Enterprise Network Packet Filtering for Mobile Cryptographic Identities

Janne Lindqvist, Helsinki University of Technology, Finland

Essi Vehmersalo, Helsinki University of Technology, Finland

Miika Komu, Helsinki Insitute for Information Technology, Finland

Jukka Manner, Helsinki University of Technology, Finland

ABSTRACT

Firewalls are an essential component of the Internet and enterprise network security policy enforcement today. The configurations of enterprise firewalls are typically rather static. Even if client's IP addresses can be dynamically added to the packet filtering rules, the services allowed through the firewall are commonly still fixed. In this paper, we present a transparent firewall configuration solution based on mobile cryptographic identifiers of Host Identity Protocol (HIP). HIP allows a client to protect the data transfer with IPsec ESP, and supports dynamic address changes for mobile clients. The HIP-based firewall learns the identity of a client when it communicates with the server over HIP. The firewall configures the necessary rules based on HIP control messages passing through the firewall. The solution is secure and flexible, and introduces only minimal latency to the initial HIP connection establishment.

Keywords: *Firewalls, Network Security, Host Identity Protocol, HIP, Packet Filtering*

INTRODUCTION

Remote access to corporate service is very challenging to set up and configure in a secure way. The simplest way is on a per-service basis, by using HTTP and TLS and introducing a login function to public servers. Unfortunately, this leaves the server open for various attacks, e.g., DoS, since it must be open to any remote client regardless of the source IP address, and unlawful access attempts are caught very late in the login process.

To enable better security, and to disable most forms of attacks on the services, a VPN solution can be used, together with a tightly controlled firewall configuration. Through a VPN, corporate services are available for the client only after a successful VPN login transaction. Yet, also here the VPN server needs to be available to the public, and can be attacked. Moreover, configuration of the firewall is a

DOI: 10.4018/jhcr.2010090905

further cause of concern. First of all, setting up proper filtering rules for corporate firewalls is not a trivial task. Secondly, changing the connectivity provider results in network renumbering which further requires a full reconfiguration of the firewall. It is also also possible that the address of single VPN client changes due to device mobility or DHCP lease renewal. In such a case, the client has to reinitialize the VPN connection.

Two additional security concerns arise from the use of a typical VPN service. First, typically all the corporate services become available to the user when VPN gateway or firewall accepts a VPN connection. Then, a malicous user, virus or worm can try to mount an attack on any service of the corporation because VPNs do not offer protection against "internal" attacks. Second, the corporate services become vulnerable to attacks to "external" attacks if the device of a user is compromised. The attacker can route its own packets using the compromised device through the VPN tunnel to the corporate network. Hence, the attacker can practically mount any type of attack on the corporate services.

The second security concern was highlighted as part of the Microsoft Windows Vista routing compartments functionality, which was supposed to be included in the new operating system. The basic idea was that remote access from the user device is controlled per application, and not per host, making it impossible to route packets between interfaces, WLAN and VPN interfaces in our example. Yet, it is still not included, and one can only guess what the reasons are. Nevertheless, the security vulnerability still remains.

Firewalls are, unfortunately, a critical component of corporate and personal networks in the Internet today. Packet filtering is typically based on the 5-tuple of sender and receiver IP addresses and port numbers, and the transport protocol. Sophisticated firewalls can also filter based on the content of application layer protocols. Commonly, the filtering rules are quite static and constrained. The firewall passes only certain services and a known set of hosts through. In more dynamic networks, for example, offeering public or subscription-based WLAN access, or nomadic enterprise environments, the firewalls are controlled and rules set up based on some authentication exchange. Typically, a client is authenticated and authorized to use a WLAN service based on a web browser login application. If the login is successful, the firewall opens predefined services for the MAC and IP address of the client device. Only then the client can start access Internet to, for example, browse the web, or initiate VPN connections.

The current situation has at least four downsides. First, authentication for network access has a number of different implementation choices, which may or may not work with the device of the user, for example, on laptop computers, PDAs, or smart mobile phones. Second, the firewall allows the client to only use certain pre-defined services even when the client is authenticated successfully and authorized to use the Internet. It would be more useful to have a separate signaling protocol dynamically manage the filtering rules associated with a given authenticated client. Third, a third party can still listen to the network communications, collect varying information, and steal the identity of an authenticated client. Fourth, network renumbering becomes a problem, because all static rules on firewalls that are based on IP address must be changed when renumbering occurs. The same problem of updating firewall rules appears in access networks, where the IP address assigned to a client can change during the session, for example, in a mobile access network when the client performs a handover.

