HELSINKI UNIVERSITY OF TECHNOLOGY
Faculty of Information and Natural Sciences
Department of Information and Computer Science

Dmitrij Lagutin

Redesigning Internet - The Packet Level Authentication architecture

Licentiate's Thesis

Espoo, May 26, 2008

Supervisor: Professor Kaisa Nyberg

Instructor: D. Sc. Hannu H. Kari

HELSINKI UNIVERSITY OF TECHNOLOGY
Faculty of Information and Natural Sciences
Department of Information and Computer Science

ABSTRACT OF LICENTIATE THESIS

Author
Dmitrij Lagutin

Date
26.5.2008

Pages
95

Title of thesis
Redesigning Internet - The Packet Level Authentication architecture

Professorship
Theoretical Computer Science

Professorship Code
T-79

Supervisor
Professor Kaisa Nyberg

Instructor
D.Sc. Hannu H. Kari

Abstract

The Internet was originally designed to be secure against external attacks while the possibility of internal threats was mostly ignored. As a result, the Internet is currently vulnerable to a variety of malicious behaviors originating from within the network, such as denial of service attacks, phishing attacks, and unsolicited e-mail. Existing end-to-end security solutions like IPSec only offer protection to communication end points and are unable to secure the underlying network infrastructure against attacks. Thus, they are ineffective if the network infrastructure is attacked and is unable to deliver valid traffic in a reliable manner.

This thesis describes Packet Level Authentication (PLA), a novel network-level solution designed to protect the network infrastructure against various attacks. PLA is based on public key digital signature techniques and it gives every network node an ability to validate the authenticity and integrity of every received packet independently, without previously established trust relation to the sender or intermediate nodes that have handled the packet. PLA is designed to be compatible with the existing Internet infrastructure, allowing gradual deployment of PLA. In addition, PLA can be used in conjunction with existing end-to-end security solutions such as IPSec and Host Identity Protocol (HIP).

The original design goal of PLA was to create a solution that is scalable from energy and performance-constrained devices, such as mobile phones and sensors, to core network routers that can transfer tens of gigabits of traffic per second. Thus, the key challenge during design of PLA has revolved around optimizing digital signature calculations for both performance and energy consumption. This thesis shows that the new digital signature methods and their hardware implementations developed for PLA make PLA suitable for core network routers. Moreover, the amount of energy required to validate packet integrity is significantly lower than the energy consumed by wireless transmission. Therefore, PLA is also applicable to mobile devices.

This thesis also examines other possible applications for PLA beyond securing the Internet infrastructure.

Keywords
network security, denial of service attack, Internet Protocol, Internet infrastructure, digital signature algorithms, elliptic curve cryptosystems

TEKNILLINEN KORKEAKOULU
Informaatio- ja luonnontieteiden tiedekunta
Tietojenkäsittelytieteen laitos

LISENSIAATINTUTKIMUKSEN TIIVISTELMÄ

Tekijä
Dmitrij Lagutin

Päiväys
26.5.2008

Sivumäärä
95

Työn nimi
Redesigning Internet – The Packet Level Authentication architecture

Kieli
Englanti

Professuuri
Tietojenkäsittelyteoria

Professuurin koodi
T-79

Työn valvoja
Professori Kaisa Nyberg

Työn ohjaaja
TkT Hannu H. Kari

Tiivistelmä

Internet on alun perin suunniteltu olemaan turvallinen ulkopuolisia hyökkäyksiä vastaan, mutta sisäisten hyökkäysten mahdollisuus oli jätetty alkuperäisessä Internetin suunnittelussa huomioimatta. Tästä johtuen Internet on haavoittuvainen monille eri tyyppisille verkon sisältä tuleville vahingonteoille, kuten palvelunestohyökkäyksille, salasanojen kalasteluhyökkäyksille ja roskapostille. Olemassa olevat päästä-päähän tietoturvaratkaisut, kuten IPSec, turvaavat vain liikenteen päätepisteet, mutta eivät pysty suojaamaan alla olevaa verkkoinfrastruktuuria hyökkäyksiltä. Siten näistä ratkaisuista ei ole apua, jos verkkoinfrastruktuuri joutuu hyökkäyksen kohteeksi, eikä pysty toimittamaan hyödyllistä liikennettä perille.

Tämä työ kuvaa Packet Level Authentication (PLA)-menetelmää, joka on uusi, verkkotasolle sijoittuva ratkaisu verkkoinfrastruktuurin suojaamiseksi monenlaisia hyökkäyksiä vastaan. PLA pohjautuu julkisen avaimen allekirjoitusmenetelmiin ja se antaa jokaiselle verkossa olevalle solmulle mahdollisuuden itsenäisesti tarkistaa jokaisen paketin aitous ja eheys, ilman aiemmin muodostettua luottamussuhdetta paketin lähettäjään tai muihin solmuihin, jotka ovat käsitelleet pakettia. PLA on suunniteltu yhteensopivaksi nykyisen Internetin infrastruktuurin kanssa siten sallien vaiheittaisen PLA-ratkaisun käyttöönoton verkossa. Lisäksi PLA:ta voidaan käyttää samanaikaisesti muiden tietoturvaratkaisuiden kanssa, jotka tarjoavat päästä-päähän tietoturvaa, kuten IPsec ja Host Identity Protocol (HIP).

PLA:n alkuperäisenä suunnittelutavoitteena on ollut tuottaa ratkaisu, joka skaalautuu energia- ja laskentaresursseiltaan rajallisista kannettavista laitteista, kuten matkapuhelimista ja sensoreista, aina runkoverkoissa oleviin järeisiin reitittimiin, joissa siirtokapasiteetit ovat kymmeniä gigabittejä sekunnissa. Siten haasteena PLA:n suunnittelussa on ennen kaikkea digitaalisten allekirjoitusten laskennan optimoiminen sekä laskentatehon että energiakulutuksen suhteen. Tämä työ osoittaa, että PLA:ta varten kehitettyjen digitaalisten allekirjoitusmenetelmien ja niiden laitteistoläheisten toteutusten ansiosta PLA skaalautuu runkoverkkojen reitittimiin. Toisaalta, paketin eheyden tarkistamisen vaatima lisäenergian tarve on merkittävästi pienempi kuin langattoman viestinnän energiantarve. Näin ollen PLA soveltuu myös kannettaviin laitteisiin.

Tässä työssä tutkitaan myös mahdollisuuksia hyödyntää PLA:n arkkitehtuuria muihinkin käyttötarkoituksiin Internetin infrastruktuurin suojaamisen lisäksi.

Avainsanat
tietoverkkojen turvallisuus, palvelunestohyökkäykset, IP-protokolla, Internetin infrastruktuuri, digitaaliset allekirjoitusmenetelmät, elliptisten käyrien menetelmät

Table of Contents

List of Figures

Acknowledgements

Work on this thesis was funded by Tekes project “Securing IP-based network infrastructures using Packet Level Authentication (PLA) - technique” and was carried out in the Laboratory for Theoretical Computer Science of the Helsinki University of Technology.

Several people have helped greatly with writing this thesis. I want to thank my supervisor professor Kaisa Nyberg for supervising and helping with my thesis.

I would like to thank my instructor Hannu H. Kari. His suggestions, support, and comments were extremely valuable during my work in the PLA project and while writing this thesis.

I would also like to thank the inspector of my thesis, Arto Karila, from the Helsinki Institute for Information Technology (HIIT) for his helpful comments.

Finally, I would like to thank the whole Laboratory for Theoretical Computer Science for providing a great and supporting working environment.

Dmitrij Lagutin
Espoo, 26.05.2008

1. Introduction

Currently, the Internet is vulnerable to different kinds of malicious behaviour such as denial-of-service attacks, break-ins, phishing attacks, and unsolicited e-mail (spam). Such frequent attacks make it increasingly difficult for benevolent users to use the Internet effectively. The signal to noise ratio is basically becoming too low on the Internet. The reason for Internet's vulnerability to various attacks lies within its original design goals, which assumed that the network will be used in a very different way than it is used today. Originally, the Internet was designed to be used by a small number of benevolent parties, thus the possibility of an internal attack was mostly ignored in its design. Nowadays, the situation is very different: the Internet is used by a large number of different users and virtually all attacks against it are internal by nature. Protecting the Internet against internal attacks is very difficult, since effective security measures against such attacks do not exist. Traditional end-to-end security solutions like IPSec are ineffective if the network infrastructure is under attack and unable to deliver packets. Distributed denial-of-service attacks are especially dangerous and difficult to protect against.

The aim of this thesis is to describe a novel method for improving the security of the Internet by providing availability and protecting the network infrastructure and its users from various attacks. The main objective of the described solution is to give every node in the network[1] the ability to verify each packet independently utilizing public key cryptography and digital signature mechanisms. As a result, various attacks against the network and its users can be more easily detected and contained, before they can cause significant damage or disturbance. Thus, the network will be able to better fulfil its basic goal: to deliver packets of valid users in a reliable and timely manner in all situations. This thesis assumes that public key cryptography is feasible to use on a large scale because of new cryptographic algorithms and advances in semiconductor technology.

The thesis is organized as follows. Chapter 2 contains background discussion including the original design goals of the Internet, current security threats, and

1 In this thesis the network includes the network infrastructure and all other nodes located in the network.

solutions. It also contains a problem statement and describes a set of new design goals for the next generation Internet. Chapter 3 introduces Packet Level Authentication (PLA), presents its design goals, and describes its architecture. Cryptographic solutions used by PLA and software and hardware implementation of PLA are described in Chapter 4, which also contains an analysis of the performance impact of PLA. Chapter 5 analyses how PLA satisfies its design goals. Applications of PLA are discussed in Chapter 6. Chapter 7 contains discussion about PLA and also discusses future work. Finally, conclusions are presented in Chapter 8.

2. Background

This chapter covers the original design goals of the Internet and discusses current security threats and currently used security solutions on the Internet. The aim is to highlight situations where current security solutions are inadequate. This chapter also contains a problem statement and presents requirements for the secure next-generation Internet.

2.1 Original design goals of the Internet

The Internet was originally designed for military use during the cold war era. In 1967, the Advanced Research Project Agency (ARPA), part of the US Department of Defence, proposed plans for a packet switched network called ARPANET [50]. The proposal was approved and eventually ARPANET evolved into the Internet.

The original design goal [17] of the network which later became the Internet was to connect different existing networks together. Other secondary design goals are listed below in the order of importance:

1. The communication must continue despite the loss of networks or gateways
2. The network must support multiple types of communications services
3. The network architecture must accommodate a variety of networks
4. The network architecture must permit distributed management of its resources
5. The network architecture must be cost effective
6. The network architecture must permit host attachment with a low level of effort
7. The resources used in the network architecture must be accountable

Because the network was originally designed with military use in mind, the communication survivability was the top goal while accountability was at the bottom of the list. Had it been developed for commercial purposes, accountability would have had a much higher importance.

It is important to note that protection from internal threats or attacks was never mentioned in the list of design goals. Since the Internet was designed originally for

military use, it was assumed that a possible attacker would always attack the network from the outside and would only try to destroy or sabotage the network infrastructure. All nodes that were connected to the network were assumed to be well behaving nodes that always worked for the common good of the network and would never try to cause any intentional damage to the network. While the design goals explicitly mention that the Internet must continue to operate despite the loss of infrastructure, like gateways, they completely ignore the situation where the gateway, or some other node in the network, is controlled by the attacker and is used to attack the network from the inside.

The original design goals also contain other assumptions. There was not any need to handle privacy issues since the Internet was to be used only by a small benevolent group. Eavesdropping was not an issue because it was assumed that all communication would be transmitted using fixed lines that would be inaccessible by an external attacker. These assumptions are a major problem in the Internet's design.

Currently, the above mentioned assumptions, including the assumption that attacks always originate from the outside of the network, do not apply. The Internet is no longer solely used by military or government agencies, and virtually all attacks against the Internet originate from within the network itself, from nodes or routers that are attached to the Internet. It is very hard to defend against such attacks, since a mechanism to effectively handle internal attacks was not part of original design goals and therefore does not really exist.

There are also other issues which were ignored in original design goals. Eavesdropping is a major problem today, especially as wireless networks become more popular. Privacy issues on the Internet are also very important as the Internet's popularity is growing and the amount of personal data stored on the Internet is also increasing rapidly.

2.2 Current threats on the Internet

There exist several types of attacks that can be launched against Internet's users, servers, and infrastructure. In a denial-of-service (DoS) attack, the attacker disturbs

the victim in a such way that the victim is unable to continue normal operation. This can be accomplished using flooding: the attacker sends a large number of meaningless packets to the victim and these packets will use all or a majority of the victim's bandwidth or computing power, making it difficult or impossible for the victim to communicate with legitimate nodes. A DoS attack can also be accomplished by flooding the victim with legitimate requests. For example, the attacker could make thousands of requests for a web-page that is located on the victim's server. One popular variant of a such attack is a distributed denial-of-service attack (DDoS), where a large quantity of nodes simultaneously attack a single victim. Because such attacks may originate from thousands of nodes that are located in different parts of the Internet, defending against such attacks is currently very difficult.

Denial-of-service attacks are usually launched against WWW servers, but there have been cases of DDoS attacks against DNS root servers. Statistics of the Finnish national Computer Emergency Response Team (CERT-FI) [15] show that the number of DoS attacks is growing rapidly. During the first half of 2007, the number of reported DoS attacks in Finland was more than twice as high as in the whole year of 2006. Furthermore, in the first half of 2007, denial of service attacks were more popular than traditional break-ins.

A good example of a serious DDoS attack is an attack launched in Estonia in May 2007 [38]. This attack was launched against several government web sites, including the sites of different ministries. Overall, the attack consisted of 128 unique DDoS attacks that lasted from less than a minute to over ten hours. This attack highlighted two major problems that exist on the current Internet: the inability to react quickly to an attack and the inability to catch the perpetrators. The authorities were powerless to stop the attack which lasted more than one week overall. At the time this thesis was written, over one year had passed since these attacks and the culprits had yet to be caught.

In spoofing attacks, forged packets are used to attack the network. There exist different kinds of spoofing attacks. The Address Resolution Protocol (ARP) is used on the network layer to resolve node's hardware address based on the network layer

address like an IP address. In the ARP spoofing attack [5], the attacker sends fake ARP reply messages containing incorrect hardware address/network address mappings. The attacker can use ARP spoofing, e.g., to map his own hardware address to the gateway's network address, and thus a victim will send his packets to the attacker instead of the proper gateway. This would allow the attacker to launch a man in the middle attack or simply cut all communications to the victim altogether.

TCP (Transmission Control Protocol) is the most commonly used transport layer protocol to transport data on the Internet. There are several TCP related attacks such as TCP reset attacks and TCP SYN flooding attacks. In the TCP reset attack [49], the attacker attempts to terminate established TCP connections by sending spoofed TCP packets. For example, if nodes A and B have established a TCP connection between themselves, attacker C can send B a packet with the TCP RST bit set, A's IP address, and a correct TCP sequence number. Upon receiving packet containing the set RST bit, B terminates the TCP connection and the attacker is therefore able to deny service between A and B. In order to carry out such an attack, the attacker must guess a correct TCP sequence number. Even though the TCP sequence number is a 32-bit integer which can have approximately four billion different values, a TCP reset attack is relatively easy to accomplish in practice since TCP uses a transmission window mechanism to send data. The size of the transmission window determines how much data can be sent simultaneously and TCP accepts out-of-order packets as long as their sequence numbers are within a valid transmission window. Basically, this means that in a TCP reset attack, the attacker does not need to try every possible sequence number: it is enough to send one packet in every possible transmission window range. This makes the TCP reset attack much more dangerous. With a commonly used window size of 16384 packets, it is enough for an attacker to send about 260 thousand packets (instead of four billion) to successfully terminate a connection. A larger transmission window size decreases the required number of packets and makes a TCP reset attack even more easier to carry out. Furthermore, if the attacker can eavesdrop the communication, he can discover a currently used sequence number, in which case the attack is trivial to launch.

A SYN flooding attack [44] is a form of denial-of-service attack where the attacker creates a large number of partially opened TCP connection to the victim, draining the

victim of computational resources such as memory. The TCP protocol uses a three way handshake to establish the connection: the initiator first sends a SYN message, the recipient replies using a SYN-ACK message, and finally the initiator sends an ACK message. The recipient also allocates resources to handle the connection after receiving the initial SYN message. These resources are freed after the recipient receives an ACK message and the connection is established. In a SYN flooding attack, the attacker sends a large amount of SYN messages using spoofed source addresses. This causes the victim to allocate resources to handle these connection requests, and because the source addresses are spoofed, the victim will never receive ACK messages and subsequently runs out of resources.

In a replay attack [45], the attacker captures valid communication between victims and replays it at a later stage. In the simplest form of a replay attack, the attacker simply creates several copies of a valid packet and sends the copies to a valid destination. In such an attack the attacker can modify a TCP sequence number field of copied packets in order to make it harder for firewalls to detect the duplicate packets. While the attacker will not be able to gain unauthorized access with such attack, the duplicated packets will unnecessarily consume resources in the network. This can cause significant damage in wireless networks where resources like bandwidth and battery power are often scarce. Usually, the aim of a replay attack is to gain unauthorized access by replaying valid packets. In a classic replay attack, the attacker intercepts a message exchange and replays it fully at a later stage. For example, the attacker could intercept packets containing a password exchange between a client and a server and then gain access to the server by resending intercepted packets. In more complex replay attacks, the attacker replays some part of a message exchange (which can be an exchange of cryptographic keys) simultaneously with an actual message exchange to gain unauthorized access or impersonate a victim. Usage of timestamps or one-time session tokens during message exchange offers protection against replay attacks. Such measures aim to guarantee that valid messages can be sent only once.

The Domain Name System (DNS) is used, among other things, to determine the IP address of a node based on its hostname. In a typical case, the client sends a DNS request containing an unknown hostname to the DNS server, and the server replies

with the corresponding IP address using a DNS reply message. In the DNS cache poisoning attack [6], an attacker feeds incorrect hostname/IP address mappings to the DNS server. For example, the attacker could map a legitimate site like a website of a bank to his own IP address. As a result, clients wishing to access the bank's website will access the attacker's own server instead.

There are also other ways to attack the DNS. DNS requests contain a 16-bit random ID number and valid reply messages must also include this same ID number. In a DNS spoofing attack, the attacker fakes a reply from a valid DNS server, either by sniffing a correct ID number from DNS request message or by guessing it. The result of the attack is the same as in the case of cache poisoning; the victim will receive an incorrect IP address for his DNS request. The DNS is vulnerable to spoofing attacks because currently DNS messages are not authenticated or protected by any means. Thus, the recipient of a DNS reply message can not guarantee that the received message is authentic. There exist a DNS Security Extensions (DNSSEC) [4] mechanism to protect DNS messages using cryptographic signatures and other means. However, DNSSEC is not yet widely used on the Internet.

Phishing attacks differ from other attacks mentioned previously because they target users directly instead of targeting the network infrastructure or protocols. The aim of a phishing attack is to lure users to voluntarily disclose confidential information like passwords. For example, the attacker can contact a victim by phone, pretend to be a system administrator from the victim's place of work, and ask for a victim's password. On the Internet, one commonly used phishing approach is to create a look-a-like web page of an on-line bank and send its URL by e-mail to a large number of people in order to lure victims to reveal their bank account usernames and passwords. The amount of phishing cases has significantly increased in the last few years and in July 2007 alone almost 31000 phishing websites were reported [3]. Phishing attacks are often quite effective since it is currently difficult for an ordinary user to guarantee the authenticity of a web page or the real sender of an e-mail message.

2.3 Current security solutions in the Internet

There are three main principles of information security which various security solutions aim to provide: confidentiality, integrity, and availability. The aim of confidentiality is to guarantee that only authorized parties can access data; confidentiality is usually accomplished via encryption. Integrity ensures that data is authentic and has not been tampered, Availability aims to guarantee that data and services are available for legitimate users in all possible situations.

The Internet architecture can be divided into several layers according to the TCP/IP model; the application layer, transport layer, internet layer, and the network access layer as shown in Figure 1, the figure also contains a comparison to the Open Systems Interconnection (OSI) model.

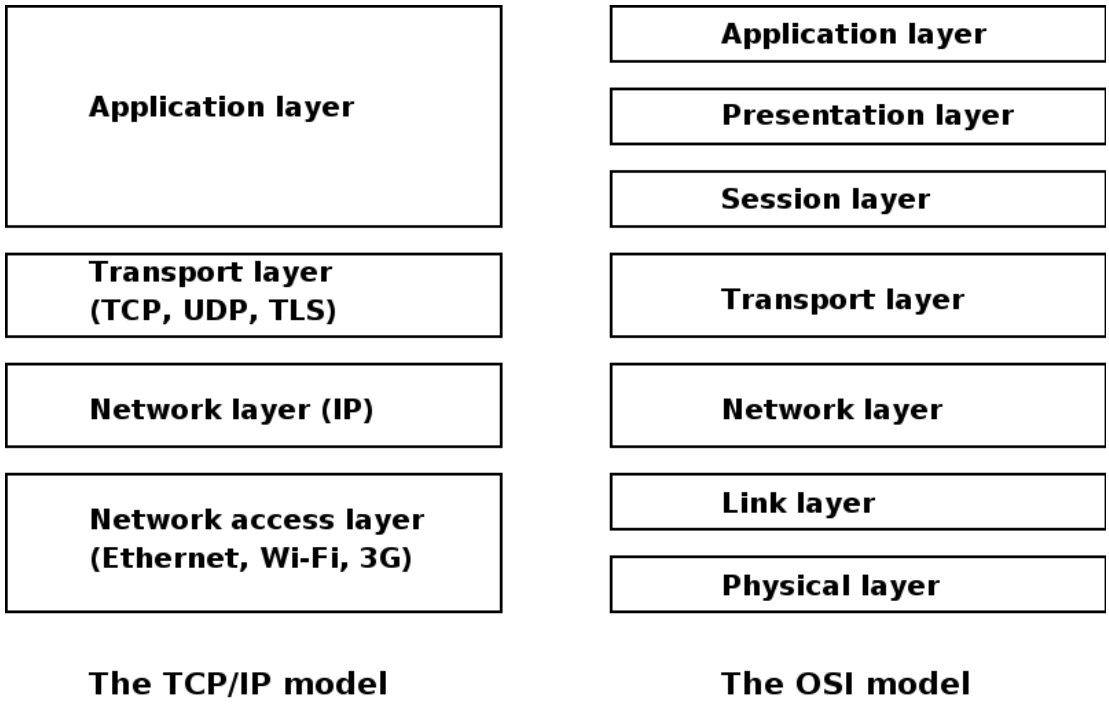


Figure 1. The Internet architecture according to the TCP/IP model and comparison to the OSI model

The network access layer of the TCP/IP model contains the physical and link layers from the OSI model while the application layer of the TCP/IP model includes application, presentation, and session layers from the OSI model. Because the aim of this thesis is to concentrate on network level security, security solutions from the

application layer of the TCP/IP model are not discussed here. Security solutions can also be classified based on their applicability as shown in Figure 2.

