Packet level authentication in military networks

Catharina Candolin, Janne Lundber, Hannu Kari
Laboratory for Theoretical Computer Science
Helsinki University of Technology
PB 5400, FI-02015 HUT, Finland
{firstname.lastname}@hut.fi

Abstract

Some of the worst problems in heterogeneous military networks are related to security. Several solutions have been proposed to ensure that the network is able to perform its tasks, namely to transport the right packets to the right place at the right time, however, the solutions have typically been tailored to specific protocols or made assumptions that are not realistic. In this paper, we present a generic security solution based on packet level authentication and briefly describe our prototype implementation. The architecture can be used for authentication, access control, firewall applications, denial-of-service prevention, and so on. It also secures all routing protocols.

Keywords

Packet level authentication, security, IPv6

INTRODUCTION

An ad hoc network is a collection of nodes that do not need to rely on a predefined infrastructure to establish and maintain communications. Most or all nodes participate in network operations, such as routing and network management, according to their capacity with respect to CPU, memory, and battery power. Ad hoc networks tend to be dynamic in nature, that is, nodes are able to enter and leave the network on frequent basis as well as to move within the network itself. Furthermore, the whole ad hoc network may be mobile.

Future military networks are designed to consist of several heterogeneous ad hoc networks that are connected to each other through a temporary or permanent backbone. The resulting "system of systems" will consist of everything from sensor networks to command and control as well as weapon systems. This interconnectivity is realized by relying on internet technology (IPv6 and other internet protocols) and COTS solutions.

However, the military networks reside in an extremely hostile environment, where an active enemy is present. Hence, for the network to be useful, it is crucial that it is secured from both external and internal attacks. Although several security proposals exist, many still focus on securing the information or are tailored into a specific protocol, such as a routing protocol. In this paper, we present a generic security solution based on packet level authentication (PLA) that can be used to protect the infrastructure from both external and internal attacks, including denial-of-service attacks, and that can be used for access control, authentication, firewall applications, QoS policing, and so on. The solution is based on the fact that all IP packets traveling in the communication network can be verified for legitimacy by any legitimate node in the network, not only the source and destination nodes. We also present our prototype implementation of our architecture and provide some cost estimates.

SECURITY ISSUES IN THE MILITARY ENVIRONMENT

The military network has to function under extreme circumstances. The following requirements with respect to security must be met:

Authentication of nodes: the nodes in the network must be able to authenticate each other and perform a key exchange even without prior communication with each other.

Access control: a solution for access control to the network and its services is required.

Denial-of-service prevention: a compromised node should not be able to launch a denial-of-service attack that is propagated in the network. The damage of the attack should be restricted to the area of the compromised node only.

Revocation of compromised nodes: a means for removing compromised nodes from the network is needed.

Ensuring network functionality: the purpose of the network is to deliver the right packets (packet integrity) to the right place at the right time. Security measures are required to ensure this functionality.

PACKET LEVEL AUTHENTICATION

The PLA concept resembles that of the security of money. Every modern note (e.g. a 100 euro note) contains several security measures to ensure the authenticity of the note, such as micro print, changing colors, watermark, hologram, metal string, etc. When a merchant receives the note from a customer, the legitimacy of the note can be verified on the spot without having to consult a bank. The requirement is that the merchant must be able to verify the authenticity of a note using predefined security procedures and without prior knowledge of the customer.

In PLA, the same idea is applied to IP packets, that is, any node is able to verify the authenticity of the IP packet using predefined security procedures and without prior knowledge of or communication with the sender. The requirement applies for any receiver on the path from the source to the final destination(s). Thus, PLA is able of detecting illegitimate, erroneous, duplicated, and delayed packets in every router, not only at the final destination. In this way it is possible to restrict several classical attacks, such as denial-of-service attacks that are based on spoofing/forging IP packets and copying or manipulating legitimate IP packets. PLA also includes additional security features to cope with duplicated and replayed packets.