Setting up IP-based rules in a firewall to protect servers with roaming clients is difficult. Since the firewall cannot know the IP addresses of roaming employees, the rules that protect the network services must be quite liberal, or access is only possible through a separate VPN tunnel.

There seems to be a need for a remote access mechanism, that does not expose the corporate services to the security vulnerabilities described in this section, requires minimal configuration, is easy to set up and is operating-system independent. We have compared at a number

of alternatives and eventually we started to investigate the use of the Host Identity Protocol (HIP) as such a mechanism.

In this paper, we present a firewall architecture that allows efficient, scalable and secure network packet filtering. Our solution solves all the problems discussed above. The firewall is based on the Host Identity Protocol (HIP) (Moskowitz & Nikander, 2006) and tracking the protocol control messages and IPsec ESP SPI values. Although the standard IPsec architecture could be used to implement firewalls (Aura et al., 2005), our architecture provides a simple way to centrally enforce security policies regardless of host IPsec security policies. The architecture also allows to group and present services with cryptographically tamper-proof identities.

Our solution primarily targets the initial connection set up. Once the HIP control message exchange has been authenticated, subsequent message filtering is simply based on the source and destination IP addresses and SPI numbers of ESP packets. Thus, the processing overhead only applies in the beginning of the data connection. Our measurements show that this processing adds a negligible overhead to the connection initiation. A modern firewall with our architecture can support thousands of connection initiations per second.

One of the key features of using HIP and a HIP-enabled firewall is that the administration of the network does not need to care about IP addresses. Thus, the network can perform renumbering, and support mobile users without changes in the firewall rules. Moreover, when the client is using HIP, it does not need to employ any additional protocol for authentication and firewall control, either inside or outside the enterprise network. Furthermore, the solution also allows encrypting the data transfer end-to-end.

The firewall solution introduced in this paper does not require Internet-wide deployment of HIP. An enterprise can deploy HIP gradually to harness the integrated security, mobility, and multihoming capabilities for employees. Services and clients that do not use HIP continue to operate with the old system. In summary, by using HIP to access a service, the client is able to perform simultaneously network access authentication and authorization, firewall control, data transfer protection, and mobility management.

One of the key strengths of our design is that HIP can be used in any kind of wired or wireless network, for example, xDSL, Ethernet, WLAN, WIMAX, 3G, and any technology beyond 3G. For example, a modern mobile intelligent device with multiple different wireless link technologies can use the same mechanism for firewall traversal and configuration regardless of the active wireless connectivity.

In the next section we discuss related work, and in Section 3 we present the Host Identity Protocol in detail. In Section 4 we describe the firewall architecture and implementation, followed by performance evaluations, and a discussion of the solution.

RELATED WORK

We have reviewed a number of solutions for firewall control that would support secure mobility and multihoming. The solutions included proposals from Tschofenig et al. (Tschofenig et al., 2005a) and the IETF NSIS working group (Stiemerling et al., 2008). However, these approaches required explicit signaling with the firewall that contradicts our goal of transparent firewall control.

Firewalls have been a well-established technology throughout most of the modern Internet. The basic IP level filtering has been complimented with different extensions to filter transport layer protocols or different application layer technologies, for example, stateful filtering for TCP (van Rooij, 2000). SOCKS (Leech et al., 1996) is a framework for application and transport level gateway technologies for monitoring network connections. The SOCKS gateway is essentially a proxy, which authenticates a client establishing a connection and relays the connection request to server. Different authentication technologies can be incorporated into the SOCKS functionality. SOCKS gateway

may also contain translation functionalities to provide communication between IPv4 and IPv6 nodes (Kitamura, 1999).

SANE is a protection architecture for enterprise networks (Casado et al., 2006). It uses a centralized domain controller to implement security policies for the whole network. Clients contact the domain controller and need explicit permission to access any resources. The security policies can be expressed in natural ways, e.g. "give the multimedia group access rights to the company's mp3 server". The architecture introduces a new layer between link and IP layer, and is implemented in network switches. The architecture supports mobility, but only within the enterprise network. Also, SANE cannot be used to enforce security policies to internetworks such as the Internet.