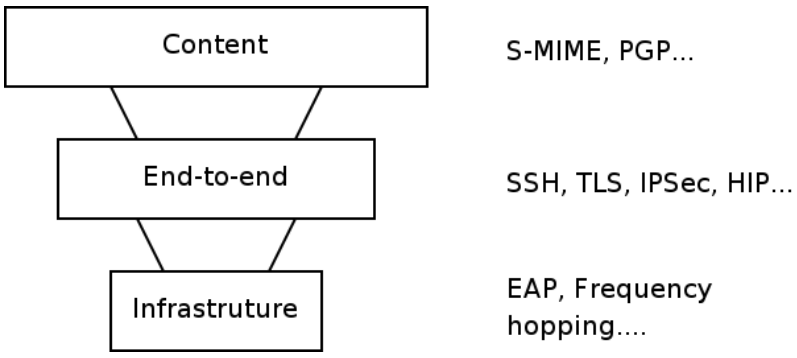


Figure 2. Security solutions divided into content level, end-to-end level and infrastructure level solutions

In this figure, content level solutions refer to security solutions which work on the application layer in the TCP/IP model and aim to secure specific content, such as e-mail messages. End-to-end solutions usually work on the application and network layers and aim to protect all content which is transmitted between two end points. Infrastructure level solutions aim to secure the underlying infrastructure on the link and physical layer against threats such as jamming or eavesdropping. Existing security solutions for different layers of Internet are explained in more detail below.

The Transport Layer Security (TLS) [20] protocol and its predecessor, Secure Socket Layer (SSL), are end-to-end security solutions which aim to provide secure communication on the transport layer. They provide authentication and confidentiality using encryption and are widely used to secure web browsing and instant messaging communications. In a case of web browsing, usually only the server is authenticated. During the establishment of such TLS connection, the server provides the client a certificate that is signed by some trusted certificate authority (CA); the client may optionally contact the certificate authority to verify the certificate's validity. The client and the server also agree on a cryptographic cipher and hash function, and generate key material for encryption and decryption during the establishment phase. Since TLS works on the presentation layer, it does not protect traffic from attacks that occur on lower layers.

The aim of the IP Security Architecture (IPSec) [29] protocol is to improve the security of IPv4 and IPv6 protocols by providing integrity and confidentiality on the network layer. IPSec works by extending a standard IP header, it uses Authentication Header (AH) and Encapsulating Security Payload (ESP) protocols to accomplish its goals. The AH includes a sequence number, Security Parameter Index (SPI), and authentication data containing a hash over the packet. The main task of the AH is to guarantee the integrity of the packet, including protection against replay attacks, and data origin authentication. The main aim of ESP is to provide confidentiality by encrypting data. IPSec provides end-to-end security, allowing two hosts to establish a secure, encrypted IPSec connection which protects the traffic between them. IPSec is widely used protocol to secure virtual private networks (VPN) [22]. One major problem with IPSec and other traditional security solutions is that they concentrate on providing end-to-end security but they cannot protect the underlying network infrastructure. If the packet protected by IPSec or a similar solution has been modified, duplicated, or delayed, only the end point of the connection can detect this, and intermediate nodes will continue to forward these invalid packets. As a result, such packets will unnecessarily consume network resources. In addition, such end-to-end security solutions are useless if the underlying network infrastructure is attacked and is unable to deliver packets to the destination as a result of, e.g., a DoS attack.

The Host Identity Protocol (HIP) [36] aims to provide confidentiality, better support for mobility, and support for multihoming. Traditionally, an IP address determines both the topological location of a host in the network and its identity. This can cause problems in case of mobility where the host receives a new IP address upon changing networks. To solve this problem, HIP introduces host identifiers (HIs) to describe the identity of the host, therefore the IP address is only used for describing the topological location of the host. Under HIP, connections are initiated to host identifiers instead of IP addresses and the host identifier is actually a hash over the host's public key, HIP also supports interoperability between IPv4 and IPv6. HIP provides confidentiality by encrypting all data traffic using the host's private key and the ESP protocol.

HIP uses a 4-way Diffie-Hellman key exchange protocol (also known as a base exchange) to authenticate hosts before establishing a connection. During the base

exchange, the recipient of the connection presents a puzzle to the initiator and the initiator must solve this puzzle before the connection can be established. The idea behind the puzzle is to require much more computational power from the initiator than from the recipient. This provides some protection for the recipient of the connection against denial-of-service attacks, since a single recipient will be able to handle a high amount of base exchange requests from different initiators. If the recipient receives a large amount of base exchange requests, it may increase the difficulty of the puzzle, therefore limiting the amount of requests initiators can make. However, such puzzle mechanism can be abused in certain situations to launch DoS attacks against peers that are communicating with the victim using HIP. Suppose a situation where several devices with very low computational power are communicating with a certain HIP server. If an attacker starts flooding the HIP server with connection requests, the server will increase the difficulty of its base exchange puzzle, and those devices with very limited resources will be unable to solve the puzzle quickly enough to continue normal operation. HIP also posseses other security related disadvantages. While HIP offers some protection for the recipient during the establishment phase, it does not protect a recipient or underlying network after the connection has been established. After an attacker has discovered a victim's IP address, the attacker can freely launch, for example, a DoS attack against the victim.

An IPv6 address consists of two parts: a 64-bit subnet prefix and a 64-bit interface identifier. Cryptographically Generated Addresses (CGA) [7] offer a method for generating the interface identifier part of an IPv6 address by hashing a sender's public key. CGA requires that a sender's public key is included in sent packets and the packet is somehow signed with sender's private key. The main aim of CGA is to offer protection against IP address spoofing; every other node in the network can verify whether a hash of the public key matches with an interface identifier. In order to spoof a valid CGA address, the attacker would need to generate a very high amount of public keys to find a hash collision which is a very resource intensive operation.

Link layer security solutions aim to improve security by providing confidentiality and authentication on the link layer. They are mostly used in wireless networks, because wireless networks are very vulnerable to eavesdropping and are easy to attach to. Wired Equivalent Privacy (WEP) was the original solution to enhance the security of

IEEE 802.11 wireless networks. However, WEP is a poor security solution for several reasons [13]. First, WEP uses shared secret keys for encrypting traffic and these keys are often stored in an insecure way on the device. In addition, WEP also lacks a key management protocol making it difficult to distribute new keys to all nodes. The biggest drawback of WEP is that its encryption can be easily broken by capturing enough encrypted packets. Several methods to break WEP encryption have been published and the fastest method [47] can discover the encryption key in less than one minute. To overcome these limitations, WEP has been largely replaced by Wi-Fi Protected Access (WPA) and WPA2 technologies. WPA and WPA2 offer significantly better encryption and also support the Extensible Authentication Protocol (EAP) [1] for user authentication. EAP is a more secure and scalable way to manage authentication as compared to using pre-shared secret keys.

Frequency hopping [21] is a mechanism that is often used in military environments to protect wireless networks against jamming and to a lesser extent against eavesdropping. When frequency hopping is employed, the transmitter and receiver change their communication frequency rapidly across a wide frequency range. Since the attacker most likely has limited resources for jamming, the attacker will only be able to jam a small portion of the used frequency range. Thus, most communications will go through despite the jamming. In order for frequency hopping to be effective, the mechanism employed to select new frequencies must not be known by the attacker.

Link layer security solutions offer protection from external attacks, although they are not effective against internal attacks. For example, if the attacker can compromise the node that is communicating using pre-shared keys with a IEEE 802.11 base station, the attacker will gain access to relevant encryption keys and will be able to listen to traffic between other clients and this base station. A similar problem exists when frequency hopping is used; if the attacker compromises one node, the attacker is likely to be able to deduce the method for selecting new frequencies and will thus be able to jam or eavesdrop the communication in the rest of the network.

Overall, existing security solutions have a two main drawbacks. First, they do not protect the end user from receiving “garbage” traffic from the Internet. Second, they

do not provide availability and do not effectively protect the network infrastructure from attacks, especially if these attacks occur within the network itself. For example, a traditional principle to protect the network against a denial of service attack is to have more resources available compared to the attacker. Naturally, this principle will not work against a distributed denial of service attack where the attacker can control thousands of nodes. Frequency of various attacks has also led to overuse of firewalls on the Internet. Firewall rules are often so strict that they block also a valid traffic making the Internet less usable for benevolent users. This thesis concentrates on latter problem, the protection of the network infrastructure by providing availability using novel methods.

2.3.1 An example of a real life security solution

Because there exist different kinds of threats on the Internet, mission critical services, like Internet banking, have to be protected on several levels using different security solutions. A proposal for a secure Internet banking solution was discussed in [42]. The proposal includes duplicated infrastructure and several layers of security to secure the service. In the proposal, the computing system of the bank is duplicated on three levels. First there exist two copies of the bank's computing centre located in different geographical locations. This is necessary to protect the system from physical attacks and natural disasters. Secondly, ISP connections are duplicated, there are two ISPs which provide services to both computing centres. Therefore, even if one ISP goes off-line, the service is not interrupted. Thirdly, each computing centre has at least two sets of identical hardware (firewalls, routers, servers, mainframes) in order to cope for hardware failures. An overview of the proposal is shown in Figure 3 below.

At the top layer there are front end routers that are connected to both ISPs. These routers forward traffic to IPS (Intrusion Prevention Systems) systems that reside in the front of firewalls. The aim of these systems is to stop trivial flooding attacks before they can cause damage to other components. On the fourth layer there are separate servers to handle SSL decryption. Since SSL encryption and decryption are computationally intensive operations, it is better to handle these operations in dedicated servers. Otherwise the attacker could flood mission critical servers with a

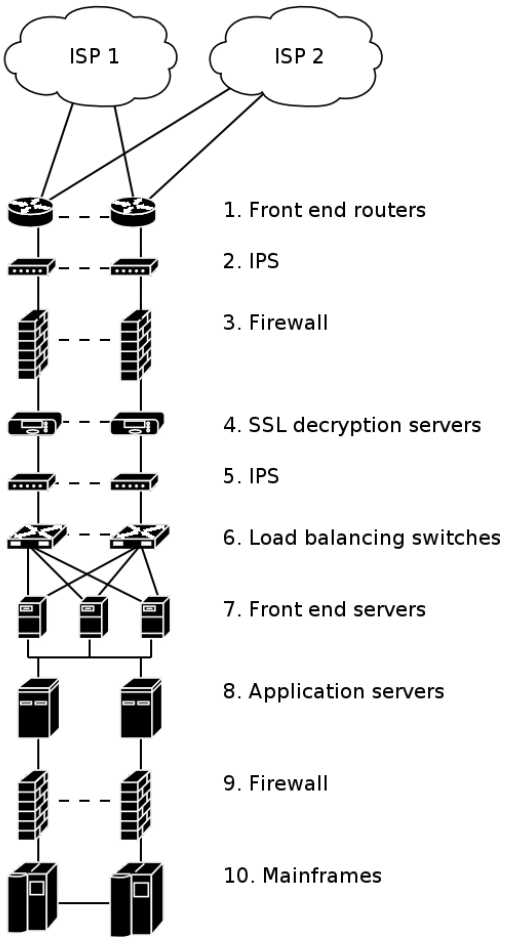


Figure 3. An example of security solution for Internet banking [42]

high amount of SSL connection requests. There is another set of IPS systems in front of load balancing switches. Load balancing switches forward traffic to an available front end server which is connected to the actual application servers. Finally, there is another set of firewalls in front of the mainframes that contain the critical data.

Such a security solution is very complex, but this complexity is necessary to ensure good protection against different kinds of attacks. Duplication of servers means that a single server failure will not interrupt operations, while having multiple security layers adds protection for servers that are actually handling critical transactions.

2.4 Problem statement

The aim objective of this thesis is to develop a new method for securing the network infrastructure and services for benevolent users of the network. The aim is to improve security by providing availability for the next-generation Internet and to protect it against various security threats, like denial-of-service attacks, which are currently present on the Internet. If good availability can be achieved, then end-to-end security solutions can effectively protect against other threats by offering data integrity and confidentiality.

Many assumptions that were part of the original Internet design, such as the scarcity of resources, are not valid any more. The amount of available Internet bandwidth is growing rapidly and this growth actually surpasses the growth of Internet traffic. According to the Global Internet Geography report [46], growth of Internet's average traffic in 2007 was 57%, while the growth of the available bandwidth was 68%. In addition, public key cryptography is relatively less computationally intensive because of new, more efficient cryptographic algorithms, rapid advances in a semiconductor technology, and increases in processing power. Therefore, this thesis assumes that it is feasible for a security solution to consume some bandwidth and computational power.

Using modern public key cryptography algorithms, it is possible to build a system where every transmitted packet in the network carries a digital signature and has an indisputable owner. Such a system would protect the network from different kinds of attacks and would make it much easier to catch attackers. This thesis aims to prove that a such solution is feasible and scalable for small portable mobile devices and high-speed Internet core network links with speeds of 10 Gbps[1] and above.

2.5 Redesigning the Internet

To overcome the problems of the current Internet, we propose several new requirements for the next-generation Internet which take into account current security threats and issues. The fundamental requirement is that the network should be able to

1 10 Gbps speed is the fastest commonly used transmission speed today of single interface. Faster routers usually use several 10 Gbps network interfaces in parallel.

fulfil its basic goal: to deliver packets of valid users in reliable and timely manner in all situations. Other requirements for the next-generation Internet are listed below. The network should support these requirements, although applied security policies will determine how strictly these requirements will be enforced in various situations.

Only valid packets are forwarded in the network

Since the Internet is currently very vulnerable to internal attacks, it is very important to detect and react to those attacks as soon as possible. Thus, only valid verified packets should be allowed to be transmitted in the network. Invalid packets that have been modified, delayed, or duplicated are considered malicious and should be discarded by the first possible entity of the network before they can cause damage or unnecessarily consume network resources elsewhere.

Every packet has an owner and all packets originate from trusted entities

There should be a way to establish the owner of every packet that is sent on the network. This is an important requirement in order to limit attacks and effectively remove entities behind those attacks from the network. In addition, every entity that sends data to the network must be authorized by a some authority which is responsible for managing the network. It is important to provide such traceability in order to make it easier to catch attackers.

Misbehaving nodes should be removed quickly from the network

A benevolent node may become dangerous for the network for several reasons. For example, the node could be hijacked by a malicious party or its software could malfunction. Therefore, there should be an effective way to quickly remove misbehaving nodes from the network before they can cause a significant damage.

Prioritizing traffic

In case of emergency, the network's bandwidth may become very limited. Thus, there should be a way to prioritize traffic in order to make sure that a really critical traffic will get through in all situations.

Manageability

The whole network cannot be managed by a single entity. Thus, there should be a way for different operators and authorities to effectively manage different parts of the network without requiring centralized control. There should also be an easy and flexible way to add new nodes to the network. This requirement resembles the fourth requirement of the original Internet but covers a wider scale. The network may be very dynamic containing a large amount of nodes which are constantly leaving and entering the network.

Controlling incoming connections

In the current Internet architecture the initiator of the connection[1] is completely in control of the connection. The initiator of the connection can decide whom he will contact and when a connection is made. However, such a policy presents many problems. The recipient of the connection might be using a wireless access network with a limited bandwidth, and the recipient might even have to pay for all incoming traffic. In addition, the recipient might be in a situation where he does not want to be disturbed by unnecessary connections, while at the same time, the recipient may want to receive very important connections from specific initiators.

Privacy protection

The traceability requirement mentioned above does not mean that users should completely give up their privacy. For example, users must have a way to create pseudonyms in such way that they can maintain their anonymity to the network in normal situations. Basically, the user should not be forced to disclose unnecessary information to other parties in the network. For example, it is not necessary to disclose a real identity to participate in an online discussion. However, if the user misbehaves, authorities should have a way to determine the real identity behind the user's pseudonym.

1 In this context, the term connection denotes the situation where the initiator is sending data to the recipient over the network by any means possible.

In addition to the above mentioned requirements, we propose other criteria which are more closely related to network services instead of the network infrastructure and will not be covered in detail by this thesis.

Digital signing of all information on content level

All information should be cryptographically signed on the content level, just as every packet is signed by the sender at the network layer. The user must have means to check that various documents like a spreadsheet, web-page or a presentation have not been modified and have been created by verifiable parties.

Dynamic and adaptive applications

Future devices, especially mobile devices, will be able to use in parallel several network interfaces with different capabilities such as Wi-Fi, 3G. or fixed Ethernet. Applications should be able to adapt to situations where the network's capabilities like bandwidth and latency change rapidly. They should also utilize available resources of the network in most efficient way and even function without network connectivity for a while.

3. Packet Level Authentication (PLA)

In order to solve the aforementioned security problems, we introduce Packet Level Authentication (PLA) [14]. The main aim of PLA is to enhance network security by providing availability and protecting the network from several kinds of attacks, like denial-of-service attacks. The main principle is that benevolent traffic should go through while malicious traffic should be detected and stopped as quickly as possible. The major difference between traditional security solutions and PLA is that PLA gives ability for nodes in the network to detect attacks immediately by checking the authenticity and integrity of every packet. In comparison, when traditional end-to-end security solutions like IPSec are used, only the end point of the connection can verify the authenticity of the packet. Unlike traditional link level solutions, PLA allows every node to verify the packet independently without having to trust nodes that have previously handled the packet. It is important to note that PLA aims to complement existing security solutions instead of completely replacing them.

The security measures in Packet Level Authentication resemble those present in paper currency. Anyone can verify whether or not a paper bill is authentic without having to contact the bank that issued the bill. It is enough to verify various security measures inside the bill, such as its watermark, a metal strip, or a hologram. The same principle applies to PLA. When PLA is used, any node in the network can verify the authenticity and the integrity of every packet without having any kind of contact with the sender of the packet because PLA includes in every packet all the necessary data to carry out such verification. Such a system has a significant advantage compared with traditional security solutions that concentrate on providing end-to-end security. Because PLA allows various attacks to be immediately detected, the network can take countermeasures against them in a more effective way, before attacks can cause a significant amount of damage. To accomplish its goals, PLA utilizes public key cryptography. The public key cryptography is very computationally intensive, but it can be used with a sufficient performance as long as dedicated hardware is used to handle cryptographic operations.

This chapter is organized as follows: Section 3.1 discusses design criteria behind PLA and an overview of the PLA architecture is presented in Section 3.2. Section 3.3 introduces the PLA header structure while Section 3.4 describes trusted third parties in the context of PLA. The bootstrapping of a new node is discussed in Section 3.5 while Section 3.6 describes how the authenticity of the packet is verified using the information contained in the PLA header.

3.1 PLA design criteria

The design criteria of PLA are listed below. They are classified as either mandatory criteria, that are essential, important criteria that are important to have, but not essential, or optional criteria that are less important but are still useful.

Mandatory

Compatibility with the existing Internet. The system shall work with existing IP networks without requiring any major changes to the network. The system shall also be compatible with existing security solutions like IPSec.

Deployability. The system must be easy to deploy on a wide scale. It shall be possible to easily add more nodes to the system and the system must also work without any additional security association setup between nodes.

Misbehaving nodes should be removed quickly from the network. A benevolent node can become hostile for the network for several reasons. For example, the node could be hijacked by a malicious party, or the node's software could malfunction. Therefore, there shall be a way to remove misbehaving nodes from the network effectively and quickly before they can cause significant damage to the network. This requirement was also present in Section 2.5.

Validation of packets. Every node in the network must be able to validate the authenticity of every packet without prior communication with the sender of the packet. It should be possible for any node to detect if a packet has been modified, duplicated, or delayed.

Two last validation requirements are important to protect the network from replay attacks which use duplicated or delayed packets.

Important

Scalability. The system should be scalable from small wireless ad-hoc networks to large networks on the Internet scale. The system should also be usable with small and portable devices. This requirement is not mandatory, since the system would still be useful in certain situations, such as in mission critical networks, even if this requirement is not satisfied.

Optional

Small power and bandwidth overhead. It is preferable for the system not to introduce significant bandwidth overhead and not consume high amounts of power. This requirement is especially important for mobile networks and small mobile devices, because such networks and devices are usually bandwidth and energy constrained.

Free of patents. The system should not use patented technologies.

It is important to note that the PLA is not designed to provide the confidentiality of end-to-end communication. Other security solutions such as IPSec and HIP can be used together with PLA to provide confidentiality.

3.2 Overview of the PLA architecture

The basic idea behind PLA is to ensure the authenticity of packets by using a public key cryptographic system to sign every packet sent over the network. When public key cryptography is used, only the holder of the private key can sign the packet, but every party can verify the authenticity of the packet using the packet's signature and the sender's public key. PLA accomplishes its goals by adding its own header to every IP packet, which contains all necessary information for verification of the packet. An overview of the PLA architecture is presented in Figure 4.

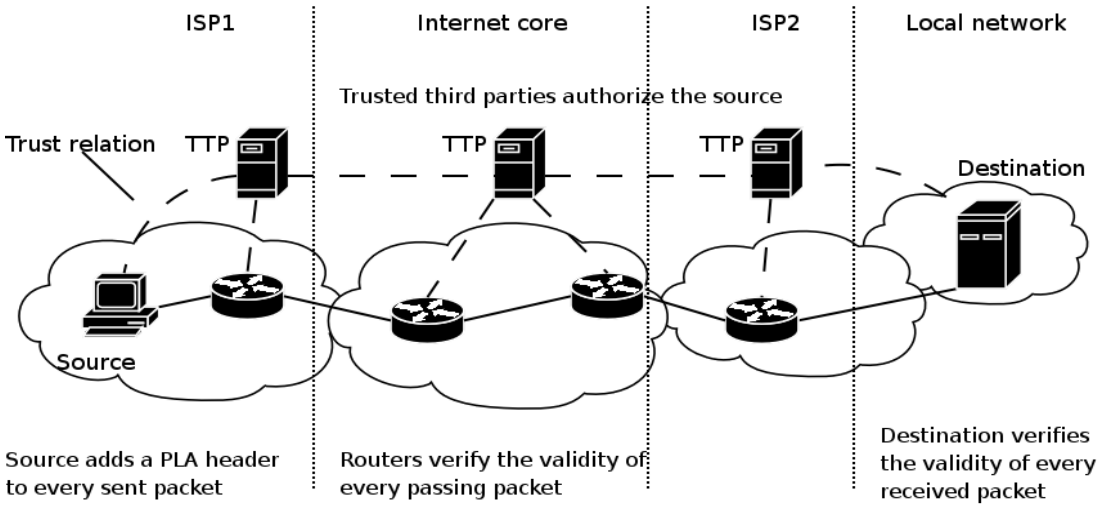


Figure 4. An example of usage configuration of the PLA based network architecture

In a simplified case, the PLA architecture consists of four major elements: a source and destination that are communicating with each other, routers between them and of trusted third parties (TTPs). Basically, PLA adds trusted third parties and the ability to verify the PLA header information at every node to a plain Internet architecture. The example figure is vertically divided into four networks; the network of the source, the Internet core network, the network of destination's operator, and the destination's local network. Solid lines denote the actual data connection while dashed lines denote trust relations between different entities.

The basic operation of PLA is as follows. The source that sends a packet adds a PLA header to each sent packet. This header is added just after the IP header using a standard IP header extension mechanism [19]. As the packet travels through the network, all routers that handle the packet verify the packet's validity using information from the PLA header[1]. If this validity check fails, the packet is discarded immediately. Finally, the packet arrives at the destination which will also perform the validity check.

In addition, there exist entities called trusted third parties which will authorize the sender. In this example, it is assumed that such authorization between the TTP and sender has already been carried out. In the scope of PLA, a TTP is responsible for the tasks of a certificate authority (CA) and registration authority (RA) from a traditional

1 In case the router does not understand a PLA header, it routes packets simply based on IP header information. This enables a gradual deployment of PLA.

public key infrastructure architecture [16]. While TTPs are not necessary for checking the validity of the packet, they add another layer of security. As the packet travels through the network, intermediate routers and the destination can contact trusted third parties to verify that the sender of the packet is a valid and trusted entity in the network.