Design issues of packet level authentication

The PLA architecture takes advantage of standard IPv6 [2] header extension techniques used for example in Mobile IP [4][7]. The PLA header is added to every IP packet and contains the following information:

Node level information

- The identity of the trusted third party that has certified the node
- The public key of the node

- The validity period
- The certificate from the trusted third party that certifies the node

Packet level information

- Timeliness of the packet
- The sequence number of the packet
- The signature of the source node over the packet

With the node level information, any node (on the path from the originator to the final destination) that trusts the third party can verify that the originator of the packet is still among the trusted nodes without prior communication with that node. With the packet level information, that node can then verify that the originator of the packet has really generated this packet and the packet is not altered on the path from the sender to the receiver. Also, the node can verify the timeliness of the packet. Duplication of the packet is detected by using sequence numbers. Thus, corrupted, old, and duplicated packets can be discarded. This restricts the possibilities that the enemy have to attack the network. In the worst case [6], the enemy can make as many copies as there are separate routes to the destination node. In other words, the worst case is that the enemy is able to make a few extra duplicates.

Public key algorithms are used for signing the certificates stating the validity of the node as well as the node's signature over the packet. The lifetime of the certificates is limited, thus making it possible to reduce the damages caused by internal attacks either by revoking the certificate or not renewing it once its lease expires.

Packet level authentication extension header

The PLA extension header is depicted in Figure 1. Since the header is based on the standard IPv6 header extension technique, it is fully transparent and interoperable with routers that do not understand PLA. This means that PLA can be used together with any other protocols, such as Mobile IP or IPsec. If the PLA header is the first header in the packet, it protects both the data and all the other headers in the packet. Since PLA uses the standard IP header technique, the maximum IP packet size (MTU) handling is done correctly.

The PLA header consists of seven fields divided into two categories:

Node level information:

- Auth_id: The identity of the trusted third party that has certified the node (i.e., the identity that has authorized the node, this can be for example a general, TTP, access-network operator, home operator)

- Pub_key: The public key of the node (this public key is used by the verifying node to validate the sending node’s signature)

- Val_period: The validity period (this field indicates when the public key is considered to be valid by the trusted third party)

- Auth_signature: The certificate from the trusted third party that certifies the node (that the public key and validity period are correct)

Packet level information:

- Time: Timeliness of the packet (i.e., time when the packet was sent; it can be used to discard old packets)

- Seq_number: Sequence number of the packet (this monotonically growing field can be used in discarding duplicates and replay attacks)

- Node_signature: The originating node's signature over the packet (this signature field ensures that the IP packet has really generated by the sending node and it has not been altered on the path).

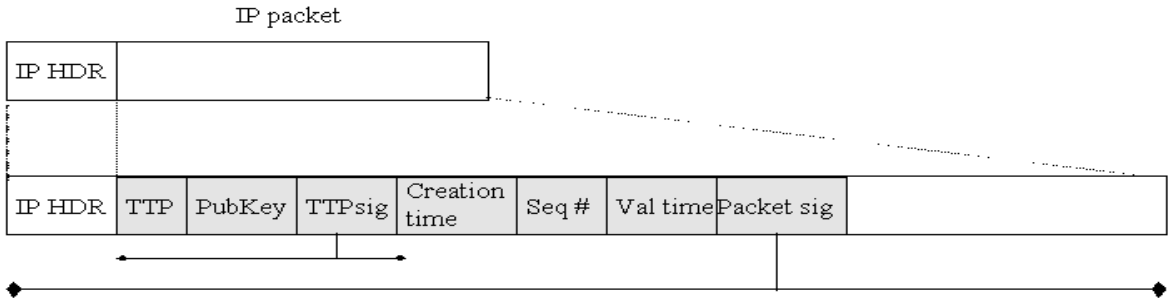

Figure 1. The PLA extension header

When the source node sends an IP packet, it fills the PLA header by taking the node level information (that is packet independent), incremented sequence number and current time. Finally, it calculates a signature over the entire packet and puts the result into the header. Then the IP packet is sent as any normal IP packet.

When a node receives the first packet from the sending node, it first validates the node level information and after that the packet level information. If any of the PLA fields is invalid or incorrect, the receiving node can discard the packet and optionally inform its upstream neighbor about the problem. For the subsequent packets from the same sending node, it is not needed to validate the node

level information if the verifier caches the previous validation result. Then, only the packet level verification is needed. Hence, the number of digital signatures that must be validated per packet is reduced from two signatures per packet to just one in case of several packets from the same sender.