Delegation oriented Architecture (DoA) proposes an extension to the current Internet architecture to facilitate the deployment of middleboxes (Walfish et al., 2004). It introduces a new layer and protocol between the network and transport layers. The new layer uses cryptographically secure identifiers similar to HIP. Using the new "middlebox" layer and identifiers, end-hosts can enforce the use of middleboxes, such as firewalls, even when they are not located on the path. Chaining of middleboxes is also possible by allowing the identifiers to be resolvable recursively to other identifiers.

SPINAT (Ylitalo et al., 2005) tackles problems related to IPsec awareness in NATs. One problem in traversing IPsec aware NATs is that the end-hosts determine the IPsec SPIs. This may cause SPI collisions especially when the end-host population within a single NAT is large. A straw-man solution is to drop the keyexchange messages with colliding SPIs and require the key exchange daemons to retry with different SPIs after a timeout. However, SPINAT proposes a more efficient solution with the IPsec SEET (Ylitalo et al., 2005) mode, which allows NATs to translate ESP SPIs upon collisions. The approach is applicable to asymmetric communication paths and can be used to integrate IPsec to overlay routing. Possibility of IPsec traffic filters are mentioned briefly, but not discussed in detail. The main problem of the SPINAT approach is related to deployment because many existing middleboxes do not support IPsec.

A number of IPsec related asecurity vulnerabilities are described by Aura et al (Aura et al., 2005). They conclude that security policies based on IP addresses can be circumvented in many ways. In HIP, security policies are based on hashes of public keys that makes HIP resilient against those types of attacks.

Ioannidis et al. (Ioannidis et al., 2000) have presented an implementation of a distributed firewall system. The authors use KeyNote to distribute the firewall policies to end-hosts. Their approach supports centralized management of security policies. A drawback is that it requires the end-hosts to update their security policies regularly. As a benefit, the approach allows fine-grained filtering at application layer and a centralized firewall is not a bottleneck for the network.

A preliminary version of this work appeared as a two-page extended abstract in the posters session of Usenix ATC 2007 (Lindqvist et al., 2007). The abstract presented motivation for the approach and notes on the preliminary implementation and performance.

HOST IDENTITY PROTOCOL ARCHITECTURE

This section presents the Host Identity Protocol Architecture explains its protocol mechanisms.

The Host Identity Protocol (HIP) (Moskowitz & Nikander; 2006 Moskowitz et al., 2008) renews the current TCP/IP architecture by introducing a new, cryptographic namespace, the Host Identity namespace, between the transport and network layers as shown in Figure 1. The new namespace consists of Host Identifiers (HIs). A HI is the public keys component of a private-public key pair. HIs can be either public or anonymous. The public HIs can be published, for instance, in the Domain Name System

(DNS) (Nikander & Laganier, 2008). The public identifiers are intended to be long-lived and the anonymous short-lived. For practical purposes, the public keys - the Host Identifiers - are represented by self-certifying hashes of the keys. The hash is called the Host Identity Tag (HIT). The advantage of using HITs instead of HIs, is that HITs are same size as IPv6 addresses and are compatible with current applications. For example, the HITs can be used to replace IPv6 address fields in other existing protocols and Application Programming Interfaces (APIs) (Moskowitz et al., 2008; Komu & Henderson, 2008). HIP also supports Local Scope Identifiers (LSIs) that can be used by legacy IPv4 software because the size of LSIs equals to the size of IPv4 addresses.

HIP architecture proposes so called identity-locator split which relieves IP addresses from their dual role of both identifying and locating end-hosts. In this new TCP/IP architecture, HIs identify endpoints and IP addresses are used to route packets between hosts. By splitting the dual role of unicast IP addresses, HIP supports end-host mobility and multihoming in a relatively straighforward way (Nikander et al., 2008).

The identifiers of the new namespace are deployed locally to the end-hosts in HIP. Alternatively, they can be deployed to global name services, such as DNS, or any overlay, such as OpenDHT (Rhea et al., 2005). However, HIP can be used without any support from the infrastructure by learning the peer's identifier in an opportunistic fashion during HIP key exchange negotiation (Moskowitz et al., 2008).

The HIP specification (Moskowitz et al., 2008) defines base exchange, which creates a secure communication context, called a HIP association, between two hosts. During the base exchange, two hosts authenticate to each other using their public keys and can create a pair of ESP Security Associations (SAs) (Jokela et al., 2008). The base exchange consists of four messages shown in Figure 2. The two hosts are referred as the Initiator (client) and Responder (server).