The position of PLA in the TCP/IP model is shown in Figure 5 below.

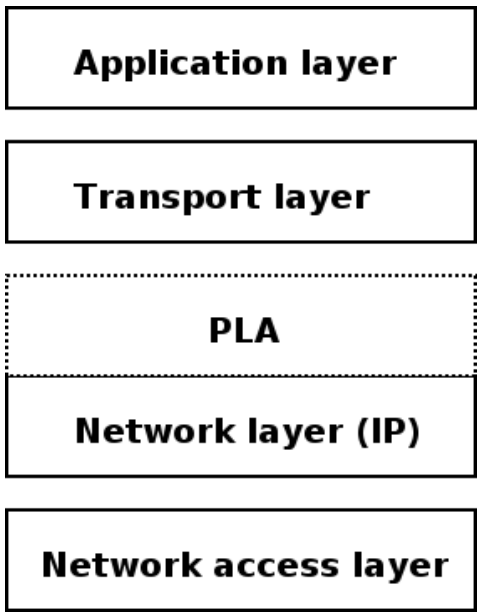


Figure 5. Position of PLA in the TPC/IP model

PLA is completely transparent to higher layers, thus PLA is able to work along with any current or future higher layer protocols like TCP, UDP, and HIP. In this figure PLA is positioned on top of a network layer as the PLA header is added just after of an IP header. However, PLA is not dependent of used network layer protocol, thus it could be argued that PLA can also be positioned below the network layer. However, such a case would require that every router in the network would support PLA.

3.3 PLA header

Figure 6 describes an example how the addition of the PLA header affects the structure of a normal IP packet which also utilizes IPSec.

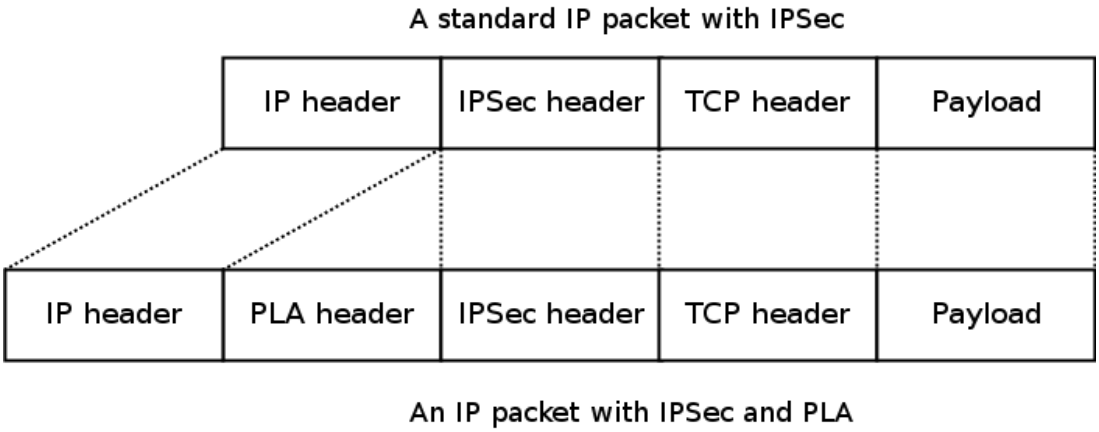


Figure 6. A comparison between a standard and PLA-enabled IP packet

In this figure, "IP header" refers to a standard IPv6 header containing source and destination addresses and ports among other fields as described in RFC 2460 [19]. In an IP header extension mechanism, the newest extension header is always added immediately after the IP header. In order to protect a whole packet, the PLA header must be added last, and thus it is positioned in the packet directly after the IP header before any additional extension headers like an IPSec header. Finally, the example packet contains a standard TCP header and a payload. The aim of this figure is also to show that PLA does not affect any higher level protocols.

An overview of the PLA header structure is shown in Figure 7 below. More detailed specifications of the header can be found in Appendix D.

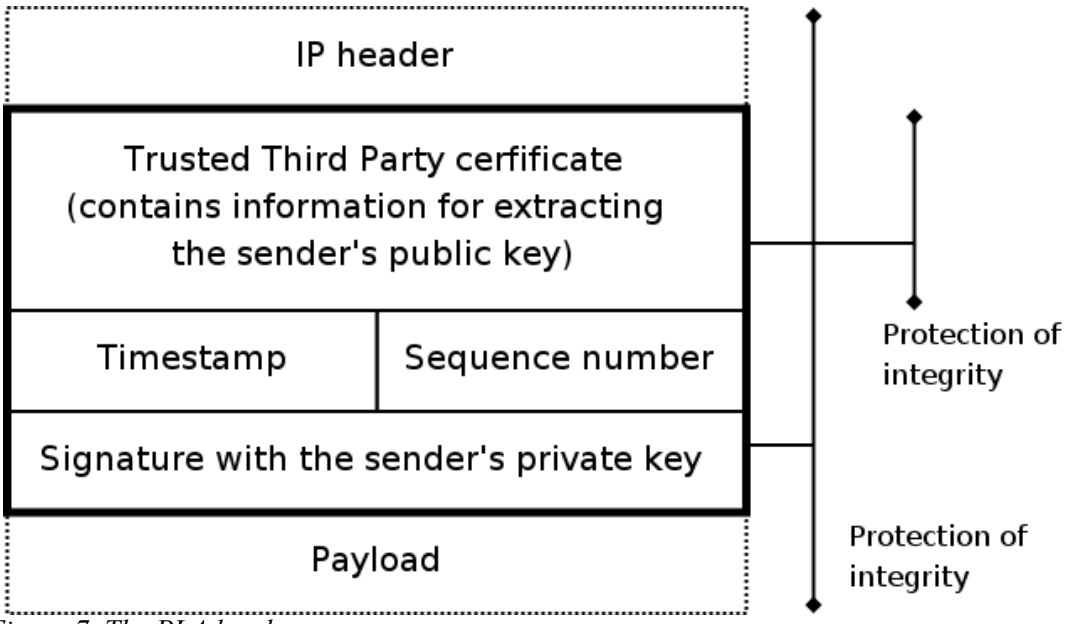


Figure 7. The PLA header

The actual PLA header is marked as a bold box in Figure 7 and it contains following fields.

Trusted Third Party certificate

Usage: This is a certificate from the TTP to the sender. Its aim is to corroborate the binding between the sender's identity and its public key. It is also used to guarantee that the sender is a valid, trusted entity that is authorized by some trusted third party. A TTP certificate is described in more detail in the next section.

To reduce computational and bandwidth overhead, PLA uses identity-based implicitly-certified keys which are described in more detail in Section 4.1. Thus, the sender's public key is not explicitly present, but it can be calculated using information present in the TTP certificate. The sender's public key, together with a signature, protects the integrity of packet and guarantees that any modifications of the packet will be detected, it also guarantees that the sender cannot deny sending a packet.

Example: A TTP can be an operator or a state authority.

Timestamp

Usage: The aim of the timestamp field is to detect packets that have been significantly delayed. Delayed packets can be a sign of a replay attack.

Sequence number

Usage: Monotonically increasing sequence number makes it possible to detect duplicated packets. It can also be used for per packet billing purposes.

Signature

Usage: The packet is cryptographically signed by the sender with the sender's private key. The signature guarantees the integrity of packet; any future modification of the packet will be detected because such modification would break the signature. Since the signature is also calculated over the PLA header, the attacker will not be able to modify other fields in the PLA header like the

timestamp or sequence number. PLA uses elliptic curve cryptography (ECC) [31][35] for cryptographic operations because ECC offers good security with small key sizes.

The signature calculation ignores some fields in an IP header, like the hop limit field, since that field can change during the lifetime of the packet. The PLA packet is considered fully valid if all fields in the PLA header are in order. The signature must be correct, the TTP certificate must be correct and issued by a valid trusted third party, the sequence number must be monotonically increasing number, and the timestamp should be recent enough according to used security policy.

3.4 Trusted Third Parties

Simply including a sender's public key with signature in a PLA header is not enough. An attacker could generate a large amount of different public keys for itself and launch an attack using thousands or even millions of different keys. Therefore, even if only a small amount of packets are sent using a single public key, the attacker could paralyse its victim or the network by flooding. To protect the network infrastructure from such attacks, separate Trusted Third Parties (TTPs) are required.

In the scope of PLA, a Trusted Third Party (TTP) is an entity that provides a binding between the user's identity and its public key and authorizes users who want to communicate using PLA by granting them certificates[1]. The TTP can be, for example, an operator, in which case the TTP would grant certificates to valid users of this operator, or a state authority, which would grant certificates to its citizens. The TTP certificate is included in the PLA header and it can be viewed as proof that the node is a well behaving entity which can be trusted. The TTP certificate has a limited validity time after which it must be renewed. If the user has been misbehaving, then its TTP certificate will simply not be renewed and the user will not be able to communicate using PLA. A TTP certificate information is utilized when validating packets going through the network, the packet must have a valid TTP certificate from a trusted TTP in order for it to be considered fully valid.

1 In the scope of PLA, TTP certificates contain rights and are similar to standard authorization certificates.

An overview of the TTP certificate format is shown in Figure 8. More detailed certificate specifications are presented in Appendix A using the S-expressions [43] format. To save bandwidth, two types of TTP certificates are used: a full certificate, and a partial certificate which omits a TTP's public key and locator. The idea is that nodes which are handling packets cache information about the TTP public key and locator, and thus the sender does not need to include those fields in each header. This makes packet handling in routers a bit more complex, but it also decreases bandwidth overhead of PLA. In order to make it possible for routers to validate the packet, the sender must naturally include a full TTP certificate in the first packet of the connection and occasionally send packets with a full TTP certificate to refresh routers' caches.

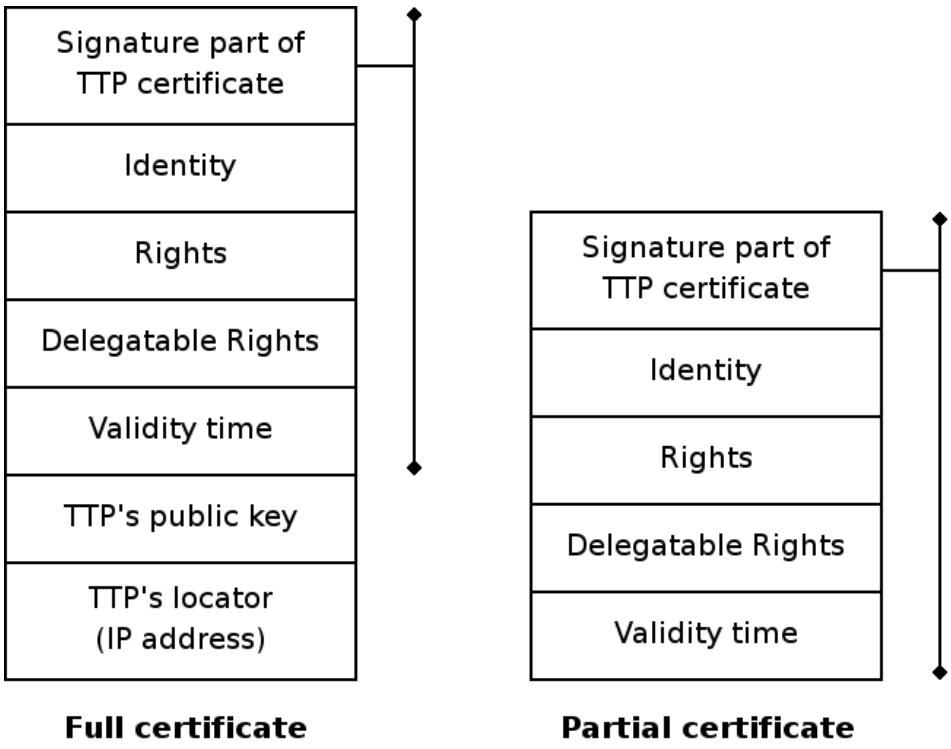


Figure 8. The TTP certificate format

Fields in the full TTP certificate are:

Signature part of TTP certificate. This field is used for calculation of the sender's public key, and it also implicitly signs the TTP certificate. If the calculated sender's public key successfully verifies the whole packet, then it also means that the TTP certificate is valid. The signature is calculated over the fields present in a partial

certificate, ignoring the TTP's public key and locator, and thus allowing the same TTP signature to be used with both certificate types.

Identity. Is the unique identity number given by the TTP to the user. The identity is necessary for identity-based implicitly-certified keys and it is utilized during calculation of sender's public key.

Rights granted by the certificate. TTP certificates can have different rights which are expressed in this field.

Delegatable rights. This field contains information about which rights can be delegated forward to another party. Delegation of rights is useful in some cases. For example, if a user wants to use another device temporarily as his own, he could delegate his existing TTP certificate to that device.

Validity time of the certificate. This contains not-before and not-after timestamps that denote the time frame during which the certificate is valid.

Public key of the TTP. The TTP's public key is used for calculation of the user's public key.

Locator of the TTP. Which contains the IP address of the TTP in order to allow nodes to contact the TTP that has authorized the sender.

In order for the TTP system to be effective, there should exist a way to guarantee that misbehaving users will not be able to have a valid TTP certificate for a prolonged period of time. Since there are hundreds of millions of users on the Internet, having a centralized revocation mechanism for all TTP certificates is not feasible. On the other hand, users who have been offline for extended periods of time should be able to continue normal communication without major issues. This problem can be solved using multiple certificate types with different rights and validity times. The type of the TTP certificate is checked at every node and the packet is prioritized accordingly. Different TTP certificate types are subsequently explained.

3.4.1 TTP certificate types and their usage

A normal certificate with a short validity time.

Packets sent with a normal, short-time TTP certificate will not have bandwidth limitations. If the owner of the short-time certificate misbehaves, the TTP will simply not renew the certificate. Thus, there is no need for a revocation mechanism for short-time certificates. The validity time for such certificate is typically on the order of hours or minutes.

A signalling certificate with limited bandwidth and long validity time.

This certificate type is designed to be used for retrieval of short term certificates instead of normal traffic. Such certificate is needed in cases where a device has been offline for a long time and its normal certificate has expired. The signalling certificate has a long validity time but a very limited bandwidth in order to reduce risks associated with a such long term certificate. For example, routers can reserve a small amount of their bandwidth, like 1%, for traffic that uses long term TTP certificates[1].

Because the validity time of the certificate is long, there should be a method to revoke certificates. Each TTP should maintain a list of signalling certificates that have been issued and revoked by this TTP. Then, the status of the certificate can be verified by querying the TTP that has issued the certificate. This method resembles the Online Certificate Status Protocol (OCSP) [37] revocation method. The validity time for such long-term certificate is on the order of months or years.

Self-signed (issued by the sender) certificate

A self-signed certificate is designed to be used in situations where the node does not have any kind of TTP certificate. This can occur when a signalling certificate has expired or when a new node is connected to the network for the first time. The traffic sent using self-signed certificates should be allowed only to the nearest router (for example, a WLAN access point). Afterwards, the router should encapsulate these

1 Routers can detect different TTP certificate types and prioritize traffic accordingly.

packets in its own PLA header. The PLA encapsulation mechanism is discussed in more detail in Section 3.4.2.

Such encapsulation provides several benefits. There will not be any packets in public networks that cannot be traced since the sender of every packet will be authorized by some valid TTP. In addition, the router will be responsible for the packets that it encapsulates, which means that it will choose carefully how much traffic that uses self-signed certificates the router will encapsulate. The router could allow only a few connections per minute using self-signed certificates with a very limited bandwidth per connection. This should be sufficient to retrieve a valid TTP certificate for those who need it, like a completely new computer or a computer whose long term certificate expired while it was offline.

3.4.2 Management of Trusted Third Parties

In order for a TTP system to be effective, there must be a way to check the validity of the TTP itself. Otherwise, the attacker could create an own TTP which would hand out valid TTP certificates to a large amount of attacker's nodes.

We use a system where TTPs form a tree-like hierarchy similar to DNS servers. Each TTP is trusted by some other TTP that resides on a higher level. On the top level the number of TTPs is relatively small and thus they can form explicit security associations between them. When a new TTP enters the network, it must be authorized by some existing TTP using the TTP certificate format discussed previously. As a result, each TTP has a certificate chain that contains certificates for all TTPs in the chain starting from the root TTP.

Figure 9 contains an example how validity of the unknown TTP can be verified. In this example, a router or a destination has received a packet with an unknown TTP X and wants to verify its validity. First, the router sends a verification request to its local TTP E containing the public key and locator of the unknown TTP X. If TTP E does not have the validity information about the unknown TTP X in its cache, it forwards the verification request to its parent. If the parent has the validity information about the requested TTP X it sends it back in a reply, otherwise it sends

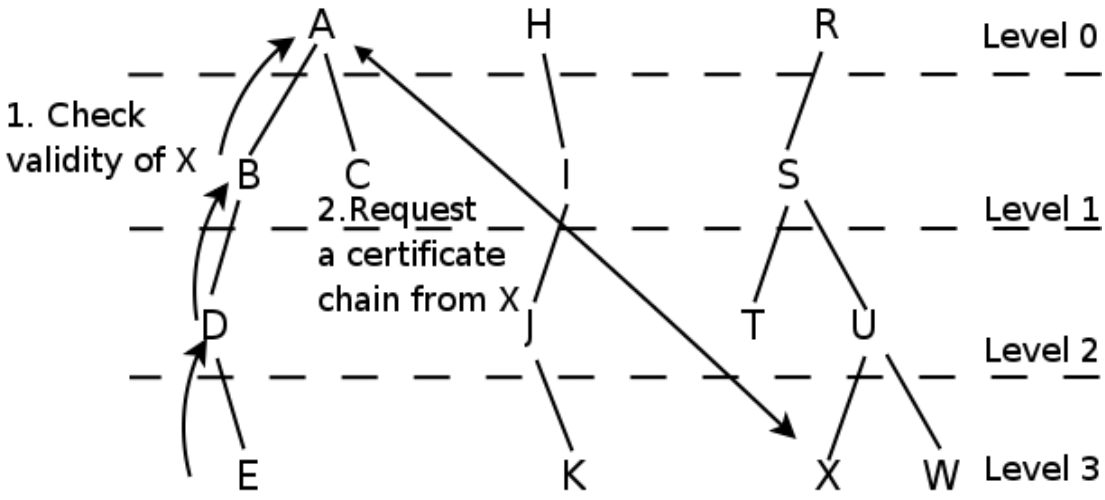


Figure 9. An example of verifying the validity of the unknown TTP

a request upwards to its parent and so on. When the request reaches the top, the top level TTP A will contact TTP X to request X's certificate chain. This chain contains a list of TTPs from the top level down to the TTP X (i.e., TTP R => S => U => X). TTP A checks that all the certificates in the chain are valid and have not been revoked. Since TTP A trusts the top level TTP R, it can also trust other TTPs in the chain including X. TTP A adds validity information about X in its cache and sends a reply to the verification request downwards and the reply eventually reaches the router that made the original request. When a TTP or a router receives a verification reply, it stores the verification information (validity status and duration) in its cache, thus it will not be necessary to send another request as long as cached validity information does not expire.

3.4.3 Encapsulation of PLA

The overall number of TTPs can be very high (several million) and this presents a problem with the performance and scalability of PLA. If a router receives packets from senders that have been authorized by a large number of unknown TTPs, the router must verify each TTP independently and such verifications will use a lot of time and network resources. This problem can be solved by encapsulating PLA packets that travel through the Internet into another PLA header. After a packet has been encapsulated, only the outer PLA header will be verified, which significantly

reduces the number of TTPs in which routers handling the packet must trust. An example of how such encapsulation could be used is shown in Figure 10.

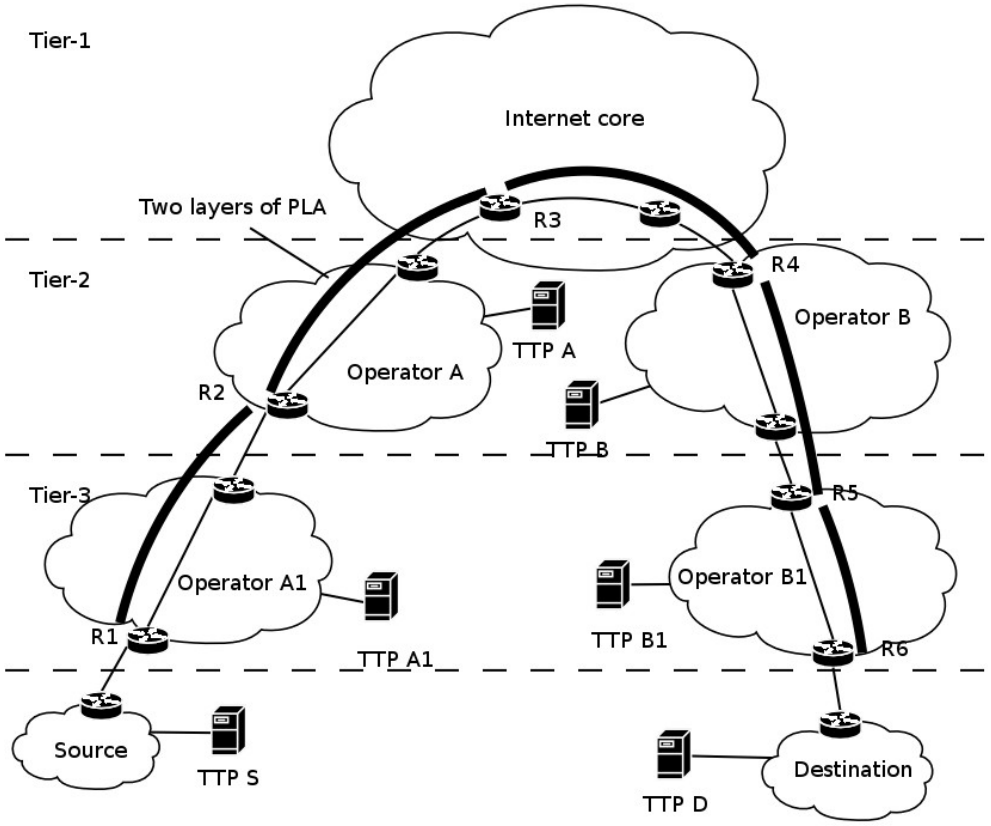


Figure 10. An example of PLA encapsulation

In the figure, clouds denote different networks which form a hierarchy: customers are connected to the Internet through their operators, small tier-3 operators are connected to larger tier-2 operators and a limited amount of the largest tier-1 operators form the Internet core network. For simplicity, only two layers of operators are shown in the figure; in reality, the hierarchy would be deeper and big operators would serve a large number of smaller operators. Thin lines in the figure denote ordinary PLA connections while bold lines denote encapsulated connections where two layers of PLA are used.

In this example, a node in the “source” network which is connected to the Internet through operator A1 is communicating with a node in the “destination” network which uses operator B1 for Internet connection. The sending node is authorized by a local TTP S. The encapsulation works as follows: as the packet arrives to router R1, which is an edge router of operator A1, the router verifies the packet's validity

including the validity of TTP S which has authorized the sender of the packet. If this validity check is successful, the router R1 encapsulates the packet in its own PLA header and sends the packet forward. This means that the new header includes R1's public key and TTP certificate received from TTP A1, since TTP A1 is responsible for operator A1's network[1]. As the packet travels through A1's network, intermediate routers will only check the outer PLA header of the packet, thus they will not need to trust TTP S that has authorized the original packet. Eventually the packet reaches router R2 which resides at the edge of operator A's network. Router R2 performs a validity check on the packet, including a check on validity of TTP A1, strips the outer PLA header from the packet, encapsulates the packet in its own PLA header and sends the packet forward. The new PLA header naturally contains a TTP certificate from TTP A. The example continues in a similar way, as the packet arrives in a new network, an edge router replaces the outer PLA header with its own header. Eventually the packet arrives to the operator of the destination network, where router R6 strips the outer PLA header and sends the original packet to its final destination.