IMPLEMENTATION

We have implemented the PLA architecture for the Linux operating system. The architecture was implemented as two user space programs which utilize the *ip6_queue* netfilter kernel module for capturing, modifying, and reinjecting packets into the network stack. Our PLA implementation is completely transparent to applications, and the applications do not need any modifications to use our implementation.

Architecture

Figure 2 illustrates the location of the Linux Netfilter hooks within the IPv6 stack. The hooks are originally intended for implementing the firewall in the Linux IPv6 stack. The figure shows three hooks called INPUT, OUTPUT and FORWARD.

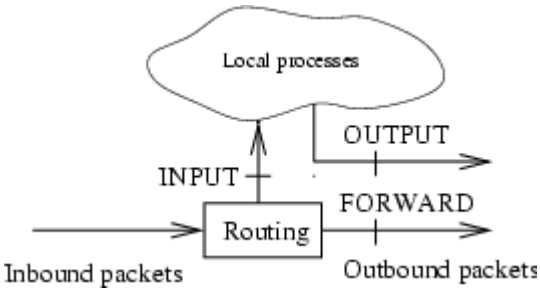

Figure 2: the Linux netfilter hooks in the IPv6 stack

The INPUT hook captures packets which are being delivered to this host. The FORWARD hook handles incoming packets and resides behind the routing functionality. This hook differs from the INPUT hook, in that it captures the packets that are being forwarded by this host. A packet that is captured by the INPUT hook is never captured by this hook. The OUTPUT hook captures all packets that originate from this host. Packets that have been received by the FORWARD hook are not seen by this hook.

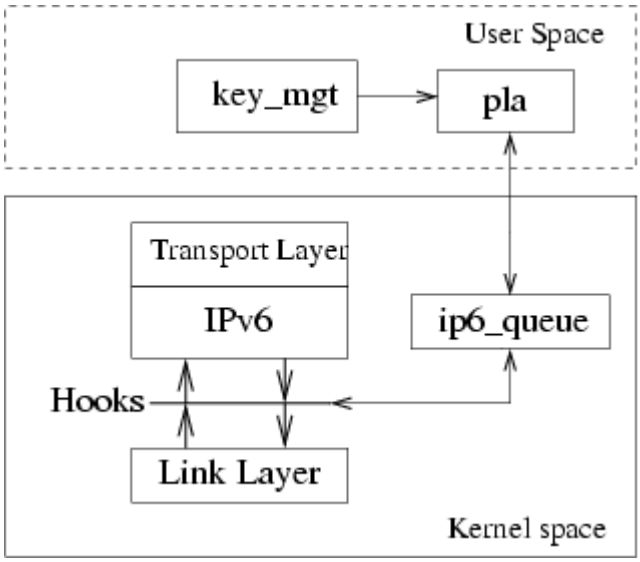

Figure 3: Implementation architecture

Figure 3 shows the communication between the modules in our implementation. The implementation consists of two modules in user space. The PLA module handles packet flows, including the decision to accept or drop any packets. The *key_mgt* module is used only for configuring the PLA module.

Inbound Packet Processing

When an inbound packets enters the IPv6 stack from the link layer, a routing decision is made. As is shown in Figure 3, the routing code can either forward the packet to another node or deliver the packet to local receivers. The packet is then intercepted either by the FORWARD or the INPUT hook depending on the routing decision. Next, the packet is given to the *ip6_queue* module which delivers the packet to the PLA daemon in user space.

The user space daemon receives the packet using the *libipq* library, which is the user space libary for intercepting and reinjecting packets into the Linux IPv6 stack. The packet is determined to be valid if it passes all of the steps in the PLA verification, and it will be reinjected into the IPv6 stack. If the packet was received via the INPUT hook, the PLA header is removed from the packet before reinjecting it, while a packet that was received through the FORWARD hook is forwarded unmodified.