The HIP control packets consist of a fixed header and variable amount of parameters. The header contains the source and destination HITs and some other fields. All of the packets are

Figure 1. HIP layering model

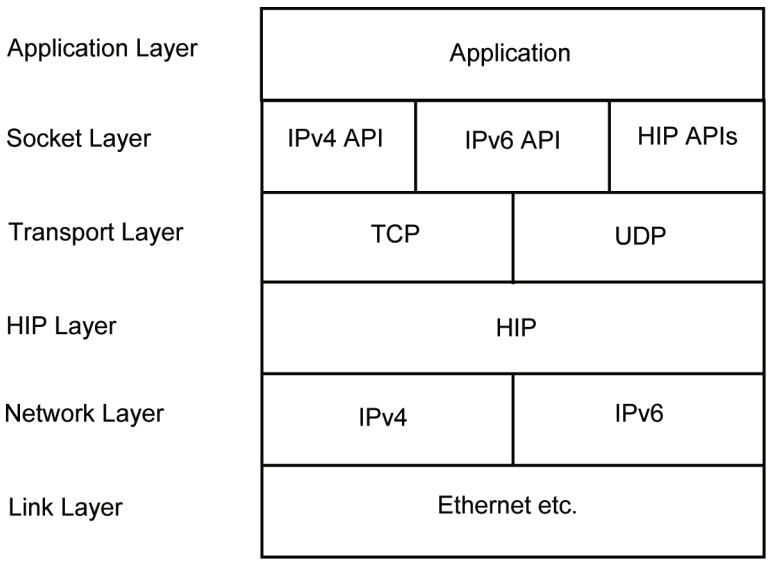
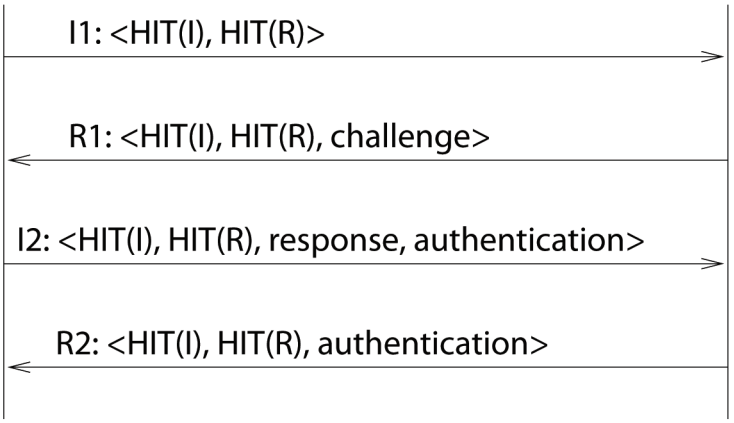


Figure 2. HIP base exchange



protected with public-key signatures except the first one. The first packet, I1, does not contain any parameters. The second packet, R1, contains the HI the Responder, Diffie-Hellman keying material and a computational puzzle (challenge) for the Initiator to solve. The puzzles are used as a mechanism for Denial of Service protection (Moskowitz et al., 2008). The Initiator sends solution to the puzzle, its HI, Security Parameter Index (SPI) for identifying incoming IPsec ESP flow and Diffie-Hellman keying material in I2 packet. The Responder remains stateless until it receives a valid I2. Upon receiving the I2, the Responder verifies the solution to the puzzle, creates state and concludes the base exchange with an R2 packet that contains its SPI number for incoming ESP flow.

HIP mobility and multihoming (Nikander et al., 2008; Nikander et al., 2003) takes place with UPDATE packets after a successful base exchange. A host moving to a different network reestablishes communications with its associated peers by sending an UPDATE packet to its peers. The packet contains parameter called LOCATOR which lists all locators of the mobile host. The parameter can be used in HIP control messages to inform other hosts about alternate addresses at which the originating peer can be reached. This ensures that address bindings can be updated dynamically without breaking the connections. HIP can also be used to establish efficient IPv4 to IPv6 handovers without tunnelling (Jokela et al., 2003).

