as proof in court even though the internal log has not been actually created in such a way to provide non-repudiation of the data stored and even though it could be possible for the internal log to be incomplete, in the sense that parts of the packets might have been lost and not recorded in the internal log.

In other critical scenarios (cyberattacks, financial services), it is highly required to know the *arrival timing* information of a packet. Thus, in this case, it is necessary to trace also the time when the packet is present on the network link or at a device interface. However, such applications do not have actually very strict constraints on the time precision, but rather on the authenticity and/or non-repudiation of time (who states for the time) and the packets' ordering. Thus, companies or entities are highly interested in a solution that can be used to create digital evidence with probative value for their own internal network activity (e.g. to compute forensic analysis) as well as of the client's and that could be subsequently used in case of disputes.

Nowadays, the network monitoring tools do not address the creation of digital evidences asserting packet's arrival time for the network activity. Most of the network monitoring techniques and tools developed so far provide typically essential inputs towards performance tuning, they are normally used to identify and reduce network bottlenecks, troubleshooting, as well as to identify, diagnose and rectify faults, or to perform planning. Such tools are also used to predict the scale and nature of necessary additional resources, to characterize network activity in order to supply data for network modeling and simulation, and to identify and correct the pathological network behavior. We will present some of these techniques and tools further below.

This chapter presents the first steps towards developing the NetTrack device, which is a tamper-proof device that could be employed in several use case scenarios requiring digital evidences of network activity, such as to provide the proof of quality of service, to achieve advanced IP traceback, in the Value Added Networks, or to perform network forensics. The NetTrack operates basically on two elementary pieces of data inputs, the IP packet and the time, and is composed of several modules, like a Network Sampler used to interface with the network node/link to capture and filter packets, an ACTS (Authenticated and Certified Time Source) used to obtain an authenticated and certified time from a time source, a NetTrack Evidence Creator module used to digitally sign the network traffic, and one used to store the evidences for short-term. In this chapter we focus mainly on the NetTrack Evidence Creator and we discuss its design, implementation and the results obtained in testing its performance.

The NetTrack Evidence Creator is the core application of the NetTrack, and is in charge with digitally signing the packet flows. In the current approach, we adopted the (hash) tree chaining technique presented in Wong and Lam (1999) for signing and verifying multiple IP packets. This technique uses a single signing/verifying operation to sign and verify a group of IP packets, called a *block*. In practice, an authentication tree or a so-called "Merkle tree" data structure (Merkle, 1982) is exploited to calculate a "block digest", which is digitally signed by using an asymmetric algorithm, e.g. RSA (Jonsson & Kaliski, 2003), and the appropriate signer's asymmetric key. Since this technique exploits the asymmetric cryptography, it may be used in scenarios requiring data integrity and authentication, and (possibly) the non-repudiation of the data. The Merkle trees have been used so far in many cryptographic applications,