Outbound Packet Processing

Outbound packets are intercepted by the OUTPUT hook in the IPv6 stack. At this point, the captured packets already have complete IPv6 headers. Like in inbound packet processing, the packets are intercepted to the *ip6_queue* module and passed on to user space through the *libipq* library. Once the packet is received by the user space daemon, the PLA header is added into the packet. After the header has been added, the user space daemon signs the packet using the host's private key, and the packet is reinjected into the IPv6 stack through the *libipq* library.

PLA PERFORMANCE

PLA uses Elliptic Curve Digital Signature Algorithm (ECDSA) [1] for packet integrity ensurance. When estimating the time needed to compute or validate one digital signature, we refer to an FPGA implementation of elliptic curve point multiplication [5], where one 163 bit ECC multiplication takes 106 ms with 90 MHz clock speed with 18079 slices (equals about 300 000 gates). This implementation is reported to perform about 5000 ECDSAs in a second [5] [3].

The ultimate goal is to be able to validate, in wire-speed, all traffic of standard 1 Gigabit and 10 Gigabit Ethernet interfaces. This means that we are then enabled to implement either a "bump-in-the-wire" module that could be attached as an add-on module to old routers or an integral part of a network interface card in new routers. In order to gain higher capacities, we shall scale up the HW by increasing the clock frequency and dice size. In Table 1, we have illustrated the number of parallel ECDSA modules and the number of gates that are needed to compute/validate all traffic with various packet sizes and wire speeds, when assuming that the chip runs at 500 MHz. The clock speed and the number of gates needed for PLA validation is still relatively small when compared with modern microprocessors that contain over 1000 million transistors and run over 3GHz clock speed.

Number of ECC modules needed		
Packet size [bytes]	1GE	10GE
100	45.00	450.00
1500	3.00	30.00
64000	0.07	0.70

Total number of gates needed		
Packet size [bytes]	1GE	10GE
100	15 750 000	157 500 000
1500	1 050 000	10 500 000
64000	24 609	246 094

Table 1: Estimated number of ECC calculation modules and total number of gates on PLA HW acceleration module

Additionally, it is possible to increase the performance at intermediate routers by applying an adaptive algorithm of packet validation. Assuming, that there are flows of packets from same originating node and there are several routers on the path from the originator to the destination, we can start the validation of every packet in every router on the path, but once the flow goes smoothly, we can lower the frequency of validation and check randomly, for example, every tenth packet in each router. Once a router detects erroneous packet, it shall inform upstream routers to start validate every packet of that flow.

CONCLUSION

In this paper, we have presented an architecture based on packet level authentication. The PLA concept can be used for authentication, access control, firewall applications, denial-of-service prevention, routing protocol security, and so on. In short, PLA secures the whole infrastructure. Our current implementation serves as a proof of concept that the architecture actually works as desired. According to our performance calculations, it is possible to achieve wire speed with PLA.

Furthermore, PLA is easy to integrate into the existing internet architecture, making it both usable in practice and cost efficient.

REFERENCES

[1] ANSI X9.62:1998, Public Key Cryptography: The Elliptic Curve Digital Signature Algorithm (ECDSA), ANSI Standard, X9F Data & Information Security Working Group, X9F1 - Cryptographic Tools Working Group, 1998.

[2] Deering, S. and Hinden, R.; Internet Protocol, Version 6 (IPv6) Specification; IETF RFC 2460; December 1998.

[3] D. Johnson, A. Menezes, and S. Vanstone. The Elliptic Curve Digital Signature Algorithm (ECDSA). International Journal of Information Security, Volume 1, Issue 1, Springer-Verlag, pages 36-63, August 2001.

[4] Johnson, D., Perkins, C., Arkko, J.; Mobility support in IPv6; IETF RFC 3775, June 2004

[5] K. Järvinen, M. Tommiska and J. Skyttä, "A Scalable Architecture for Elliptic Curve Point Multiplication", proceedings of the 2004 IEEE International Conference on Field-Programmable Technology, FPT 2004, pages 303-306, Brisbane, Queensland, Australia, December 6 - 8, 2004.

[6] Lundberg, J.; Packet level authentication protocol implementation; In Military Ad Hoc Networks; Series 1, No 19, Helsinki 2004

[7] Perkins, C.; IP Mobility Support for IPv4, IETF RFC 3220; 2002

COPYRIGHT