Rendezvous servers (Laganier & Eggert, 2008) are complementary middleboxes in the HIP architecture. The rendezvous servers have a fixed IP address and serve as a stable contact points for end-hosts. End-hosts update their current location to their rendezvous servers always when they move. As an example, the rendezvous server are useful when two communicating end-hosts cannot publish their new location to each other directly after relocating simultaneously. Instead, they contact each other indirectly through their rendezvous servers that always know where the end-hosts are located. The rendezvous servers forward the first HIP control messages until the end-hosts have synchronized their new locations to each other and can communicate directly.

FIREWALL ARCHITECTURE

Our firewall operates on HIP control messages and ESP flows introduced in the previous section. Next, we describe the details of filtering design and the security consequences.

The Basic Firewall Design

The HIP-based firewall uses HITs to filter packets, but also certain other properties of network packets can be used in the firewall rules. When an Initiator sends an I1 through the firewall, it verifies that the HITs of the I1 message match the filtering rules and then records the HITs and IP addresses of the Initiator and Responder. The firewall has no means to validate the I1 because it does not contain any signatures. Therefore, a forged I1 can reach the Responder through the firewall. However, the firewall blocks the ESP data packets between the two hosts until the base exchange is completed successfully.

The responder sends an R1 and the firewall checks the HITs from its ACLs. This can be used to enforce access control restrictions on the Responders behind the firewall. The firewall records the HITs of the Initiator and the Responder and their IP addresses from the R1.

Upon receiving the R1, the Initiator solves the puzzle and replies with an I2 packet. The signed I2 packet contains the public key of the Initiator. The firewall verifies the signature either using the public key in the I2 packet or a preconfigured public key. If the verification fails, the firewall discards the I2 packet. Similarly, the firewall verifies signature of the concluding R2 packet from the Responder. The I2 and R2 packets contain the SPI values for IPsec ESP that the firewall requires to track ESP traffic. The firewall also tracks UPDATE messages to continue the tracking of IPsec ESP flows when the IP address of an end-host changes.

Further, the firewall expires the associated state when there is no traffic between the two related end-hosts for a certain time period. This guarantees that the state is removed when the firewall is no longer on the path between the two end-hosts. This can occur, for example, when an end-host moves to a different network or shuts down.

Service Identifiers

The IPsec architecture supports encryption between two hosts. The firewall architecture presented in this paper filters traffic based on HIP control messages and ESP flows. The firewall does not receive the ESP encryption keys of two communicating hosts, and therefore cannot inspect e.g. port numbers in the related ESP flows. Thus, the filtering granularity is lesser than for unprotected traffic where the five-tuple is visible. However, this problem exists even without HIP. For example, it is present in all communications that use IPsec ESP. The problem is also present in TLS, although the port numbers are visible in TLS.

Since the port numbers are not visible in the payload of IPsec ESP, a HIP-aware middlebox requires another way to distinguish between different ports, i.e. services, available at a server. In order to allow more granularity in filtering, we propose using the Host Identifiers also as service identifiers. For each service a server offers, it creates a different public/private key pair. This way, a HIP-aware middlebox or the server itself can separate different services and allow only certain clients to access certain services.

This service identifier approach is compatible with current name look up services. It is a common practice to have separate host names for different services, such as smtp.my.org and www.my.org. Introducing HIP-based service identifiers to the existing DNS would just require adding HIs to DNS (Nikander & Laganier, 2008), with each service a different HI. The HIs can be owned by a single host or multiple hosts. The approach is backwards compatible in the sense that existing servers could still be able to serve non-HIP clients and use existing filtering methods.

An alternative solution to this problem is to introduce a protocol extension to HIP that allows to share the ESP encryption key with the firewall. This can be used also for content filtering purposes, such as, removing viruses from the traffic. This approach is, however, beyond the scope of this paper.

Implementation

Figure 3 shows the design of the firewall implementation (Vehmersalo, 2005). It is based

on Linux Netfilter framework (Ziegler, 2001) to intercept network traffic. We used C-based HIPL implementation (Candolin et al., 2003) in our experimentation.

OVERVIEW

The main module of the firewall receiving packets from network interfaces and analyses the packets. It uses the other components of the firewall to produce verdicts based on properties of the packets received. The verdict decides whether the firewall accepts or drops the packet.