The advantage of such a system is that only the end point of the connection needs to perform the costly validity check of TTP S that has authorized the sender of the packet, this validity check resembles a reverse DNS check performed today. Operator edge routers need to trust in the TTPs of their “child” and “parent” operators, while intermediate routers inside operator networks need only trust in their own local TTP, because encapsulation guarantees that every packet which is transmitted in an operator's network is sent by a node which is certified by that operator's own TTP. Inside the core network, routers would only need to trust the TTPs of major operators. In this example, router R3 would need to trust TTP A. Therefore, this approach significantly reduces the amount of TTPs that intermediate routers must trust, in most cases a router need only trust its local TTP, and thus the overhead produced by TTP verification queries is greatly reduced. On the downside, the encapsulation produces some extra bandwidth overhead because there are two PLA headers in each packet, and requires more computational power in routers which perform the encapsulation since a new signature must be generated to every packet.

1 Use of encapsulation would also mean that CGA address checks would fail, since CGA addresses are generated from the sender's public key which is different from public key used for encapsulation. However, this does not pose a significant problem if the CGA address check is performed only in access networks where packets are sent without encapsulation.

However, encapsulation is still feasible to use within operator networks since they usually have a high amount of bandwidth available and routers will have powerful dedicated hardware to perform cryptographic operations.

Theoretically, routers could remove the original PLA header altogether and just replace it with their own header. Such a system would satisfy the requirement that only valid packets are forwarded in the network. However, in order to satisfy the rest of the requirements presented in Sections 2.5 and 3.1, the sender of the packet must be identifiable and thus the original PLA header must be present in every packet.

The use of encapsulation also means that there is no need to have a centralized revocation scheme for TTPs of major operators. If an operator's TTP becomes compromised, then the operator simply informs its “parent” and “child” operators of this and their nodes will stop trusting the compromised TTP. The number of minor TTPs can be very high, and thus it is more effective to use certificates with a short validity times (couple of hours) to manage TTPs. Each TTP will receive a short-time certificate from its parent TTP and this certificate must be renewed. If the TTP misbehaves, its parent will not renew the certificate and the TTP will not be trusted by other parties.

3.5 Bootstrapping a new node to use PLA

PLA relies on certificates from trusted third parties and a new device must have a method for retrieving all relevant certificates for communication. This section describes one possible bootstrapping procedure for a new device, i.e., how a new device can retrieve certificates to communicate fully using PLA. The bootstrapping example shown in Figure 11 assumes that the device is brand new, it does not have any certificate from a TTP and it does not even have a public/private key pair. It is also assumed that the device stores its private key locally. It would also be possible to store a private key in a some kind of secure portable device, in which case the user could carry his private key with him and use it with several devices.

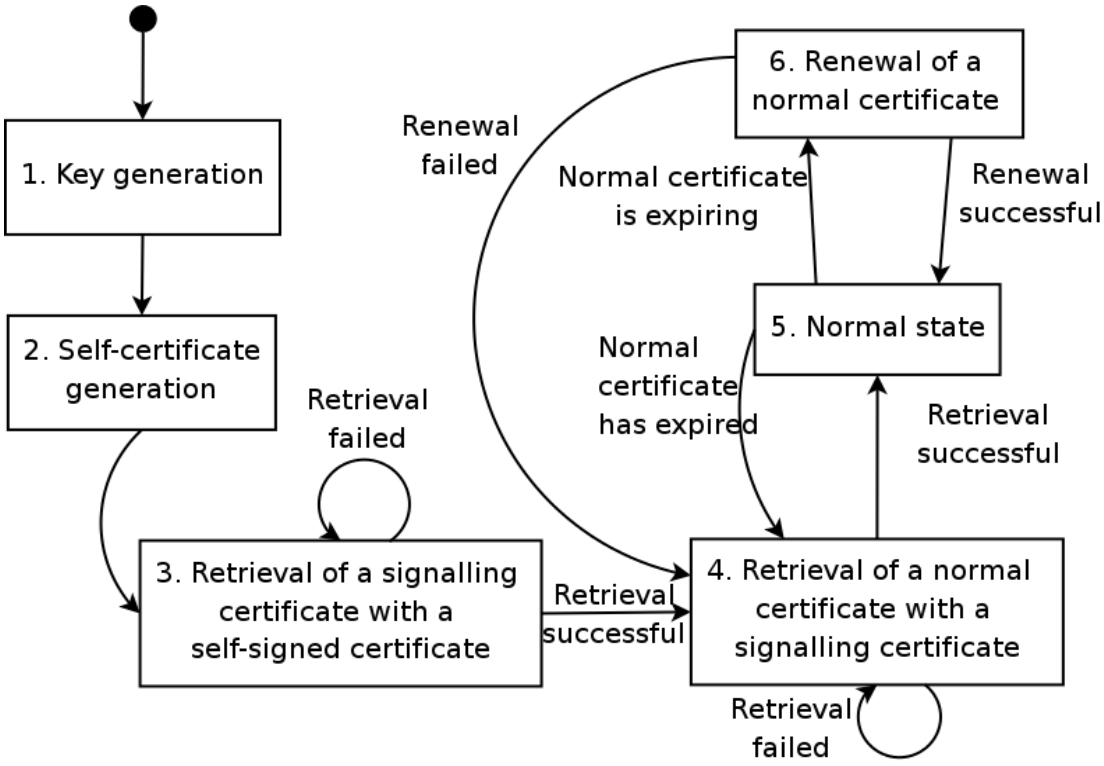


Figure 11. The bootstrapping procedure of a new node

After the new device is turned on, it moves to the key generation state where it generates a public/private key pair for itself. In step 2, the device generates a self-certificate that has the same format as a TTP certificate but is signed with device's own private key. Now the device has all of the necessary keys and certificates for limited communication, and it can retrieve a long term signalling certificate from a TTP with its own self-signed certificate in state 3. Afterwards, the device retrieves a normal short term certificate from a TTP using the previously received signalling certificate. If this retrieval is successful, the device enters normal state 5, where it can communicate normally with PLA without any bandwidth limitations. When this normal short-time certificate is about to expire, the device needs to request renewal from the TTP in state 6. If the renewal fails or if the normal certificate has already expired, for example because the device was turned off for a long period of time, then the device moves back to state 4 where it needs to retrieve a normal certificate. Theoretically, also the long term signalling certificate can expire, in which case the device would move back to step 3. This step (retrieval of the initial signalling certificate from the TTP) is discussed in more detail in the next section.

3.5.1 Retrieval of the initial certificate from the TTP

Figure 12 illustrates how a new device can retrieve an initial certificate from the TTP. The certificate format used in the figure is an S-Expression format which is described in Appendix A. In this case, the TTP service is provided by an operator with whom the owner of a computer has a contract with and the user requests a TTP certificate for his PC. The idea is that the operator provides an authentication token to the user, and the token is used by the user to prove his identity during the certificate retrieval. The token acts as a one time password which is bound to a specific public key and the token can be exchanged offline between the user and the operator.

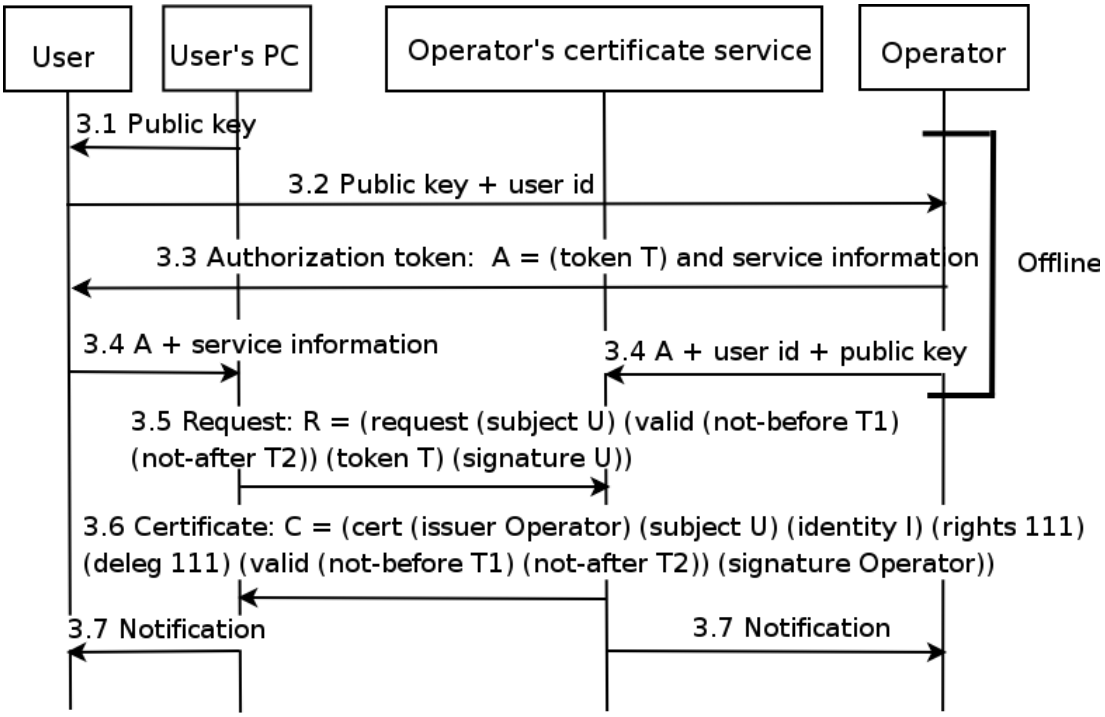


Figure 12. Process of retrieving initial certificate from the trusted third party

The retrieval process proceeds as follows. In steps 3.1 and 3.2, the user retrieves the public key[1] of the PC and sends it together with some kind of user id, such as customer number, to the operator. In step 3.3, the operator gives the user an authorization token containing service information such as the address of operator's TTP. Then in step 3.4 the user enters the service information to their PC while the operator sends given authentication token, user id, and the public key to its TTP. In

1 The public key mentioned here is the preliminary key used during the certificate retrieval process. A final public key is extracted from information received from a TTP in step 3.6 based on formulas mentioned in Section 4.1.2.

step 3.5, the request for a new certificate is sent. This request contains a public key, a requested validity time for the certificate, the authentication token, and a signature. The operator's TTP now verifies that the public key and authentication token in the request match the values sent by operator. If they match the TTP sends a final TTP certificate back to the user in step 3.6[1]. In the final step 3.7, the user and the operator are notified that the certificate has been issued and received successfully.

3.5.2 Delegation of TTP certificate to another device

In some cases it is useful to have the ability to delegate rights to communicate (e.g., to request a valid TTP certificate) to another device either permanently or temporarily. For example, if a user wants to use a friend's computer as his own for a while, he could temporarily delegate his rights to that computer. Or if the user wants to take into use some home appliance which lacks traditional input methods, he could permanently delegate his rights to that appliance.

In principle, delegation of rights works as follows. A valid user authorizes another device with a certificate, thus creating a certificate chain: TTP => user => device. Using this certificate chain, a third party can then request a new certificate from the user's TTP. An example of such delegation is shown in Figure 13.

In the first step, an appliance generates a public key for itself which is then transmitted to the user. The means by which data is transmitted in steps 1 and 2 is irrelevant, it can be transmitted using a network or offline using a separate memory stick. In the second step, the user transmits two certificates to the appliance: the first is the user's TTP certificate which states that the user is a valid and trusted entity, and the second is the certificate issued by the user to the appliance. These certificates form a certificate chain: TTP => user => appliance and thus with these certificates, the appliance can request its own certificate from the TTP in steps 3 and 4. After this certificate C3 has been received in step 4, the appliance will be able to communicate using the PLA.

1 To protect privacy, traffic in steps 3.5 and 3.6 must be encrypted by some means, such as SSL.

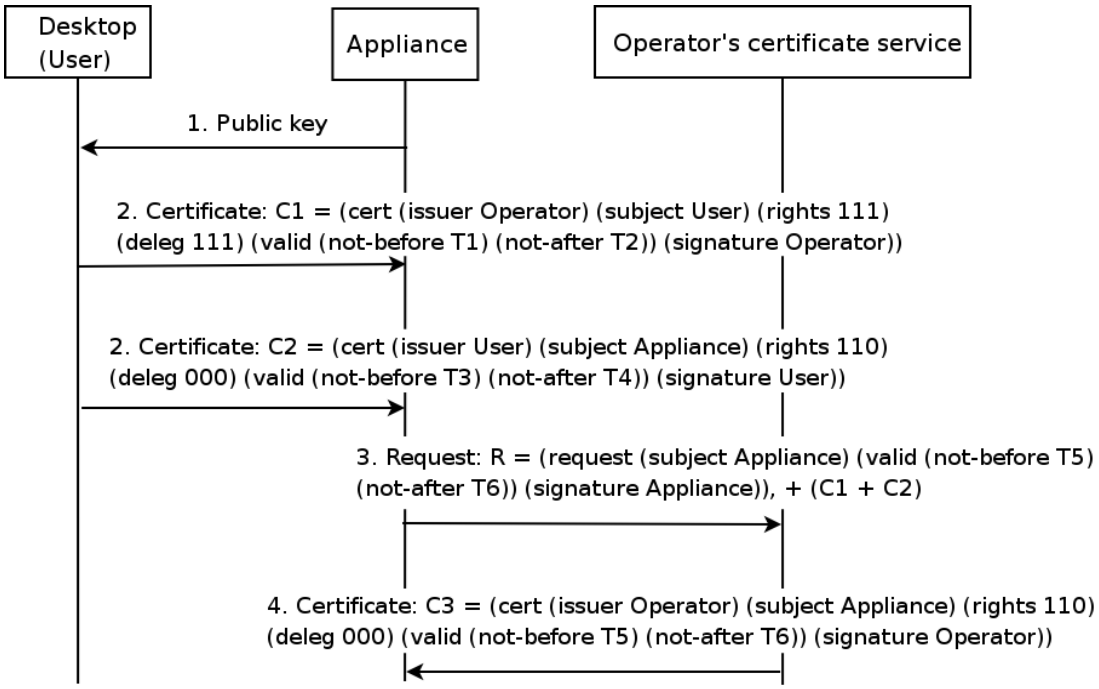


Figure 13. An example of delegation rights to another device.

3.6 PLA header verification procedure

Every packet's PLA header must be verified at every node that handles the packet in order for PLA to be effective. The verification of the PLA header consists of several steps and a basic case is shown as a state diagram in Figure 14. The aim of this example is to show what steps are always present in PLA header verification and what are possible outcomes. This example assumes that the TTP certificate inside the header is a normal short term certificate without bandwidth limitations.

The exact order of steps may differ, in certain cases performing header verification in different order might be more efficient in terms of computational resources or offer better protection against denial-of-service attacks. The main aim of PLA header verification is to check whether a packet is authentic and how much a sender of the packet can be trusted. Based on the outcome of these tests, a node will determine how to handle the packet.

In the beginning of the verification procedure, the timestamp and the sequence number are checked, if either of them is invalid then the packet has been significantly delayed or duplicated and is thus discarded. In the second step, the sender's public

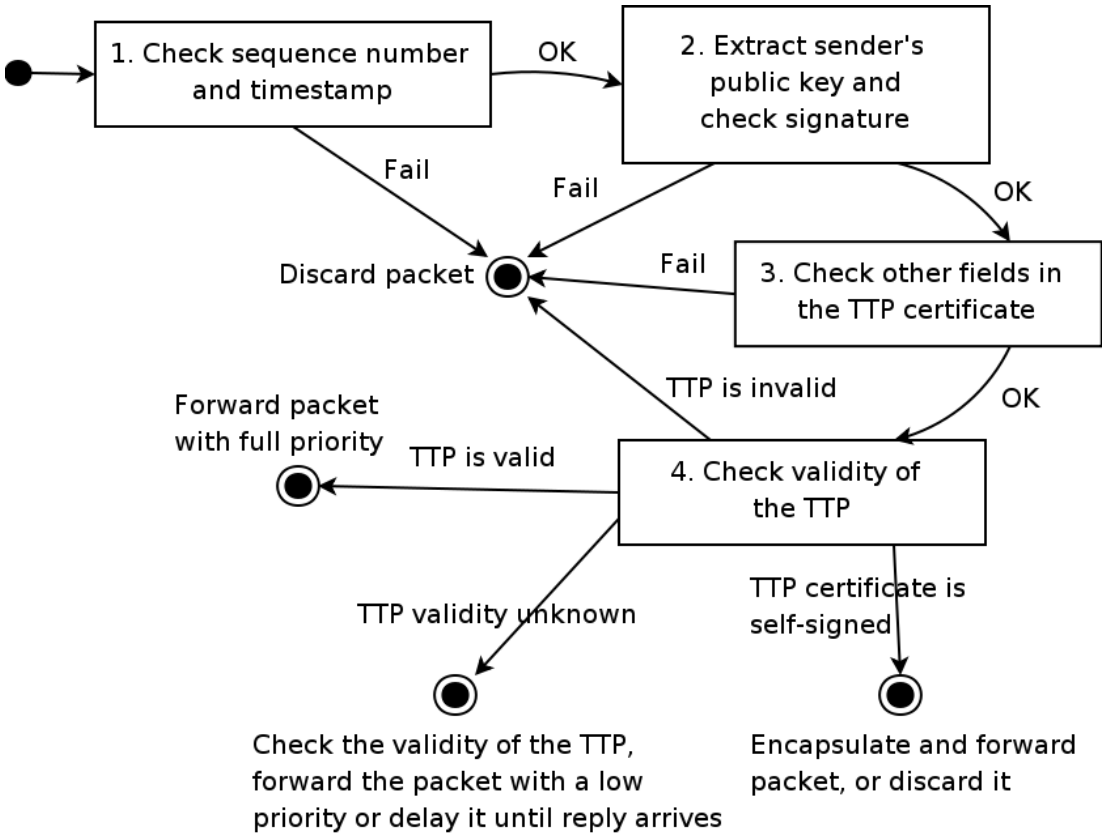


Figure 14. A state diagram of PLA header verification

key is extracted from the TTP certificate and the cryptographic signature of the packet is verified using the extracted public key. If this verification fails, the packet is inconsistent and is therefore discarded. In the third step, other fields of the TTP certificate, such as the validity time and rights of the certificate, are checked. If the TTP certificate has expired or if it does not have valid rights, the packet is discarded.

In the final step of the verification procedure, the validity of the TTP which issued the certificate is checked. If the TTP is invalid (cannot be trusted), the packet is discarded. If the TTP is known to be valid, everything is in order and the packet is forwarded with full priority. If the TTP's validity is unknown, the node checks the validity of TTP by sending a request to an own TTP. While waiting for a response, the node forwards the packets in question with a low priority, or alternatively, the node could delay the packets while awaiting a reply regarding the TTP validity. The latter option would enhance security, although it would introduce an additional latency for the initial packet. Finally, if the TTP certificate is generated by the sending node of the packet (i.e., it is self-signed), the node has two options. Either

the node simply discards the packet, or the node can encapsulate the packet in its own PLA header and it forward it. Encapsulating the packet would mean that the node is taking full responsibility for the packet, and thus the node should only allocate a small portion of bandwidth for such packets with self-signed TTP certificates. Self-signed certificates are only used when the sending node has just started operating and does not have any valid certificate from a TTP yet.

The router has a lot of flexibility in terms of performance/security trade off when deciding how to prioritize packets. The router can adopt a "paranoid" policy, in which case it would:

- Delay or discard packets which were sent by a user that has been authorized by an unknown TTP, until the validity of the TTP has been verified.
- Query the status of encountered signalling certificates from TTPs that have issued those certificates, and allocate very limited bandwidth to packets that use a signalling TTP certificate.
- Discard all packets that use self-signed TTP certificates.

Or a more relaxed policy, like:

- Forward all packets immediately, even if their TTP is unknown.
- Encapsulate packets that use self-signed certificates in an own PLA header.
- Allocate more bandwidth to self-signed and signalling certificates.

Most importantly, PLA offers routers a consistent way to decide how different kinds of packets should be handled. Routers can then make a decision depending on their policy. The optimal policy for each router depends on several factors and determining the optimal policy is beyond the scope of this thesis. For example, routers in civilian and military networks may have a very different policy for handling packets, just like routers in access networks may have a different policy compared to routers in core networks.

4. Implementing PLA

This chapter describes cryptographic solutions used with PLA and proof-of-concept software and hardware implementations of PLA. It also contains some performance results and an analysis of the performance impact of PLA.

A proof of concept Linux implementation of PLA has been made [40]. The implementation uses IPv6 and supports packet authenticity checking and offers support for hardware acceleration for signature verification. The current implementation consists of a kernel module and userspace applications. The implementation supports three modes of operation:

1. The PLA header is added to a plain IP packet and the PLA enabled packet is sent forward.
2. The PLA header is removed from an existing PLA enabled packet (after checking the packet's validity) and the plain IP packet is sent forward.
3. The PLA enabled packet is simply forwarded after performing validity checks.

The implementation also supports several PLA layers (PLA encapsulation). The proof of concept implementation makes it possible to build a network consisting of non-PLA enabled nodes and PLA enabled nodes. A general architecture of the PLA software implementation is presented in Figure 15.

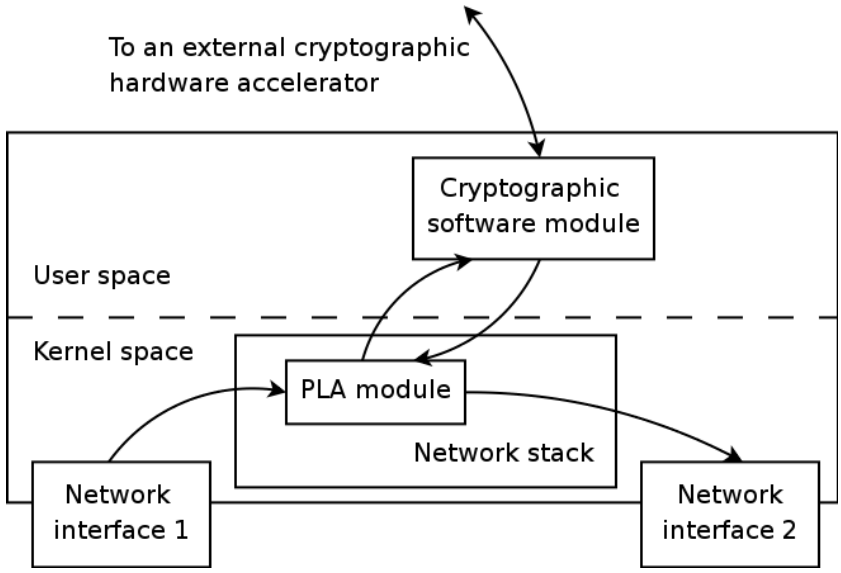


Figure 15. PLA software implementation architecture

As the packet arrives to the network interface, it is handled by the network stack of the operating system. The operating system contains a PLA module which will handle PLA related operations such as adding a PLA header to a plain IP packet and verifying the validity of the PLA header. The PLA kernel module communicates with a separate cryptography module which resides in the userspace. For testing purposes this cryptography module supports both software and hardware based cryptographic solutions. In addition, the implementation supports adding a PLA header to packets with a blank signature without making any cryptographic computations. This feature is useful for testing the non-cryptographic overhead of the PLA implementation.