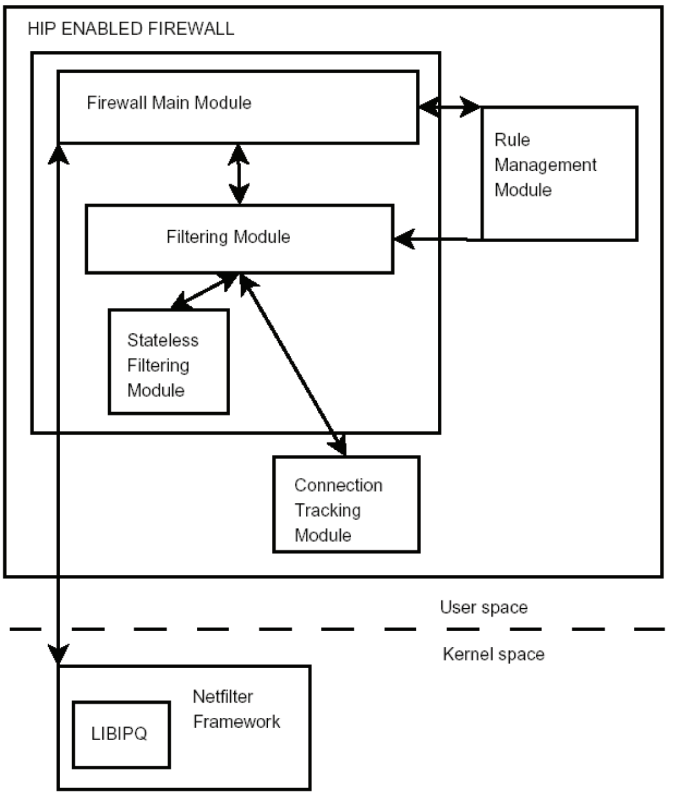
The firewall rules define the local security policies and are contained in the firewall rule set. The firewall rule management module manages the rules and verifies the rule syntax.

The Linux netfilter module contains hooks to the Linux networking stack to intercept packets. The HIP firewall registers to QUEUE target of netfilter and subscribes to HIP-related packet events. The QUEUE target allows userspace applications to read packets from the networking stack and assign verdicts on them.

PACKET FILTERING

The packet filtering consists of two functionalities. First, the firewall analyses packets based on the properties defined in the firewall policies. Second, the firewall assigns a verdict based on the analysis results and policy.

Figure 3. Overall implementation of the firewall architecture. Arrows denote interactions between different components



The firewall provides a number of static properties for packet analysis. The properties include include identity-based authentication of packets, HIP packet type and the direction of the packet (incoming and outgoing). The authentication compares source and destination identities with the firewall rules and also verifies packet signatures. The identities can be specified either has HITs or HIs in the firewall rules.

CONNECTION TRACKING

The packet filtering module calls the connection tracking module when a packet has to be filtered according to the connection state. The main purpose of connection tracking module is to maintain necessary state information to map individual packets to HIP associations. Connection tracking uses source and destination HITs of the HIP control headers, or HIs when available, to associate a HIP packet to a HIP association.

HIP base exchange and mobility-related packets include SPI numbers that the firewall uses to map ESP data packets to the corresponding HIP associations. When firewall analyzes a base exchange, the connection tracking module associates the SPI numbers to the HIP association. This way, the connection tracking can filter unwanted ESP communications based on identities.

The connection tracking module can also authenticate packets and verify packet signatures similarly as the packet filtering module. However, instead of static verification, connection tracking module extracts HIs from HIP traffic dynamically and uses the HI to authenticate the end-point in further communications. The authentication has different nature in connection tracking than in packet filtering. Connection tracking does not verify the identity against static rules but instead attempts to assure the property of sender invariance (Tschofenig et al., 2005b). The sender invariance guarantees that independent of the particular identity, the traffic can be trusted to be originating from the same responder throughout the lifetime of the connection. This is necessary, for instance, in a situation where trusted host inside the network initiates a connection to a previously unknown external host. Thus, the sender invariance makes it more difficult for attacking hosts to abuse the dynamically created access through the firewall.

The connection tracking module also analyzes UPDATE packets. When host introduces a new destination address related to an SPI, or an entirely new SPI, the connection tracker saves the new information to its state structures. The connection tracking must take into account that the two end-points maintain separate state information. This affects, for example, rekeying situations, where old information must remain valid until the other endpoint has acknowledged the new information. In practice, data packets with an old SPI could still be on the way when new SPI is announced. This principle is also discussed by van Rooij (2000) in the context of TCP protocol.