4.1 Cryptographic solutions

PLA is based on public key cryptography with public keys and signatures included in every packet. Thus, to limit the overhead produced by PLA, the cryptographic solution used should offer good security with relatively small key and signature sizes. For this reason, PLA uses Elliptic Curve Cryptography (ECC) for cryptographic tasks. ECC is a very efficient algorithm in terms of security per key size, a 163-bit ECC key has roughly the same cryptographic strength as a 1024-bit RSA key or an 80-bit symmetric key [34]. Although PLA itself is not dependent on the used cryptographic algorithms, the ECC algorithm is currently the only feasible solution, since other solutions would require significantly longer cryptographic keys and signatures to achieve the same level of security.

The PLA implementation uses a standardized Kobliz curve K-163 [18] defined over a binary field because binary field operations are fast in both hardware and software. PLA uses 164 bit compressed public keys, 163-bit private keys, and 326-bit signatures. These key sizes provide security roughly equivalent to that of 80-bit symmetric algorithms. According to the U.S. National Institute of Standards and Technology (NIST) recommendations [8], such a level of security is sufficient until the year 2010. As computational power increases in the future it will become easier to use a brute force attack to break cryptographic algorithms and stronger cryptographic solutions will be required. In this respect, ECC is a good solution because its efficiency in comparison to RSA increases as key sizes become larger. For example, increasing an ECC key size from 163 to 233 bits (an increase of 43%)

would increase security as much as doubling a key size in the RSA algorithm from 1024 to 2048 bits. According to the NIST recommendations, such security should be sufficient until the year 2030.

4.1.1 Per packet signature generation and verification

Per packet signatures are generated as follows using ECC. Let G and r be elliptic curve related global parameters and s a private key. A public key W is generated by computing $W = sG$. A cryptographic signature on message m consists of two values (c,d) and is computed as follows:

$c = [uG]_x + \mathrm{H}(m) \pmod r$ where $u \in_R Z_r^*$

$d = u - sc \pmod r$

where $[\mathrm{P}]_x$ is the x-coordinate of the elliptic point P converted to an integer and H is a collision-resistant hash function. A hash is computed over the whole packet ignoring the hop limit field of an IPv6 header. Any other party can verify the packet's signature (c,d) by checking that:

$\mathrm{H}(m) = c - [dG + cW]_x \pmod r$.

4.1.2 Trusted Third Party certificates

In order to reduce packet overhead and save computational power, PLA uses identity-based implicitly-certified keys [12], which means that it is not necessary to include the sender's actual public key in the PLA header. Generating public keys with implicit certificates for the sender works as follows. Let G and r be elliptic curve related global parameters, ID user's identity[1], s_{TTP} a TTP's private key, and W_{TTP} a TTP's public key. In the beginning, the user generates and sends kG to the TTP where $k \in_R Z_r^*$. The TTP calculates:

$$
\begin{aligned}
(\bar{r}, b) &= \textsc{compress}(kG + k_T G) \text{ ,where } k_T \in_R Z_r^* \\
r_u &= \bar{r} + \mathrm{H}(ID) \\
\bar{s} &= k_T - r_u s_{TTP} \pmod r
\end{aligned}
$$

where COMPRESS is the point compression function giving the x-coordinate of uG and the compression bit b. The TTP sends its signature $(r_u, b), \bar{s}$ back to the user

1 String *ID* contains several fields of the TTP certificate presented in Figure 8: identity, rights, delegatable rights, and validity time.

who calculates his private key $s = k + \bar{s} \pmod{r}$ and public key $W = sG$. The signature part of a TTP certificate which is included in PLA header is (r_u,b). Afterwards, a third party can extract the user's actual public key W from the signature part of the TTP certificate (r_u,b), the user's identity ID, and the TTP's public key W_{TTP} by calculating:

$$W = \text{DECOMPRESS}(r_u - \text{H}(ID), b) - r_u W_{TTP},$$

where DECOMPRESS is the point decompression function given an x-coordinate and compression bit b. If the extracted public key W successfully verifies the packet, then it automatically guarantees that the implicit certificate given from the TTP to the user is valid.

4.1.3 Improving the efficiency of cryptographic operations

PLA also uses novel ideas for increasing the efficiency of various ECC related calculations. A new method for performing the public key extraction and verification mentioned above using only a single three-term simultaneous scalar multiplication is described in [9]. Another method for improving the performance is discussed in [10]. PLA also utilizes a new method of computing the integer equivalent of random Frobenius expansions [11], which significantly speeds up generation of cryptographic signatures. Fast signature generation is beneficial for nodes that are sending a large amount of PLA packets and for nodes that encapsulate PLA traffic.

4.2 Hardware acceleration of cryptographic calculations

Since public key cryptography is very computationally intensive, it is not feasible to use general purpose CPUs for such cryptographic computations. One of PLA's design goals is to scale to high capacity networks and to devices with low computational resources, thus dedicated hardware acceleration is required to handle cryptographic tasks. In the future, such a cryptographic accelerator could be integrated into technologies such as Trusted Platform Module (TPM) [48], which aim to enhance the security of a personal computer by using an additional security module.

There are two computationally intensive problems where hardware acceleration is useful: validation of packet signatures, and generation of signatures for packets. The

former operation is performed by any node that forwards PLA traffic while the latter is performed by nodes that send packets or encapsulate traffic. Both problems are very parallel in nature since packets received by a node can be verified independently. Thus, the throughput (the amount of packets that a node can process in a given time frame) is more important than the speed of a single operation (the latency) as long as the latency is low compared to the latency of the network itself.

Figure 16 describes one possible architecture for a PLA hardware accelerator, assuming that cryptographic operations are fully done in hardware. The left side of figure contains PHY and MAC interfaces which are also present in normal network interface chips while the right side contains PLA-related functionality. The hardware accelerator would work as follows. After incoming packets pass the PHY and MAC interfaces, they are forwarded to the hash unit and ECC modules which are responsible for performing ECC related cryptographic operations.

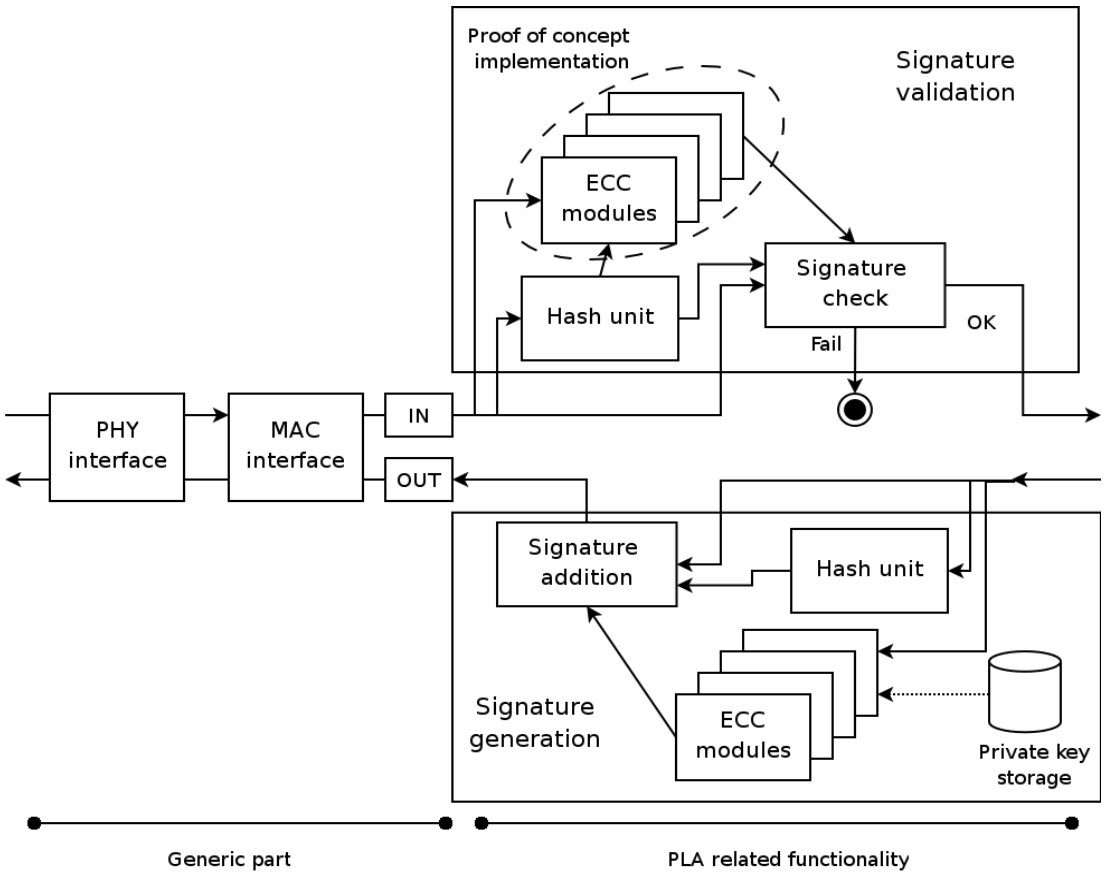


Figure 16. An example hardware acceleration architecture for PLA

Afterwards, the packet's signature is checked, and bad packets are discarded immediately while good packets are sent forward to upper layers of the network stack where the remaining PLA-related validity checks are performed. For outgoing packets, a signature is generated using the same principles and similar dedicated hardware units. There also exists a storage for private keys which are used for signature generation.

4.2.1 Proof of concept PLA hardware accelerator

A proof of concept implementation of the hardware accelerator for ECC related calculations has been made for PLA [26]. The initial hardware accelerator is based on a field programmable gate array (FPGA) and supports acceleration of an elliptic curve point multiplication $Q=kP$, where Q and P are points on an elliptic curve and k is an integer. This type of operation is an essential part of the ECC algorithm. Basically, the proof of concept implementation covers cryptographic modules from the upper part of Figure 16.

An FPGA is a chip containing a large amount of programmable logic, including logic gates, I/O, and memory. The main advantage of an FPGA is its flexibility. The FPGA can be programmed to perform different tasks, therefore FPGAs are very suitable for prototyping purposes. On the downside, FPGA offers significantly lower performance than the application specific integrated circuit (ASIC) [32]. An FPGA usually runs at relatively low clock speeds but contains a large amount of computational units which can perform several computations in parallel, making it a suitable solution for the acceleration of PLA cryptographic computations.

The proof of concept PLA hardware accelerator consist of a separate board containing the FPGA chip. This board is connected to the host computer by Ethernet and communication between the host and the accelerator is handled using TCP. At a high level, the accelerator works as follows; the host computer sends values of k and P to the accelerator, which performs the point multiplication and returns Q to the host computer. An overview of the accelerator is presented in Figure 17, while Figure 18 in the next section describes how the test network utilizing the proof of concept accelerator is configured.

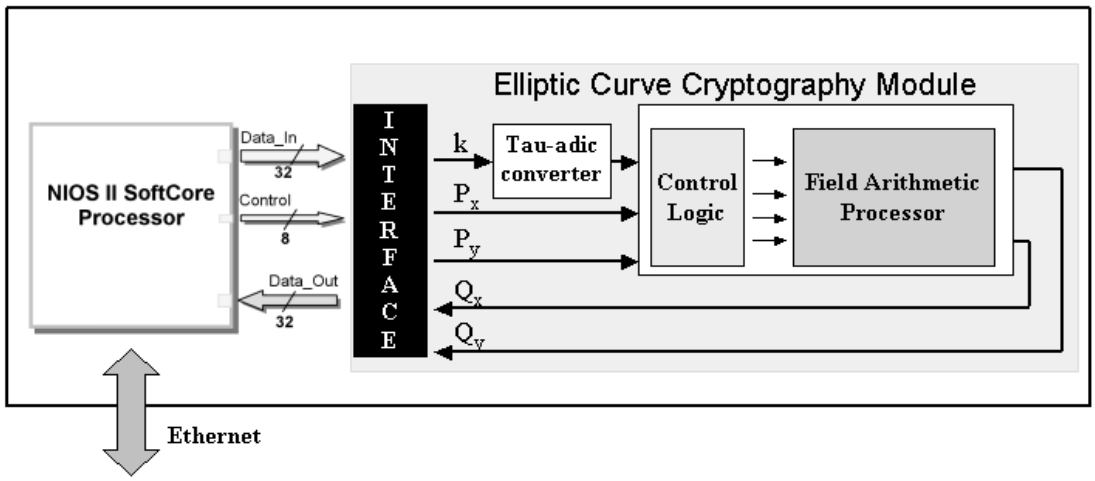


Figure 17. An overview of the proof of concept PLA hardware accelerator

The accelerator is divided into two main parts: an I/O part and the actual cryptography module which accelerates ECC calculations. A Nios II soft-core processor is connected to the cryptography module over an internal bus and is used for I/O communication with the host computer over Ethernet. This I/O part is necessary for the proof of concept implementation where a majority of the workload is done on a router's CPU. In a final accelerator design, the cryptography module would be directly connected to the network hardware. The elliptic curve cryptography module consists of tau-adic converters and field arithmetic processors (FAPs). A FAP processor is the part of the accelerator which is responsible for performing the elliptic curve point multiplication.

Koblitz curves, which are used in the ECC implementation, require that the scalar k used in the point multiplication $Q=kP$ is converted into a so-called tau-adic expansion [30]. This conversion is not trivial to perform and it needs to be handled by hardware in order for the accelerator to achieve its full potential. Thus, the cryptography module contains separate tau-adic converters which are described in more detail in [27].

Currently, the proof of concept hardware accelerator has some limitations. The Ethernet interface of the accelerator is limited to 100Mbps speed, which becomes a bottleneck in certain situations. In addition, the PLA software implementation is not optimized and communication latency between the host PC and the proof of concept accelerator is relatively high.

An optimized design for performing verifications has been made based on an Altera Stratix II EP2S180 FPGA board. The used FPGA chip is built on a 90 nm manufacturing process and it contains 96 digital signal processing elements and approximately 1 Megabyte of memory. The simulated performance of this FPGA accelerator is very good. When optimized for throughput, it achieves 166,000 verifications per second with a latency of 114 µs per verification [28]. In this case, 19 FAP processors are used in parallel at a clock speed of 164MHz. The design can also be optimized for latency, in which case the latency per verification decreases to 35 µs [25].

4.3 Performance impact of PLA

Since PLA uses public key cryptography to check every packet, it requires a lot of computational resources. One important question is, how much computational overhead does PLA cause? To test the performance of the PLA implementation, a test network, shown in Figure 18 was built[1].

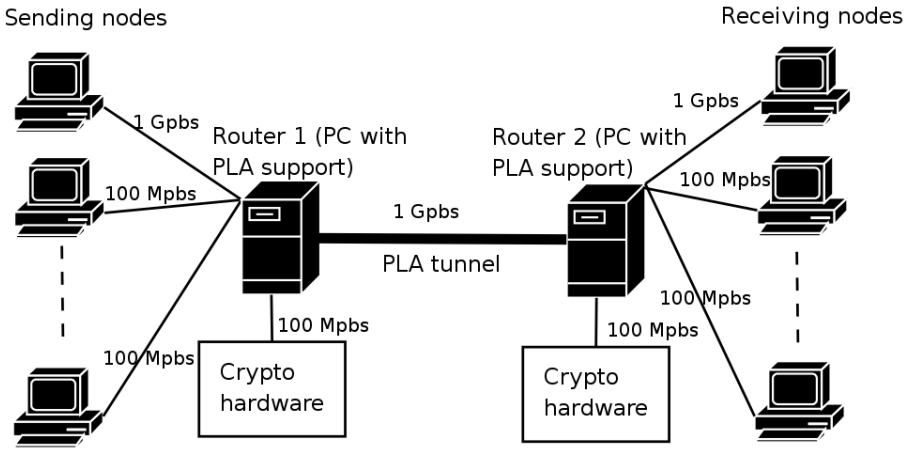


Figure 18. Overview of the PLA test network

The test network consist of three major parts: sending nodes that just send a large amount of packets, receiving nodes that receive those packets, and PLA enabled routers in between. Sending and receiving nodes are ordinary low-end PCs that simply generate and receive a large amount of plain IP packets. The routers are more

1 The connection between cryptographic accelerators and routers is limited to 100 Mbps because the proof of concept hardware accelerator supports currently only 100 Mbps network interface.

powerful PCs that also have a connection to external hardware that accelerates cryptographic tasks.

The test network works as follows. Traffic from sending nodes arrives to Router 1, which adds a PLA header to each packet. Then PLA enabled packets are sent to Router 2. Router 2 removes PLA headers and sends plain IP packets to receiving nodes. Performance results are discussed in more detail in the next sections.

4.3.1 Latency impact without the signing and the verification of packets

Results shown in Table 1 below indicate the amount of overhead that PLA processing adds when the signing and the verification of packets is not performed. These results show the impact of non cryptographic overhead of the PLA implementation, like the impact of adding PLA headers to plain IP packets or removing them. A latency was measured with *ping6* software using normal and flood pings, in both cases, 100,000 packets were sent between the sending and receiving nodes. The latency values below include the time consumed by four PLA related operations: addition of the PLA header in Router1, removal of the PLA header in Router2, and the same steps in reverse during the lifetime of a reply packet. All tests were run three times and the results were averaged.

Table 1. Additional latency of PLA without any cryptographic operations

Type of ping	***Average round trip latency (µs)***	***Standard deviation***	***Average per hop latency (µs)***
Without PLA			
Normal	427,3	0,6	106,8
Flood	460,7	16,3	115,2
With PLA			
Normal	519,0	1,0	129,8
Flood	524,0	14,2	131,0

As can be seen from results adding and removing the PLA header adds about 16-23 microseconds of latency per operation, increasing the round trip latency in the test

from approximately 430 µs to 520 µs. Such a latency increase relatively insignificant, if we assume that an average route contains about 15 hops, then these operations would add only about 0.6 milliseconds to the round-trip latency. It is also important to note that these values were achieved by a proof of concept software implementation of PLA. In an optimized case, the latency would be significantly lower.

In addition, cryptographic operations like generating signatures for packets and verifying signatures will add additional latency. The impact of this latency is discussed in the next section.

4.3.2 Analysis of the performance impact and power consumption

In stand-alone simulations, the FPGA based cryptographic accelerator achieved 166,000 signature verifications per second. If we assume that the average size of an IP packet is about 6000 bits (750 bytes), then such accelerator could verify approximately 1 Gbps worth of traffic. With a minimum packet size of about 1000 bits (minimal IPv6 packet with a PLA header), such an accelerator could handle 166 Mbps of traffic. According to Altera's PowerPlay simulation tool, the power consumption for this FPGA implementation would be about 20 W under load. Thus, about 20 W of power is required for verifying 1 Gbps of average traffic in this case and the energy consumption per validated packet would be about 120 µJ. It is important to note that these results were achieved with a programmable FPGA, which achieves significantly lower performance compared to a dedicated application specific integrated circuit (ASIC) made on the same manufacturing process. In addition, the FPGA used in testing was released in 2005 and is manufactured on a relatively old 90 nm process and is thus not the fastest FPGA available on the market today.

Altera offers a Hardcopy structured ASIC technology [2] which allows an existing FPGA design to be easily converted into a structured ASIC. Such a structured ASIC would achieve a significantly higher performance and lower power consumption compared to a programmable FPGA. However, the structured ASIC would still not be as efficient solution as a fully customized ASIC. Based on Altera's simulation

tools, converting the existing FPGA PLA accelerator design to a Hardcopy II HC240F1508I 90nm structured ASIC would result in a performance of 850,000 verifications per seconds with an estimated power consumption of 25W. The Hardcopy ASIC would also have a higher clock speed of 210MHz and thus the latency of a single operation would decrease to about 89 µs. Therefore, such an ASIC would require only about 5 W of power to verify 1 Gbps of traffic with an average packet size of 6000 bits per packet, and the energy consumed per packet would decrease to about 30 µJ.

Currently, the most modern manufacturing process available is a 45 nm process which is two generations ahead of the 90 nm process used in the current FPGA implementation. The transistor density of an integrated circuit roughly doubles with each new process generation while the supply voltage of the circuit decreases, which reduces overall power consumption. Thus, due to more advanced manufacturing process and because a customized ASIC achieves a better performance and lower power consumption than a structured ASIC, it is reasonable to estimate that a customized ASIC solution manufactured on a 45 nm process would be able to verify tens of gigabits of average traffic with a power consumption of less than a couple of watts per gigabit of traffic[1].

Another important PLA-related performance issue is the additional latency produced by PLA. This latency can be divided into three main parts: the latency of cryptographic computations, the latency produced by communication between the main processor and cryptographic accelerator, and the latency added by non-cryptographic tasks such as checking the timestamp and the sequence number within a packet. According to the results from the previous section, the latency of PLA header addition or removal is about 20 µs; however, this operation is not performed at every node and this result was achieved by a proof of concept software implementation. The latency of communication between the main processor and the cryptographic accelerator would not be significant in an optimized case, because in a dedicated PLA enabled router, the hardware accelerator would reside either on a PCB

1 The current FPGA design is optimized for a specific elliptic curve type and length, this is feasible since the FPGA design can be easily updated. A customized ASIC should include support for several ECC related parameters which would slightly decrease it efficiency, still it would be far more efficient than an FPGA solution.

connected directly to the main network processor by a fast, low latency bus, or inside the network processor itself, in which case the latency would be even lower. Thus, when measuring latency impact of the PLA in an optimized case, the latency of actual cryptographic operations is the most crucial. This latency amounts to 90 μs per verification using a structured ASIC and this latency depends linearly on the clock speed of the circuit. Thus, in an optimized case this latency would be decreased further.

Overall, the latency impact of PLA is not significant. If we assume that a typical connection uses about 15 hops and the total latency impact of packet verification in an optimized case is around 100 μs per router (70 μs for cryptographic operations and 30 μs for other operations including communication overhead), then this would produce only a 1.5 ms extra delay in one direction and a 3 ms increase in round-trip time.

4.3.3 Improving the efficiency of PLA

One way to further increase the efficiency of the PLA implementation at high network speeds is to use jumbo or super jumbo frames, which have a frame size of 9000 or even 64000 bytes instead of a normal frame size of 1500 bytes. Therefore, inside high speed networks, several PLA packets could be encapsulated into a single jumbo frame, significantly decreasing the amount of verifications needed per unit of traffic. For example, if six packets are encapsulated into a single jumbo frame, then the amount of PLA verifications would drop to one-sixth of the original. Using jumbo frames would provide three main benefits: less computational resources would be required for packet verification, power consumption would be also decreased, and bandwidth overhead produced by encapsulation would be lower, since an additional PLA header would use only a very small part of a jumbo frame payload.

Depending on the used security policy, routers do not necessarily need to check every passing packet. In a normal situation, it could be sufficient to check every third or every tenth packet randomly without significantly reducing the security of the network. If the attacker floods the network with a large amount of invalid packets, routers would catch at least some of invalid packets sent by the attacker even if they

do not check every packet that they forward. In addition, connections usually go though more than ten routers, and thus it is probable that every packet will still be checked at least once as it travels through the network.

Overall, PLA is a very scalable solution which can be used in a very high bandwidth networks as long as a dedicated hardware is used for accelerating cryptographic operations.