The connection tracking module inserts a timestamp into a connection data structure. The timestamp is then updated whenever valid packets of the connection are encountered. For detecting idle connections, the connection tracker checks the timestamps against a predefined timeout value. Idle connection could result, for example, when a host roams to another network where data is no longer intercepted by the firewall, or just shutdowns. Also, the state created in the firewall by the insecure I1 - R1 exchange does not reserve resources of the firewall indefinitely because of the time-out mechanism.

DATA STRUCTURES

HIP connection tracking module has structures similar to Netfilter's connection tracking. The structures are illustrated in Figure 4. As with Linux Netfilter, a tuple data structure contains

information that directly carried by a packet. The implementation provides tuples for both HIP and ESP packets, each in their own data structures.

PERFORMANCE EVALUATION

We conducted some measurements with the firewall prototype to understand its performance. The evaluation environment consisted of a server and five clients. The five clients were located in their own network, separated from the server using a single router that acted also as the HIP-based firewall. The network interfaces operated at 100 Mbit speed and we used IPv6 for connectivity. All of the hosts had a single Pentium 4 processor (3 Ghz) and their Linux kernel version was 2.6.17.3. We used 1024 bit RSA keys as Host Identifiers. The symmetric keys for IPsec were AES (128 bits) for HIP encryption, SHA1 (160 bits) for IPsec authentication and 3DES (192 bits) for IPsec encryption.

We measured the time observed by an client application to complete UNIX connect() system call, which executes a TCP handshake. This time was under 1 ms on the average without HIP. HIP and HIT verification at the firewall, this time was 65 ms on the average due to the extra processing cost of the base exchange. The verification of public key signatures at the firewall caused an extra delay of 1 ms at the maximum. The TCP handshake performance is summarized in Figure 5.

Thus, the firewall prototype introduced only a millisecond delay to a HIP-based TCP connection establishment in our test environment. In other words, the firewall implementation can support thousands base exchanges and mobility updates per second. Filtering ongoing connection creates a similar processing load as, for example, IP address based packet filtering in a traditional firewall. Hence, the HIP-based firewall architecture can scale well on middleboxes and migitates most of the processing cost at the end-hosts.

Figure 4. Connection tracking data model. Arrows represent pointer references between data structures

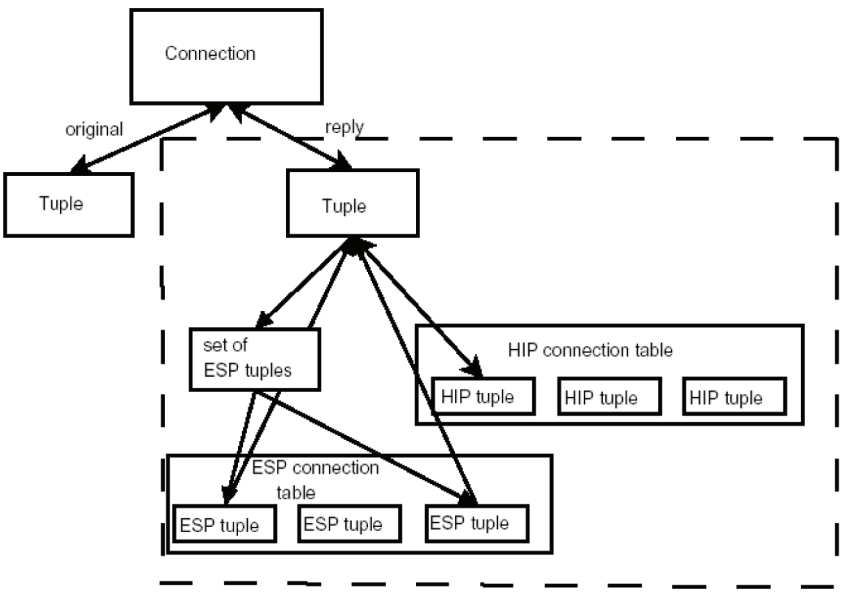
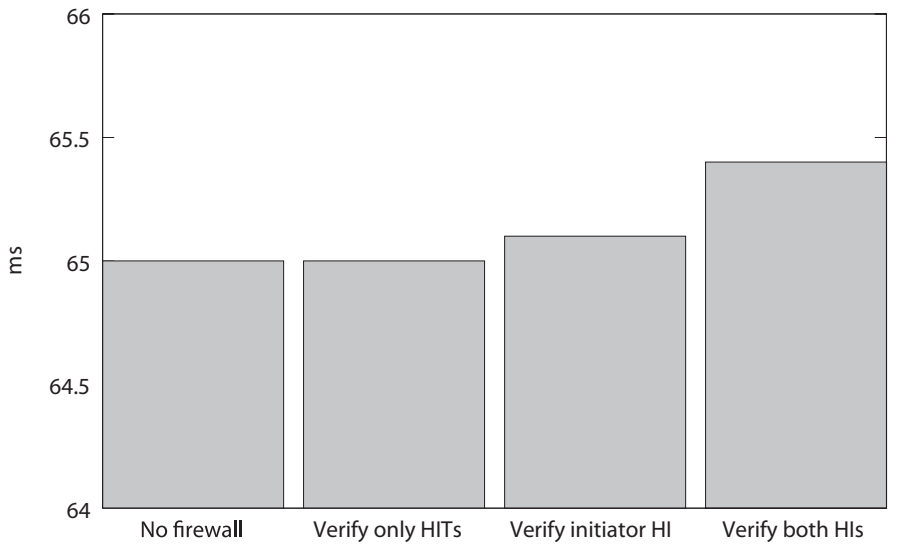