5. Analysis of PLA implementation

This chapter analyses how well PLA meets its design criteria presented in Section 3.1 and the requirements for the next-generation Internet presented in Section 2.5.

5.1 PLA design criteria analysis

The design criteria of PLA are divided into three parts: mandatory, important, and optional criteria.

5.1.1 Mandatory criteria

Compatibility

Because PLA simply adds its own header to an IP packet using a standard extension mechanism, it is compatible with existing IP networks. Routers which do not support PLA can forward PLA packets simply based on IP header information. In such cases, traffic would continue as usual but features of PLA would not be utilized. PLA can also be used together with other IP based protocols and security solutions like IPSec or HIP.

In a hypothetical situation, a router could add a new header on top of the PLA header (i.e., between the IP header and the PLA header). In such case, there are two possible outcomes for the rest of the network. The first alternative is that the packet is forwarded as a plain IP packet, ignoring the PLA header altogether. Alternatively, if routers are intelligent enough, they can retrieve the PLA header from the packet even if there exist other additional headers between the IP header and the PLA header. Thus, they could still use information from the PLA header to check the integrity of the original PLA secured packet. However, in such situations it would not be possible to protect the integrity of additional headers, because these headers would have been added after the PLA header.

Deployability

PLA is compatible with current IP networks, thus it can be deployed gradually. In addition, PLA does not require separate security associations between nodes because all information required for packet verification is included in the PLA header. This is very useful in dynamic network environments such as ad-hoc networks, where the network topology can change frequently. Thus, even if the path along which packets travel changes significantly, every node along the path can fully verify packets if PLA is used.

Misbehaving nodes should be removed quickly from the network

TTP certificates with a limited validity time offer a way to remove misbehaving nodes from the network. If a node has been misbehaving, the TTP will not renew its certificate and without a valid TTP certificate, the traffic that the node sends will be blocked at the first router. In addition, routers can block public keys which are used to send large amounts of invalid packets, thus preventing misbehaving nodes from communicating further.

The validity time of short-time TTP certificates is basically a trade off between security and overhead. A shorter validity time means better security since a potential attacker has a smaller window of opportunity to launch an attack, although it would also increase overhead because certificates would need to be renewed frequently across the whole network. A longer validity time increases the window of opportunity to launch an attack, but it decreases the overhead of the certificate renewal process. The effect of different TTP certificate validity times on security and PLA related overhead is discussed in more detail in Chapter 7. The validity time could be changed on the fly depending on the situation, for example if the network is under constant attack, decreasing the validity time of TTP certificates would make sense. In addition, a mission critical network, like a military network with a limited amount of nodes and a lot of available bandwidth, could use a centralized revocation scheme to instantly revoke certificates of compromised nodes or TTPs.

Validation of packets

The PLA header contains all necessary information to perform validity checks on packets. The signature together with the public key allow detection of forged packets. The timestamp and the sequence number can be used to detect delayed or duplicated packets.

5.1.2 Important criteria

Scalability

The most crucial matter regarding the scalability of PLA is the amount of certificate related traffic. Nodes and TTPs within the network must renew their certificates periodically and their certificates may be revoked, and the amount of this type of traffic should remain reasonable even with a very large number of nodes and TTPs. The effect of this traffic is analysed in the next section in more detail under the topic of manageability.

The TTP architecture presented in Section 3.4 allows TTPs to be added and removed in a flexible manner. In order to limit the overhead of TTP certificate verification traffic, two PLA layers should be used within Internet core networks as mentioned in Section 3.4.2. While using encapsulation and two PLA layers will produce higher bandwidth overhead in core networks, it will significantly decrease the amount of TTPs in which routers must trust. This reduces the overhead produced by TTP verification requests and will thus improve overall scalability. In addition, a router which encapsulates traffic into an own PLA header basically takes also responsibility for that traffic, meaning that routers should only encapsulate traffic in which they fully trust.

The scalability of PLA for a small and portable devices depends on the power overhead produced by PLA and it is discussed below.

5.1.3 Optional criteria

Small power and bandwidth overhead

The overhead caused by PLA can be divided into two main categories: bandwidth overhead, which is caused by the addition of the PLA header to each packet, and power overhead, which is caused by public key cryptography operations like signing and verifying the packets.

Since PLA adds its own fixed-length header to every packet, the average size of sent packets determines the relative bandwidth overhead of PLA. To estimate an average bandwidth overhead of the PLA, network traffic traces from the National Laboratory for Applied Network Research (NLANR) [39] were used to determine an average packet size. Several of the latest traces dated 30th April 2006 were analysed. These traces contained data about 5.6 million packets with an average packet size of 735 bytes or 5880 bits. The size of the PLA header is 89 or 125 bytes, depending on whether a full or a partial TTP certificate is used. If we assume that 10% of packets are sent with a full TTP certificate, then the average overhead produced by PLA would be about 93 bytes per packet. This means that the PLA header adds roughly a 13% bandwidth overhead per PLA layer compared to plain IP traffic. The overhead would increase to 25% in cases where two layers of PLA are used as a result of encapsulation. Such overhead is not critical, considering that the Internet has a lot of excess bandwidth available, especially in core networks [46] where encapsulation would be used. Figure 19 describes the bandwidth overhead of the PLA header with different packet sizes.

The x-axis in the figure indicates the packet size in bytes while the y-axis indicates the overhead percentage. The overhead produced by a full PLA header (including TTP public key and locator) is shown as bold line while overhead produced by a partial PLA header is shown as dotted line. Since PLA adds a fixed length header, relative overhead is larger with small packet sizes and decreases as packet size increases. When packet size is between 500 and 1000 bytes, the relative overhead is roughly 13-25% for a full PLA header and 9-18% for a partial PLA header. An increase in relative overhead occurs with packet sizes of about 1400 bytes and above because adding a PLA header increases the size of a packet above the commonly

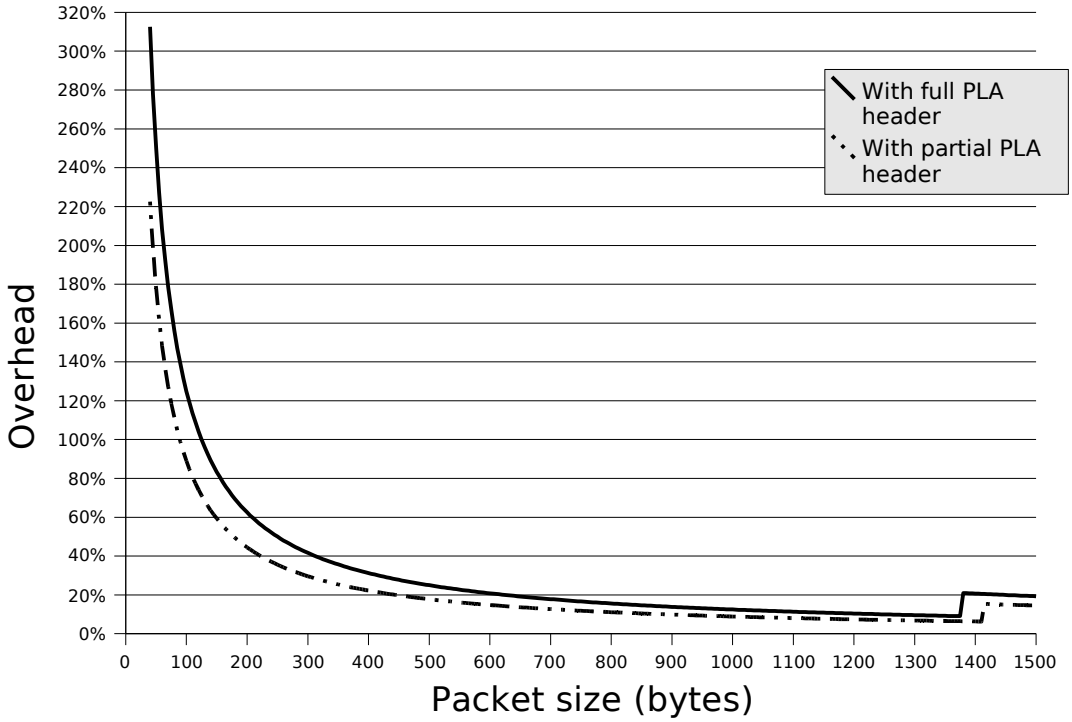


Figure 19. Relation between packet size and PLA bandwidth overhead

used maximum transmission unit of 1500 bytes. Therefore, the packet needs to be fragmented and additional PLA and IP headers are needed. This example assumes that a packet contains only a standard IPv6 header of 40 bytes in addition to the PLA header, if other headers are also present, the overhead produced by fragmentation would slightly increase.

Another important issue is the power consumption of cryptographic operations. Because packets must be verified at every node that handles the traffic, power consumption overhead should be reasonable. As mentioned in Section 4.3.2, a dedicated hardware accelerator based on an ASIC built on a modern manufacturing process would consume less than a couple of watts per Gbps of verified traffic, thus power consumption of PLA is relatively low. In addition, the bandwidth and power efficiency of PLA can be further improved by using larger frame sizes in high speed networks. In the future, the amount of traffic which routers will be able to handle will greatly increase, but at the same time semiconductor technologies will continue to improve. As a result, the overhead produced by PLA does not prevent it from scaling to future high speed networks.

While wireless mobile devices have a very limited computational and battery power, they do not usually handle large amounts of traffic and thus they would not require powerful hardware for handling PLA. In addition, wireless transmission itself requires a non-trivial amount of power. Several power efficient transmission methods for IEEE 802.11a/h wireless LANs were analysed in [41] and the best method achieved a power efficiency of approximately 70 Mbits of traffic per Joule of consumed energy over a transmission range of 10 meters. This is equivalent to a power consumption of about 14 W per Gbps speed or about 86 μJ of energy per an average 6000-bit packet. As the transmission distance was increased to 25 meters, efficiency dropped to about 9 Mbit of traffic per Joule corresponding to 111 W power consumption per Gbps or to 667 μJ of consumed energy per packet. As a comparison, the power consumption of the Hardcopy ASIC PLA cryptographic accelerator would be significantly lower, around 5 W per Gbps or 30 μJ per packet, while the power consumption of a customized ASIC manufactured on a modern manufacturing process would be even lower.

The power consumption of cryptographic operations is also insignificant when compared to the battery capacities of ordinary cell phones. A standard cell phone battery rated for 3.7 V operating voltage and 1000 mAh capacity contains about 13320 Joules of energy. Even with the power efficiency of a cryptographic accelerator based on the Hardcopy ASIC, this amount of energy would be enough to validate 444 million packets, which would correspond to 333 Gigabytes of average-sized traffic. With a 1 Mbps network connection it would take one month of continuous usage to transmit such amount of data. Thus, the power overhead produced by PLA is not significant in comparison to the overall power consumption of wireless transmission and in comparison to the overall power consumption of many mobile devices like cell phones. Finally, the signature generation that needs to be performed by a node that sends packets requires less computational resources and power than the signature verification performed by routers and by the destination.

Even thought PLA produces some power and bandwidth overhead, it may achieve better power efficiency in mobile networks in certain cases. For example, mobile ad-hoc networks are very vulnerable to various attacks such as DoS or route spoofing. A successful attack can quickly drain the whole ad-hoc network of its resources like a

battery power or bandwidth. Thus, if the ad-hoc network is frequently attacked, PLA can actually save power in the network by limiting attacks quickly, before they can bring the whole network down by consuming its all available energy.

Free of patents

Various implementation of ECC algorithms are covered under several patents. Most ECC related patents are owned by the Canadian company Certicom, which claims to have over 300 issued and pending patents related to ECC and public key cryptography. Most of ECC related patents are set to expire by the year 2020, thus for now PLA implementation is protected by patents and does not satisfy this requirement. While PLA is not dependent on the used cryptographic solution, ECC is currently the only feasible solution for PLA because ECC can achieve a strong security with relatively small key and signature sizes. Thus, it would be necessary to license ECC related patents for a commercial PLA implementation.

Overall, PLA satisfies its design requirements well. Mandatory criteria are fully satisfied. The scalability and low power overhead requirements are satisfied quite well and these can be further improved in the future. The free of patents requirement is the only requirement that cannot be satisfied until ECC related patents expire.

5.2 Redesigning the Internet criteria analysis

This Section contains an analysis how PLA satisfies the requirements for the next-generation Internet which were presented in Section 2.5. Some of these requirements are equivalent to those presented in the previous section and thus they will not be covered here again.

Only valid packets are transmitted in the network

PLA fully satisfies this requirement, every node that forwards the packet checks the integrity of the packet using the information contained in the PLA header. Only unique, authentic packets will be sent forward. Delayed, forged, or duplicated packets are discarded immediately.

Every packet has an owner and all packets originate from trusted entities

PLA satisfies this requirement by including information for calculating a public key of the sender in every packet and by using trusted third parties. Since every packet is cryptographically signed by the sender of the packet and contains the sender's public key, the sender is not able to deny sending the packet. Thus, every sent packet in the network has an owner which can be traced. The aim of the trusted third parties is to guarantee that the sender is really a valid and trusted entity in the network. Therefore, untrusted entities will not be able to send data to the network.

Prioritizing traffic

PLA allows traffic to be divided into different groups which can be prioritized as necessary. These traffic groups are: valid data traffic from valid users, valid data traffic from unverified users (the TTP which has authorized the user has not been fully verified yet), and valid signalling traffic. A router could reserve a small amount of bandwidth for the latter two categories and allocate a majority of available bandwidth for valid traffic originating from valid users.

A TTP certificate mechanism could be extended further to provide more diverse rights in addition to current signalling and full traffic rights. For example, there could be several traffic levels defined in the rights field of a TTP certificate. Ordinary users of the network would obtain rights to send data on normal priority while critical systems and authorities would get rights to send data at higher priority. This would enable the network to prioritize traffic more effectively. In case of a serious emergency when only a small amount of bandwidth is available, routers could delay or even discard low-priority packets.

Using this principle, PLA could be used to implement a more secure version of Virtual LAN (VLAN) [24] technology. The main aim of VLAN is to make network management easier by allowing LANs in different physical locations to be grouped into virtual LANs. VLAN adds a separate tag to the Ethernet frame containing a VLAN identifier and a priority level of the frame. The downside of the current VLAN solution is poor security; any router can change the value of the VLAN tag

within the Ethernet frame, potentially disrupting the network. Since the VLAN tag is included in the Ethernet frame, network layer security solutions like IPSec cannot be used to protect it.

In the scope of VLAN, PLA could help as follows. Each virtual LAN would be managed by a separate TTP and the TTP's public key would act as a VLAN identifier. Since the TTP's public key and locator are included in a full TTP certificate, a separate VLAN identifier would not be necessary. Priority information of the frame could be expressed using the rights field of the TTP certificate. Basically, all VLAN related information would be included in a PLA header, and thus it would be protected against forgery by the packet's signature. Such a mechanism would also prevent unauthorized access to the VLAN network since all users wishing to use the network would need to receive a valid TTP certificate from the TTP that manages the VLAN.

Manageability

The TTP certificate mechanism presented in Section 3.4 allows for efficient management of a large number of nodes. The TTP certificate includes a validity time and several levels of rights and it is also possible to specify which rights can be delegated forward. Section 3.5 presents a mechanism for bootstrapping and managing various devices, including devices which lack traditional input methods.

Since users will continuously renew their TTP certificates, they will use the TTP's bandwidth and computational resources. However, the load produced on the TTP is not significant. Let's assume that a major TTP would have around one million customers. If each customer requests a new TTP certificate every hour, then there would be roughly 278 certificate requests per second, which is not a high amount of requests to handle for a large server. The bandwidth overhead would not be significant either. A certificate request and reply would each require a single packet, consuming roughly 3000 bits of bandwidth in total (2000 bits for two PLA headers and 1000 bits for other information including a new TTP certificate). Thus, 278 requests per second would consume only 834 kbps of bandwidth.

Controlling incoming connections

PLA together with traditional certificates can be used to control incoming connections. The recipient of a connection could grant certificates to trusted initiators allowing them to make incoming connections. PLA is used to enforce that only the data from valid initiators is allowed to be transmitted to the recipient. Section 6.2 contains a more detailed discussion about using PLA to control incoming connections.

Privacy protection

Including information about the sender's public key together with an identity number given by a TTP in every packet naturally produces a privacy risk, since the sender of the packet can be easily tracked in the whole network using an unique public key and identity received from a TTP. However, this problem can be solved by using multiple public keys for each user which will act as pseudonyms. For example, the user can generate a large amount of public keys and request short-time TTP certificates for these keys from the TTP using his main public key which is already authorized by the TTP. After the user makes a connection using his new public key, he will be practically anonymous to the rest of the network. The user's TTP will be the only entity in the network that will know the mapping between the user's real identity and all of his public keys. Thus, privacy will not be a problem as long as the user changes his used public key frequently and as long as an above mentioned TTP certificate retrieval processes is protected by encryption. Such a process could also be automated in software to make it more easy to use for the user. The user's key management software could be configured to periodically generate a new public key, request a TTP certificate for it, and use it for future traffic without any user interaction.

Another option to enhance privacy and provide flexibility is to utilize several public keys simultaneously for performing different tasks. For example, the user could use one public key to read work related e-mail, while using another for instant messaging with his friends. Thus, with the help of PLA, it is possible to produce a good trade-off between the security and privacy. If the user misbehaves, the authorities can

determine the real identity of the user by contacting the TTP that has authorized his public keys. Otherwise, the user will be able to maintain reasonable privacy[1].

One problem with privacy in the current Internet is the lack of trust during anonymous communication. It is hard to communicate effectively while retaining complete anonymity because other parties on the Internet do not have the means to verify effectively whether that anonymous person can be trusted or whether the data is really coming from the same anonymous person or from someone else that impersonates him. PLA provides a help to this issue because it allows for users to have several pseudonyms that can be validated by other parties. For example, suppose that A and B want to communicate with each other. A wants to retain his anonymity, thus he discloses one of his numerous public keys together with the associated TTP certificate to B without disclosing any personal details. Now B can contact A's TTP and verify that A is really a valid entity and his TTP certificate is still valid. As a result, B can trust A without knowing A's real identity. Since A has numerous public keys in use, he can allocate one for communication with B and A will not need to change that key frequently. In addition, by checking the signatures of received packets, B can be sure that those packets are coming from A and not from some other party that impersonates A.

1 It is important to note that a network layer solution like PLA cannot offer a complete privacy solution by itself. Network interface cards contain a unique MAC address which is included in the Ethernet frame and thus offers a way to trace the user. Thus, also other methods must be used with PLA to provide a good privacy.

6. Applications of PLA

The main aim of PLA is to improve the security of users, services, and the network by protecting them from several kind of attacks. However, PLA also has some other uses. For example, it can be used for controlling incoming connections and for billing purposes.

6.1 Securing the network infrastructure

PLA can be used to secure the network infrastructure because of its ability to immediately detect and react against different kinds of attacks. A node that receives a forged, duplicated, or delayed packet can discard the packet immediately, preventing this packet from consuming resources in the rest of the network and stopping a possible attack quickly. In addition, since the PLA header contains information for calculating a sender's public key which is certified by some trusted third party, the attacker can be traced and caught easier compared to current solutions. The applicability of PLA for the prevention of different kinds of attacks is discussed below.

Denial-of-service attacks

PLA can protect the network against denial-of-service attacks in several ways. The main advantage of PLA is that it can detect and discard invalid packets immediately. In a classic DoS attack, the attacker sends a large amount of packets to its victim. If such an attack is detected and PLA is used, routers along the path can block the attacker's public key and therefore stop the attack. The victim who has received a lot of invalid packets could express that he does not want to receive any traffic from the attacker by sending a control message containing the attacker's public key back to the attacker. Routers on the way would take note of such control message and depending on the policy, routers could either block all traffic from the attacker or merely prevent the attacker from sending traffic to the victim (i.e., prevent a source with public key X from sending packets to destination Y).

A denial-of-service attack can also be launched by attacker who resides in the middle of the network and makes a large amount of copies of valid packets. An example of such an attack is shown in Figure 20 below.

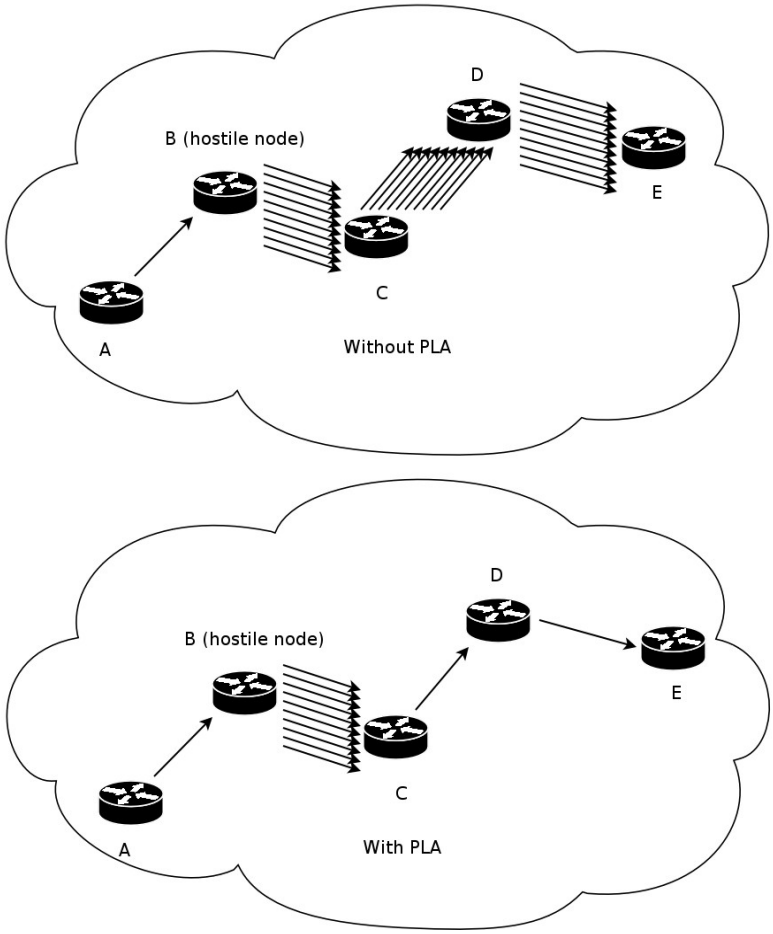


Figure 20. Preventing a denial-of-service attack with PLA when the attacker floods the network with copies of valid packets

In this example case, node A sends a packet to node E through nodes B, C and D. Node B is a hostile node which makes a large amount of copies of each packet that passes through it, thus flooding the network with useless packets. Without PLA, nodes C and D would simply forward those duplicated packets. Even if a traditional security solution like IPSec is used, nodes C and D can not determine that the extra packets are just copies of the original, because they cannot look inside an encrypted IPSec packet. If PLA is used, node C will notice that extra packets are just copies of the original by looking at the timestamp and sequence number fields in the PLA header. Thus, node C can drop those extra packets immediately before the can consume resources in the rest of the network.

PLA also offers protection against DoS attacks in other ways. Public keys of nodes that are participating in the attack can be blocked by routers, preventing these nodes from continuing the attack and reducing the load on the router. In addition, routers can prioritize traffic based on rights presented in the TTP certificate in order to make sure that really critical packets will get through despite an attack.

Phishing attacks

PLA helps protect against phishing attacks in several ways. First, combining PLA-level encryption keys with application level identities would offer protection against identity thefts which are often used in a phishing attack, since the attacker would be unable to send packets in the victim's name without gaining access to the victim's private key. Second, since a public key is included in every packet, a recipient can check the validity of packets that it receives from, e.g, a WWW server. The recipient can also check whether the server's public key is trusted by a reliable TTP. This decreases the probability of a successful phishing attack, since the attacker participating in phishing attack would eventually lose its TTP certificate.

Spoofing-related attacks

PLA includes information about a sender's public key in every packet along with a signature that is generated using the sender's private key. This offers good protection against various spoofing attacks. The attacker is unable to send spoofed packets in the victim's name and the attacker will also be unable to launch an attack by taking a valid packet and changing some fields in it because such modification would break the signature of the packet.

Replay attacks

Replay attacks can be easily detected with PLA, since the PLA header contains a time stamp and a sequence number. Thus, packets which are duplicated or significantly delayed will be detected and discarded.

6.2 Benefits of PLA in a real life security solution

In this section, we will revisit an Internet banking security example which was presented in Section 2.3.1 to discuss what benefits PLA could provide in a such a situation. In order to provide maximum security and reliability, the original proposal contained several layers of security solutions. With the help of PLA, it is possible to reduce the number of security layers as presented in Figure 21.

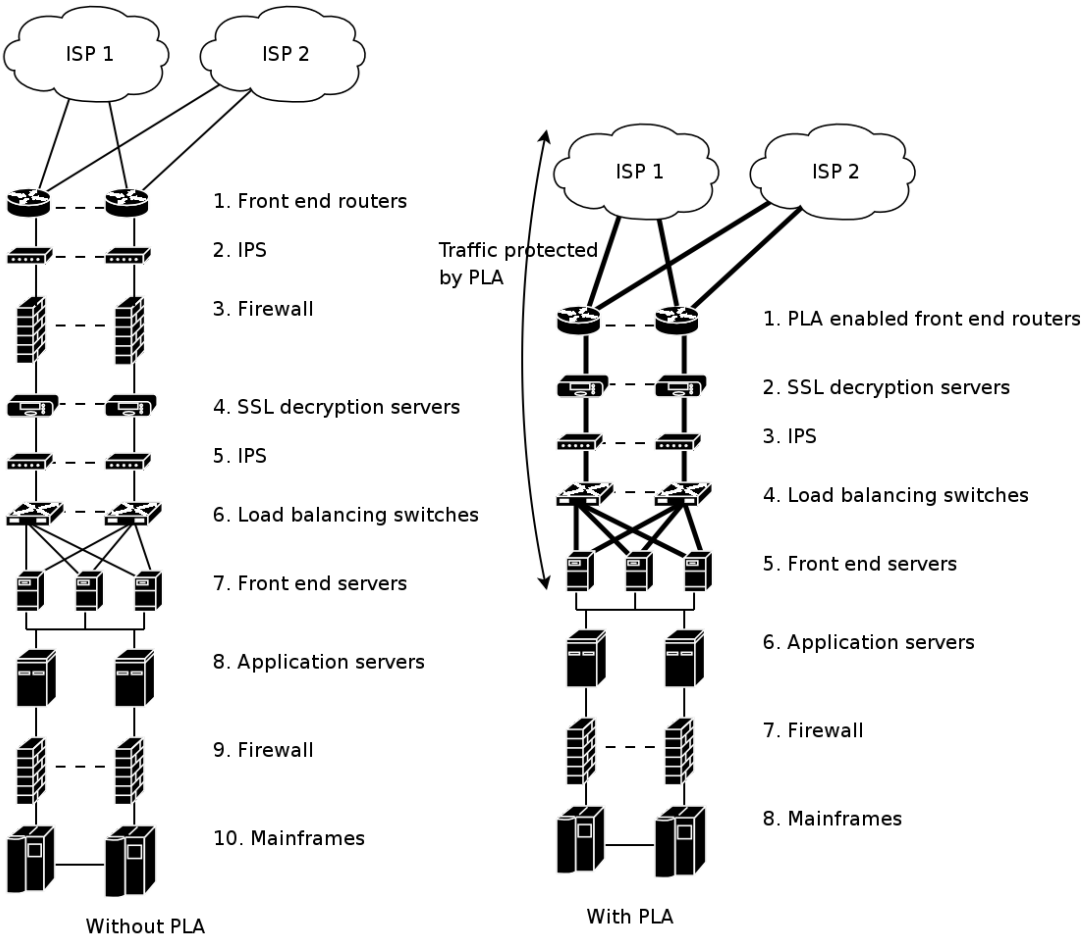


Figure 21. An example of a security solution for Internet banking utilizing PLA

The original proposed security solution had five security related layers before actual application related servers: front end routers, two Intrusion Prevention Systems (IPS), firewall, and SSL decryption servers. Having PLA enabled front end routers could simplify the solution by removing the need for a separate IPS and a firewall. These front end routers would check the authenticity of incoming packets and the validity of users and let only valid packets from trusted users through. Because SSL decryption

is a resource intensive operation, having dedicated servers to handle decryption tasks is preferred in any case. In this example, PLA-enabled traffic would be transmitted all the way until front end servers, which would then remove the PLA header from the incoming traffic and send plain IP packets forward to the application servers.

To further improve security, the public key of the PLA header could be utilized during the user authentication process of an online banking session. When obtaining an online banking account, a user would present his public key to the bank and the user's username/password combination would be usable only if it is accompanied by the user's public key in the PLA header. This would provide additional security, since even if a hostile party is able to capture the username/password combination, it would not be able to gain access to the victim's bank account without having access to the private key of the victim. Such a system would naturally support delegation of rights as presented in Section 3.5.2, thus a user could temporarily transfer his rights to another computer and use it to access his bank account.

6.3 Controlling incoming connections

One way to solve the problem of unwanted incoming connections presented in Section 2.5 is to block all incoming connections to the recipient that are not explicitly allowed. Such blocking can be naturally done in a personal firewall, but in that case incoming connection attempts would still consume resources in the recipient's access network. Thus, the network should provide means to block unwanted incoming connections already at the access network level, before incoming connections even reach the destination. Such blocking would also make it much more difficult to launch DoS attacks against the recipient or the recipient's network.

PLA together with a traditional certificate mechanism can be used to solve this problem [33]. Such a method would work as follows, the recipient of the connection grants explicit certificates to trusted initiators to initiate incoming connections to the recipient. PLA is used to guarantee on the packet level that connections to the recipient really originate from trusted initiators. An example of such a system is shown in the figure below.

The system presented in Figure 22 consists of four main parts. The initiator is the party that initiates the connection. The proxy is an entity in which the recipient trusts. The task of the proxy is to give certificates to trusted initiators for making incoming connections to the recipient. The proxy also keeps track of the recipient's IP address

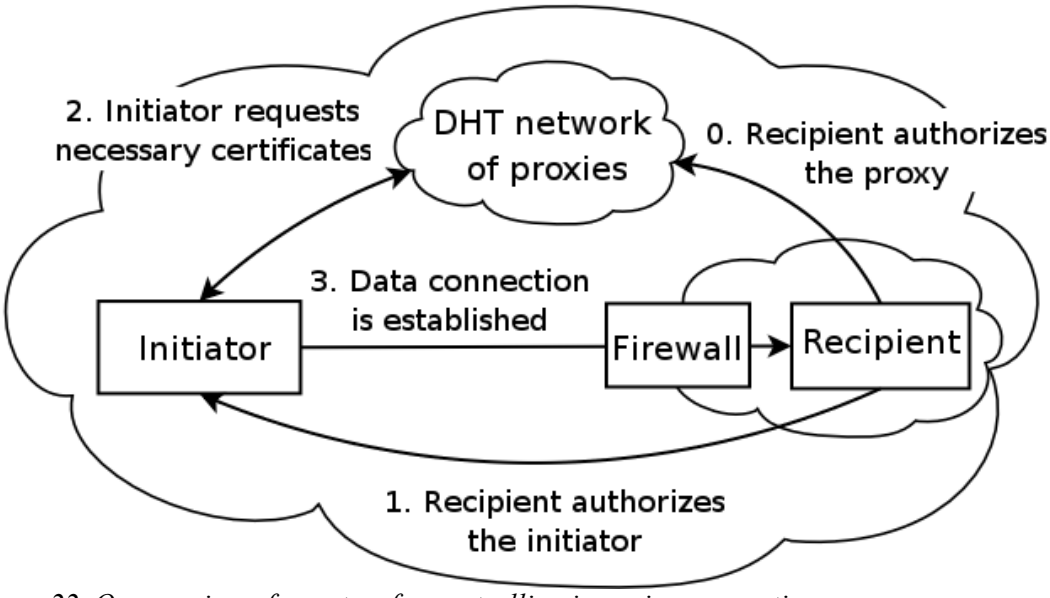


Figure 22. On overview of a system for controlling incoming connections

if the recipient is changing networks. In order to eliminate a single point of failure, proxies form a Distributed Hast Table (DHT) [23] network. The firewall is located in the recipient's access networks. It takes notice of certificates that are passing through it and the firewall blocks all incoming connections to the network unless the recipient is a valid entity within the network and the incoming connection to the recipient has been explicitly allowed via certificates.

In the very beginning of the example, the recipient must authorize the proxy using a certificate. The recipient also authorizes a trusted initiator by giving him a certificate that certifies the initiator's public key, this certificate exchange can be also carried off-line and it is shown as step 1 in the figure. After the initiator has received a certificate from the recipient, the initiator can contact the proxy to request all necessary certificates for making an incoming connection. Exact details of those certificates are not covered here, but they include a certificate that was given by the recipient to the proxy in step 0, a certificate from the recipient to the initiator, and a certificate from the firewall (i.e., an access network) to the recipient. In addition, the

proxy reports the recipient's current IP address to the initiator. In the last step, the initiator first sends all those certificates to the recipient using a control message and afterwards the data connection can be established. The firewall checks that the certificates form a valid certificate chain: firewall => recipient => proxy => initiator. Such a chain shows that the recipient, which has a right to use the access network, has authorized the proxy that has authorized the initiator. If this certificate chain is in order, the firewall will allow the initiator to establish a connection to the recipient. Naturally, revocation and delegation of certificates for making incoming connections is also supported under such a system.

In a such application, PLA is necessary to ensure that the data sent to the recipient is really coming from a trusted initiator. The initiator's public key inside the PLA header of data packets must match the initiator's public key present in a certificate chain, only in that case will the firewall let traffic through. Thus, a malicious party will not be able to make incoming connections by using captured certificates that have been granted to other trusted initiators.

6.4 Other applications

Timestamp and sequence number fields in the PLA header can be utilized for various billing purposes. Since the sequence number of the packet increases monotonically, it can be used for per-packet billing. If the sequence number is increased by the size of the packet, per-traffic billing would also be possible. Such an application would be relatively simple to implement; the operator would occasionally record the sequence number of packets that are sent by its clients and compare it to previously stored values. Such a feature would also be quite secure, since each packet is signed by the client's private key the client cannot deny sending packets. In addition, if the client tries to cheat by decreasing its sequence number, then the packet will not pass the standard validity check and will not reach the destination. It would also be impossible for a hostile party to send data in the client's name without having access to the client's private key.

Using PLA for billing purposes would also provide flexibility. For example, a user visiting his friends with his own laptop could utilize friend's Internet connection

using with his credentials and thus he would be pay for all used traffic. In addition, the operator could sell users rights to send only a specific amount of data to the network. After the user reaches this limit, he would need to purchase rights for additional traffic. By utilizing the above mentioned method for controlling incoming connections, the operator could also charge for the incoming traffic on a per-packet or per-traffic basis.

A timestamp which is present in the PLA header could be utilized to create time-dependent tariffs. For example, the operator could offer cheaper bandwidth at night time when overall traffic usage is low. Since the timestamp is actually a 32-bit UNIX timestamp, it also contains information about the date when the packet is sent. This even offers the possibility for creating seasonal tariffs.

7. Discussion and future work

This Chapter discusses various aspects of PLA and outlines future work.

PLA is flexible in terms of dealing with attackers. Routers which have received a high amount of invalid traffic can either block all traffic from a misbehaving node or simply block the traffic going from a misbehaving node to a certain destination. Furthermore, the rights of nodes participating in the attack can be revoked either permanently or temporarily by the TTP depending on the policy. For example, if the misuse is serious, the TTP may require the user to contact the TTP offline before a new certificate is granted.

By including information for calculating a sender's public key in every sent packet, PLA supports traceability thus making it easier to catch attackers. However, in many current attacks, attackers are using a large amount of compromised nodes (a botnet) to carry out a distributed denial-of-service attack for example. In such cases, real attackers may not necessarily be caught even if PLA is used. However, from the network's point of view, PLA still accomplishes it goals and protects the network infrastructure by providing means for identifying and restricting entities participating in the attack.

The TTP certificate system used by PLA is also flexible and there are several options regarding certificate validity times. One possible way to further improve security is either to make normal TTP certificates revocable or reduce their validity time. We evaluate two alternatives for this. A first alternative would be to keep the current validity time of hours, but make certificates revocable (thus other routers should verify the revocation status of the certificate). A second alternative would be to use irrevocable traffic certificates with a validity time of only a few minutes. Since the validity time is so short in this case, good security can be achieved without a support for revocation of such certificates. The bandwidth required for requesting a new certificate and checking the revocation status of a certificate is roughly equivalent, in both cases two packets are sent containing a TTP certificate and some additional information. In a first case, nodes with whom the user is communicating must contact

his TTP to check the revocation status of the user's TTP certificate, while in a second case, the user must contact his TTP frequently to renew his certificate. Thus overall, the load on TTPs is quite similar. However, in the first case where the TTP certificate has a longer validity time and can be revoked, the user can more easily attempt to attack his TTP by sending a large amount of garbage data to a large number of different recipients. Those recipients in turn would verify the validity of the user's TTP certificate and thus create a load on the user's TTP.

In order to offer protection against replay attacks, PLA-enabled routers should check the timestamps of packets and only forward packets which have not been significantly delayed. Thus, PLA assumes that all participating nodes have their clocks synchronized. While the clock synchronization can be managed easily with the Network Time Protocol (NTP) or a similar solution, in certain cases the clock synchronization requirement may cause some problems. A node might have an incorrect clock and it will not be able to send data to the network, and therefore will also be unable to synchronize its clock. One way to overcome this problem is to allow limited traffic using certain protocols (such as NTP) even if the sending node does not have a properly synchronized clock. Use of encapsulation would also mitigate this problem since only the router in the access network would see the incorrect timestamp. Afterwards, the router would encapsulate the packet in its own PLA header containing a valid timestamp.

Overall, PLA transforms the network security problem into a management problem. PLA offers a mechanism how each packet in the network and each sender in the network can be validated independently by every node. In order to take full advantage of this mechanism, a security policy which determines the best courses of action in various situations is needed. For example, important policy related questions include: how the router should handle packets whose senders have been authorized by unknown TTPs? How much traffic should the router reserve for different TTP certificate types? Should every packet be checked at every router, or is it enough to randomly check some packets? The answers to these question will depend on the network's usage (e.g., civil and military networks would have very different security policies) and other circumstances like whether the network under attack.

7.1 Future work

PLA can be improved in several ways. The current PLA implementation is a proof of concept implementation which is not optimized for the best performance. There are several ways how the performance could be improved. First, PLA-related packet handling can be optimized at the OS level. Currently, each received packet is sent to the hardware accelerator separately and this causes additional latency. Sending several packets at once for verification to the hardware would improve the situation. Furthermore, the PLA header verification may be further optimized by changing the order of various header fields and using padding to word-align them. Finally, the hardware accelerator can be improved by adding a support for accelerating signature generations and transferring the design to more modern and faster FPGAs.

Currently, a full TTP certificate contains a TTP public key and its locator which is a 128-bit IPv6 address. Such a separate locator is necessary for contacting the TTP in the current solution. A more efficient solution could utilize a DHT overlay network. The public key of the TTP could be used as its DHT key and a DHT network would be used for contacting TTPs, decreasing the space requirements of the TTP certificate. The downside of such solution would be increased latency when contacting TTPs.

One important issue to solve is the need for secure and efficient reporting of misuse within the system. If some party is under attack in the network, it should be able to report the attack to the TTP which has authorized the parties that are participating in the attack. However, such reporting should be done in a secure and verifiable way. The party reporting about the attack should be able to prove that the attack has indeed happened and that the specific attacker has participated in it. Otherwise, it would be possible to launch a denial-of-service attack against a victim by making false claims to the victim's TTP that the victim has participated in an attack.

Since the PLA header already includes information for calculating the sender's public key, PLA is suitable to be used together with Cryptographically Generated Addresses (CGA). A PLA packet along with an CGA IP address would provide proof that the sender has a right to use his IP address. Thus, there would not be a need for a

separate IP address negotiation when entering the network and IP address spoofing would be much more difficult to achieve.

PLA could also be combined with the Host Identity Protocol (HIP). In such case PLA would guarantee the integrity of packets while HIP would provide confidentiality and support for mobility and multihoming. Combining HIP with PLA could also simplify HIP. Since PLA can validate the authenticity of the packet, the 4-way base exchange mechanism of HIP may not be necessary. PLA and HIP headers could also be merged into a single header to reduce bandwidth overhead.

Additional uses of PLA can also be researched and improved further. A mechanism for controlling incoming connections presented in Section 6.3 describes how unwanted connections can be blocked in the recipient's access network before they reach the recipient. This method can be extended further in a such a way that blocking unwanted connections would already occur in the initiator's access network. Basically, this would mean that no data connection would be allowed outside the initiator's access network unless the initiator of the connection had already received an explicit permission to send data from the recipient. Thus, unwanted connections would be blocked quicker, before they would be able to consume resources outside the initiator's access network. Such a system would require that a limited amount of signalling traffic would be allowed outside the initiator's access network for rights negotiation.

8. Conclusions

This thesis introduced Packet Level Authentication, a novel way to secure the network infrastructure. PLA aims to provide availability and the fundamental principle of PLA is that every transmitted packet in the network has an undeniable owner and every node within the network can verify the authenticity and validity of the packet independently, without trusting other nodes. PLA guarantees that the packet has not been modified, has not been delayed and is not a duplicate of another packet. This gives PLA an advantage compared to traditional end-to-end security solutions in which only end points of the connection can verify validity of the packet.

PLA is based on public key cryptography and the main benefit of PLA is that attacks against the network can be detected immediately. Thus, attacks can be confined before they can cause significant damage to the network, and actions against attackers can be taken swiftly. PLA is compatible with existing IP networks and can be used together with other security solutions like HIP or IPSec.

Public key cryptography consumes a significant amount of computational resources, but our work shows that PLA is scalable to high speed networks and low power devices as long as a dedicated hardware accelerator is used for cryptographic operations.

In addition to improving network security, PLA can also be used for several other purposes, such as controlling incoming connections and billing.

References

[1] Aboba, B. *et al.* Extensible Authentication Protocol (EAP). The Internet Society, Network Working Group, Request for Comments: 3748. 2004.

[2] Altera. HardCopy Structured ASICs: Technology for Business [online]. Available from: http://www.altera.com/products/devices/hardcopy/hrd-index.html [Accessed 21st May 2008].

[3] Anti-Phishing Working Group. Phishing Activity Trends Report - July 2007 [online]. Available from: http://www.antiphishing.org/reports/apwg_report_july_2007.pdf [Accessed 22nd May 2008].

[4] Arends, R. *et al.* DNS Security Introduction and Requirements. The Internet Society, Network Working Group, Request for Comments: 4033. 2005.

[5] Atkins, D. *et al.* Internet Security. New Riders, pp. 257-316. 1996.

[6] Atkins, D. and Austein, R. Threat Analysis of the Domain Name System (DNS). The Internet Society, Network Working Group, Request for Comments: 3833. 2004.

[7] Aura, T. Cryptographically Generated Addresses (CGA). The Internet Society, Network Working Group, Request for Comments: 3972. 2005.

[8] Barker, E. *et al.* Recommendation for Key Management – Part 1: General. National Institute of Standards and Technology, NIST Special Publication 800-57, March 2007.

[9] Brumley, B. Efficient three-term simultaneous elliptic scalar multiplication with applications. In Viiveke Fåk, editor, Proceedings of the 11th Nordic Workshop on Secure IT Systems--NordSec '06, pp 105-116. Linkoping, Sweden, October 2006.

[10] Brumley, B. Left-to-right signed-bit τ-adic representations of *n* integers (short paper). In Prodeecings of Information and Communications Security, 8th International Conference - ICICS '06. December 2006.

[11] Brumley, B. and Järvinen, K. Koblitz Curves and Integer Equivalents of Frobenius Expansions. Revised Selected Papers of the 14th Annual Workshop on Selected Areas in Cryptography, SAC 2007, pp. 126-137. Ottawa, Canada, August 2007.

[12] Brumley, B. and Nyberg, K. Differential properties of elliptic curves and blind signatures. In Proceedins of Information Security, 10th International Conference - ISC '07, volume 4779 of Lecture Notes in Computer Science, pp. 376-389. Springer-Verlag, 2007.

[13] Cam-Winget, N. *et al.* Security flaws in 802.11 data link protocols. Communications of the ACM, Volume 46, Issue 5. May 2003.

[14] Candolin, C. Securing Military Decision Making In a Network-centric Environment. Doctoral dissertation, Espoo, Finland, 2005.

[15] CERT-FI. CERT-FI yhteydenotot nimikkeittäin [online]. Available from: http://www.cert.fi/katsaukset/tilastot.html [Accessed 17th May 2008].

[16] Chokhani, S. and Ford, W. Internet X.509 Public Key Infrastructure - Certificate Policy and Certification Practices Framework. The Internet Society, Network Working Group, Request for Comments: 2527. 1999.

[17] Clark, D. The design philosophy of the DARPA internet protocols. ACM SIGCOMM Computer Communication Review, Symposium proceedings on Communications architectures and protocols SIGCOMM '88, Volume 18 Issue 4. 1988.

[18] Daley, W. and Kammer, R. Federal Information Processing Standards Publication 186-2: Digital Signature Standard [online], January 2000. Available from: http://www.governmentsecurity.org/articles/articles2/fips-186-2.pdf_fl/ [Accessed 21st May 2008]

[19] Deering, S. *et al.* Internet Protocol, Version 6 (IPv6) Specification. The Internet Society, Network Working Group, Request for Comments: 2460. 1998.

[20] Dierks, T. The TLS Protocol Version 1.0. The Internet Society, Network Working Group, Request for Comments: 2246. 1999.

[21] Ephremides, A. *et al*. A design concept for reliable mobile radio networks with frequency hopping signaling. Proceedings of the IEEE, Volume 71, Issue 1. January 1987.

[22] Gleeson, B. *et al*. A Framework for IP Based Virtual Private Networks. The Internet Society, Network Working Group, Request for Comments: 2764. 2000.

[23] Gribble, S. D. *et al*. Scalable, Distributed Data Structures for Internet Service Construction. In proceedings of the 4th Symposium on Operating System Design and Implementation (OSDI 2000), pp. 319-332, 2000.

[24] IEEE Computer Society. IEEE Standard for Local and metropolitan area networks - Virtual Bridged Local Area Networks. May 2006 [online]. Available from: http://standards.ieee.org/getieee802/download/802.1Q-2005.pdf [Accessed 20th May 2008].

[25] Järvinen, K. and Skyttä, J. High-Speed Elliptic Curve Cryptography Accelerator for Koblitz Curves. In Proceedings of the 16th IEEE Symposium on Field-programmable Custom Computing Machines, FCCM 2008. Palo Alto, California, USA, April, 2008.

[26] Järvinen, K. and Skyttä, J. On Parallelization of High-Speed Processors for Elliptic Curve Cryptography. IEEE Transaction on Very Large Scale Integration (VLSI) Systems. In press.

[27] Järvinen, K. *et al*. Efficient Circuitry for Computing tau-adic Non-Adjacent Form. In Proceedings of the 13th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2006, pp. 232-235. Nice, France, December 2006.

[28] Järvinen, K. *et al*. FPGA Design of Self-certified Signature Verification on Koblitz Curves. In Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems, CHES 2007, pp. 256-271. Vienna, Austria, September 2007.

[29] Kent, S. and Atkinson, R. Security Architecture for the Internet Protocol. The Internet Society, Network Working Group, Request for Comments: 2401. 1998.

[30] Koblitz, N. CM curves with good cryptographic properties. In Proceedings of of the 11th Annual International Cryptology Conference on Advances in Cryptology, pp. 279-287. August 1991.

[31] Kobliz, N. Elliptic Curve Cryptosystems. Mathematics of Computation. Volume 48, pp. 203-209, 1987.

[32] Kuon, I. and Rose, J. Measuring the gap between FPGAs and ASICs. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. Volume 26, no. 2, pp. 203-215, February 2007.

[33] Lagutin, D. and Kari, H. H. Controlling incoming connections using certificates and distributed hash tables. In Proceedins of The 7th International Conference on Next Generation Teletraffic and Wired/Wireless Advanced Networking (NEW2AN 2007), pp. 455-467. St. Petersburg, Russia, September 2007.

[34] Menezes, A. Elliptic Curve Cryptosystems. CryptoBytes, Vol.1 No.2, Summer 1995.

[35] Miller, V. Use of Elliptic Curves in Cryptography. In proceedings of the Advances of Cryptology – Crypto '85, Santa Barbara, USA, August 1985.

[36] Moskowitz, R. and Nikander, P. Host Identity Protocol. Internet draft, work in progress, June 2006.

[37] Myers, M. *et al.* X.509 Internet Public Key Infrastructure - Online Certificate Status Protocol (OCSP). The Internet Society, Network Working Group, Request for Comments: 2560. 1999.

[38] Nazario, J. Estonian DDoS Attacks - A summary to date. The Arbor Networks Security Blog [online]. Available from: http://asert.arbornetworks.com/2007/05/estonian-ddos-attacks-a-summary-to-date/ [Accessed 21st May 2008].

[39] NLANR - network traffic packet header traces [online]. Available from: http://pma.nlanr.net/Traces/ [Accessed 20th May 2008].

[40] Packet level authentication [online]. Available from: http://www.tcs.hut.fi/Software/PLA/new/ [Accessed 10th May 2008].

[41] Qiao, D. *et al.* Interference analysis and transmit power control in IEEE 802.11a/h wireless LANs. IEEE/ACM Transactions on Networking, Volume 15, Issue 5. October 2007.

[42] Rajaniemi, A. Verkkopankin toimintavarmuuden turvaaminen tietoverkon näkökulmasta. Master's thesis, Helsinki University of Technology. 2005.

[43] R. Rivest. SEXP (S-expressions) [online]. Available from: http://theory.lcs.mit.edu/~rivest/sexp.html [Accessed 21st May 2008].

[44] Schuba, C. *et al*. Analysis of a Denial of Service Attack on TCP. In Proceedings of the 1997 IEEE Symposium on Security and Privacy, May 1997.

[45] Syverson, P. A Taxonomy of replay attacks. In Proceedings of the 7th IEEE Computer Security Foundatation Workshop (CSFW 94). Franconia, USA, June 1994.

[46] TeleGeography Research. Global Internet Geography Report, Exucutive Summary. [online]. Available from: http://www.telegeography.com/products/gig/samples07/GIG_Exec_Summary.pdf [Accessed 21st May 2008].

[47] Tews, E. *et al*. Breaking 104 bit WEP in less than 60 seconds [online]. Available from: http://eprint.iacr.org/2007/120.pdf [Accessed 22nd May 2008].

[48] Trusted Computing Group. TPM Work Group [online]. Available from: https://www.trustedcomputinggroup.org/groups/tpm/ [Accessed 20th May 2008].

[49] Watson, P. Slipping in the Window: TCP Reset Attacks. CanSecWest conference, 2004.

[50] Zakon, R. Hobbes' Internet Timeline v8.2 [online]. Available from: http://www.zakon.org/robert/internet/timeline/ [Accessed 22nd May 2008].

Appendix A: TTP certificate format

The high level structure of the certificate is presented below in the S-expressions format.

```
<cert>:: "(" "cert" <issuer> <subject> <identity>
<validity> <rights> <deleg> <sig> ")";
```

The certificate is granted by `<issuer>` to `<subject>` and `<validity>` determines the time period when the certificate is valid. Fields `<rights>` and `<deleg>` determine what rights the certificate contains and which rights can be delegated to other parties. Finally, there is an issuer's signature over the certificate. Each field is described in more detail below.

PLA uses identity-based implicitly-certified keys, thus the actual subject's public key is calculated using information present in a TTP certificate as described in Section 4.1, and the subject's public key is not physically included in the certificate.

The "issuer" field

```
<issuer>:: "(" "issuer" <issuer-details> ")";
<issuer-details>:: "(" "pub-key" <pub-key> "locator"
     <locator> ")";
<pub-key>:: <byte-string>;
<locator>:: <byte-string>;
```

The `<issuer>` field consists of two elements, the public key of the issuer and the issuer's locator (e.g., an IP address). If the certificate is issued by a normal user instead of a trusted third party, the locator field is zero.

The "subject" field

```
<subject>:: "(" "subject" <pub-key> ")";
```

Subject's public key is calculated using information of other fields of the TTP certificate.

The "identity" field

```
<identity>:: "(" "identity" <id> ")";
```

```
<id>:: <byte-string>;
```

This field contains a user's identity which is given by a trusted third party. The TTP gives an unique identity to each of its users.

The "validity" field

```
<validity>:: "(" "valid" <not-before> <not-after> ")";
<not-before>:: "(" "not-before" <timestamp> ")";
<not-after>:: "(" "not-after" <timestamp> ")";
<timestamp>:: <byte-string>;
```

The validity field contains timestamps that determine the period when the certificate is valid. Both fields `<not-before>` and `<not-after>` are always present. If the `<not-before>` field is zero, the certificate is valid immediately. The `<timestamp>` uses Unix timestamp format and it contains number of seconds after 1st January 1970.

The "rights" field

```
<rights>:: "(" "rights" <bits> ")";
<bits>:: <string>;
```

The `<rights>` field determines the rights given by the certificate. For trusted third party certificates, there are three different rights expressed as bits:

xx1 – The right to delegate other rights.

x1x – The right to send the data to the network.

1xx – The right to request a new certificate.

The "deleg" field

```
<deleg>:: "(" "deleg" <bits> ")";
```

This field determines which rights can be delegated to other parties. The format is the same as with the `<rights>` field:

xxxx1 – The right to delegate the right to delegate rights.

xxx1x – The right to delegate the right to send data to the network.

xx1xx – The right to delegate the right to request a new certificate.

The aim of the first right is to let the issuer have more control over delegability of other rights. If the “rights” field is in a form of xxxx1 and the “deleg” field is xxxx1 then it means that the rights can be delegated indefinitely forward. If the “rights” field is xxxx1 but the “deleg” field is xxxx0 then the subject can delegate rights to another subject, but this another subject cannot delegate rights forward any more. If both fields are in a form of xxxx0 then it means that no rights can be delegated to other parties. Thus, the issuer can allow rights to be delegated only once or indefinitely or not allow delegation at all.

The “signature” field

```
<sig>:: "(" "signature" <signature> ")";
<signature>:: <byte-string>;
```

The signature field contains the `<issuer>`'s cryptographic signature over the certificate ignoring the `<issuer>` field.

Appendix B: Certificate format for controlling incoming connections

The structure of the certificate is presented below in the S-expressions format.

```
<cert>:: "(" "cert" <issuer> <subject> <validity>
<rights> <deleg> <sig> ")";
```

The certificate is granted by `<issuer>` to `<subject>` and `<validity>` determines the time period when the certificate is valid. Fields `<rights>` and `<deleg>` determine what rights the certificate contains and which rights can be delegated to other parties. Finally, there is an issuer's signature over the certificate. Each field is described in more detail below.

The “issuer” field

```
<issuer>:: "(" "issuer" <issuer-details> ")";
<issuer-details>:: "(" "pub-key" <pub-key> "locator"
     <locator> ")";
<pub-key>:: <byte-string>;
<locator>:: <byte-string>;
```

The `<issuer>` field consists of two elements, the public key of the issuer and the issuer's locator (e.g., an IP address). If the certificate is issued by a normal user instead of a trusted third party, the locator field is zero.

The “subject” field

```
<subject>:: "(" "subject" <pub-key> ")";
```

The subject field contains a public key of the subject.

The “validity” field

```
<validity>:: "(" "valid" <not-before> <not-after> ")";
<not-before>:: "(" "not-before" <timestamp> ")";
<not-after>:: "(" "not-after" <timestamp> ")";
<timestamp>:: <byte-string>;
```

The validity field contains timestamps that determine the period when the certificate is valid. Both fields <not-before> and <not-after> are always present. If the <not-before> field is zero, the certificate is valid immediately. The <timestamp> uses Unix timestamp format and it contains number of seconds after 1st January 1970.

The "rights" field

```
<rights>:: "(" "rights" <bits> ")";
<bits>:: <string>;
```

The <rights> field determines the rights given by the certificate. For trusted third party certificates, there are three different rights expressed as bits:

xxxx1 – The right to delegate other rights.

xxx1x – The right to send the data to the network.

xx1xx – The right to request a new certificate.

x1xxx – The right to create an incoming connection.

1xxxx – The right for session initialization management. This right might be necessary when creating an incoming connection.

The "deleg" field

```
<deleg>:: "(" "deleg" <bits> ")";
```

This field determines which rights can be delegated to other parties. The format is the same as with the <rights> field:

xxxx1 – The right to delegate the right to delegate rights.

xxx1x – The right to delegate the right to send data to the network.

xx1xx – The right to delegate the right to request a new certificate.

x1xxx – The right to delegate the right to create an incoming connection.

1xxxx – The right to delegate the right for session initialization management. This right might be necessary when creating an incoming connection.

The aim of the first right is to let the issuer have more control over delegability of other rights. If the "rights" field is in a form of xxxx1 and the "deleg" field is xxxx1 then it means that the rights can be delegated indefinitely forward. If the "rights" field is xxxx1 but the "deleg" field is xxxx0 then the subject can delegate rights to another subject, but this another subject cannot delegate rights forward any more. If

both fields are in a form of xxxx0 then it means that no rights can be delegated to other parties. Thus, the issuer can allow rights to be delegated only once or indefinitely or not allow delegation at all.

The “signature” field

```
<sig>:: "(" "signature" <signature> ")";
<signature>:: <byte-string>;
```

The signature field contains the signature over the whole certificate with the `<issuer>`'s private key.

Appendix C: Format for certificate requests

```
<request>:: "(" "request" <subject> <validity> <token>
     <sig> ")";
<request>:: "(" "request" <subject> <validity> <sig> ")";
```

The certificate request contains subject's public key and requested validity time. The request may also contain an optional authorization token. Token field is explained below while subject and validity fields are identical to fields mentioned previously.

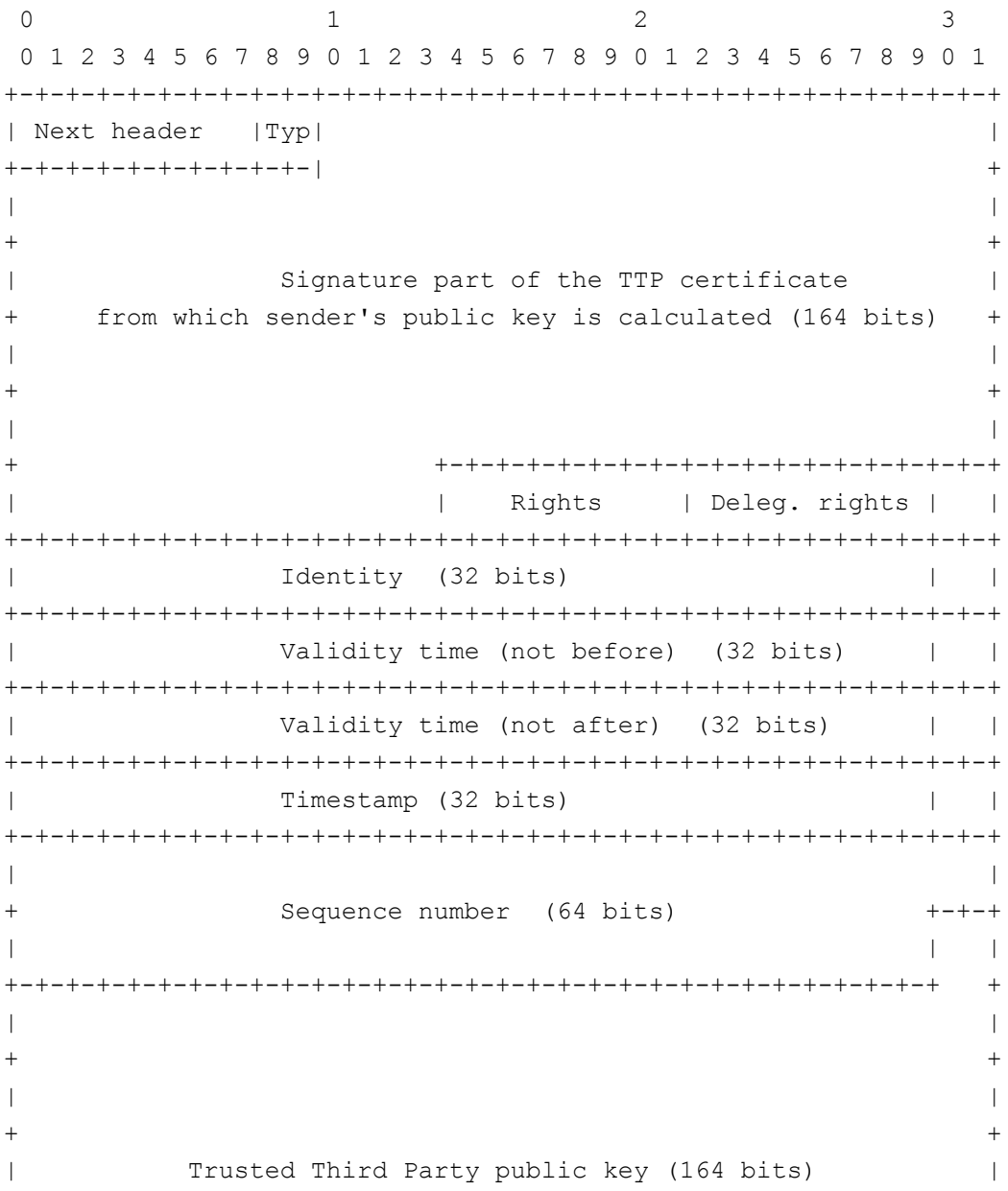
The “token” field

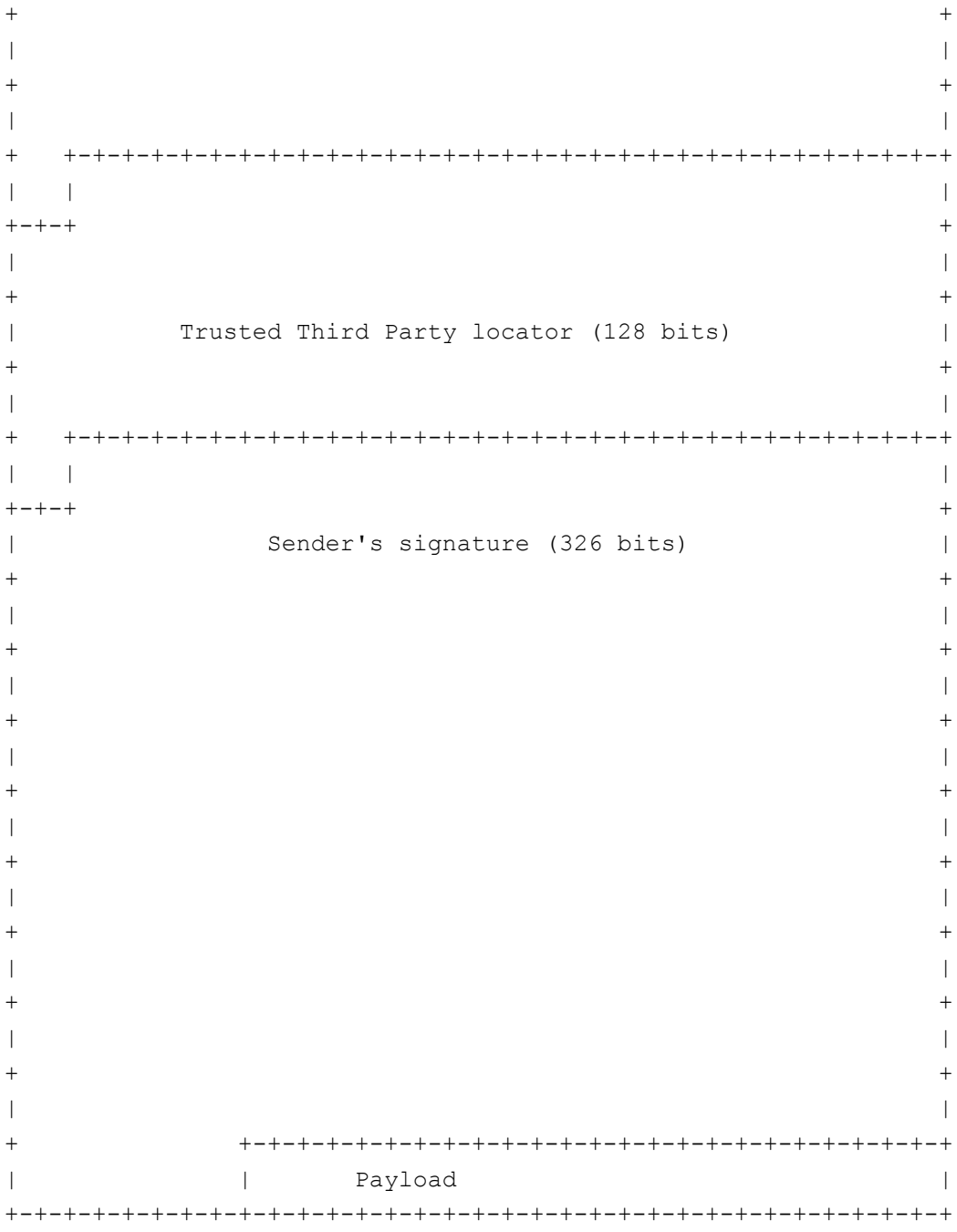
```
<token>:: "(" "token" <t> ")";
<t>:: <byte-string>;
```

This field contains an authorization token which may be present in a certificate request. The aim of the token is to guarantee that the requesting party has a right to request a certificate.

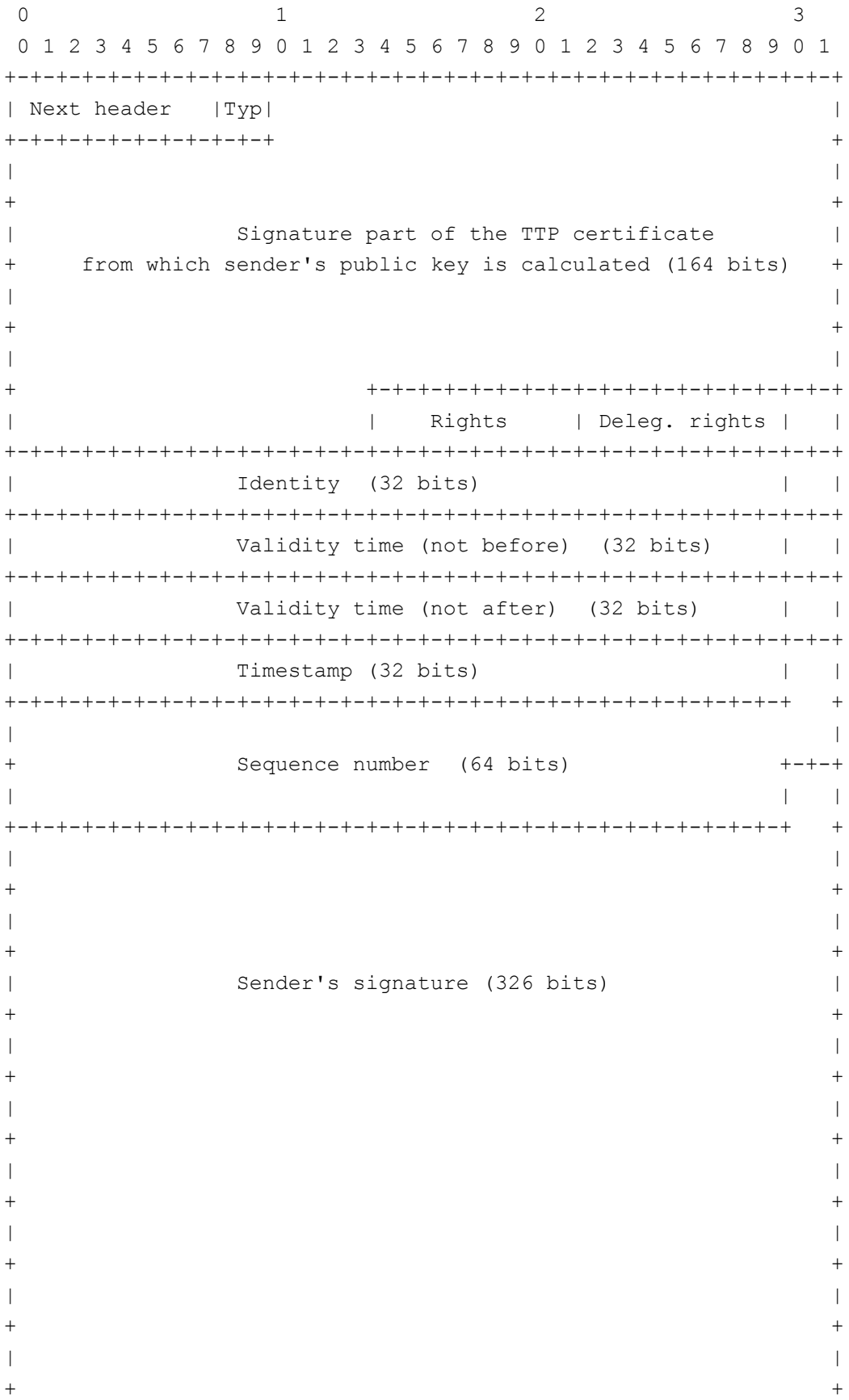
Appendix D: PLA header

The PLA header is added on top of an IPv6 header using an extension header id: 0x09, there was no specific reason for choosing this number, it was just a first id number available. Next header field is a standard field in IPv6 extension header format, it contains an id of the next header. Type field refers to PLA header type, 00 denotes a full PLA header while 01 denotes a partial header without a TTP locator and public key. The size of a full PLA header is 1000 bits (125 bytes) and it is presented below.




Trusted Third Party locator (128 bits)
Sender's signature (326 bits)
Payload

Below is presented a partial PLA header with length of 712 bits (89 bytes).



```
|                                                               |
+       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       |Padding|             Payload                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```