Figure 5. TCP connection establishment time



In addition to latency, we measured also throughput with TCP. The clients streamed 1 minute of TCP stream from the server through the firewall. We used "iperf" tool with default options for the measurements and varied the number of simultaneous file transfers. The results in Figure 6 indicate that the firewall did not affect TCP throughput significantly in our testing environment. The difference between HIP-based and non-HIP-based TCP throughput was approximately 2 Mbit/s.

We also measured TCP data transfer performance under two DoS scenarios and a under third scenario without DoS. In the first scenario, there was 1-4 rogue initiators that were flooding the responder with I1 packets while there was a data transfer in process from the server to the client. The second scenario was similar as first one, but the initiators were using a forged HIT that allows I1 traversal through the ACLs of the firewall. The third scenario includes throughput of varying number of simultaneous and legimitate data transfers. The results are show in Figure 7.

The two DoS attack scenarios have the same throughput performance. With random HITs, the I1 packets of the attackers stop at the firewall. With forged HITs, the firewall accepts the I1 packets of the attackers and they arrive at the responder. However, this causes insignificant processing cost at the responder because the responder has precreated a spool of R1 packets.

I1 flooding slowly degrades TCP throughput, but the throughput with multiple attackers and single legimitate transfer was still larger than with multiple, legimitate TCP data transfers. For example, data transfer with four attackers offered a performance of 29 Mbit/s, where as five simultaneous legimitate data transfers offered 18 Mbit/s. We assume that this was affected by the small size (40 bytes) of I1 packets. TCP packets are much larger and therefore multiple TCP streams reduce the throughput more than smaller I1 packets.

As a summary, the results are quite promising. The overhead of the firewall is negligible, both with control and data traffic. However, we realize that it would be useful to repeat the

Figure 6. Throughput of simultaneous data transfers

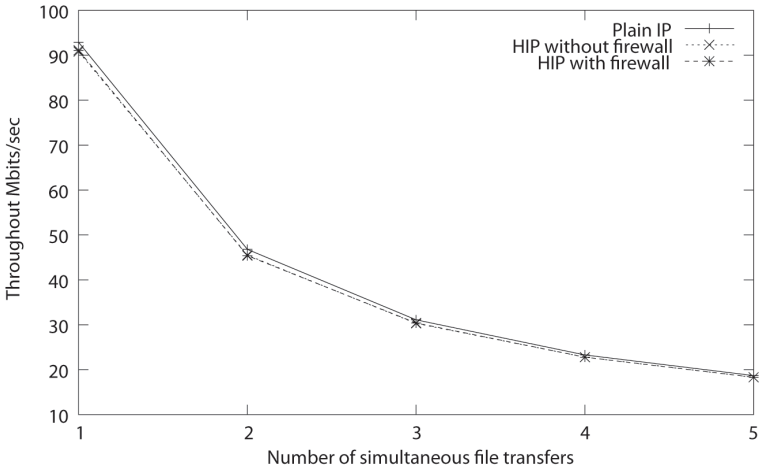
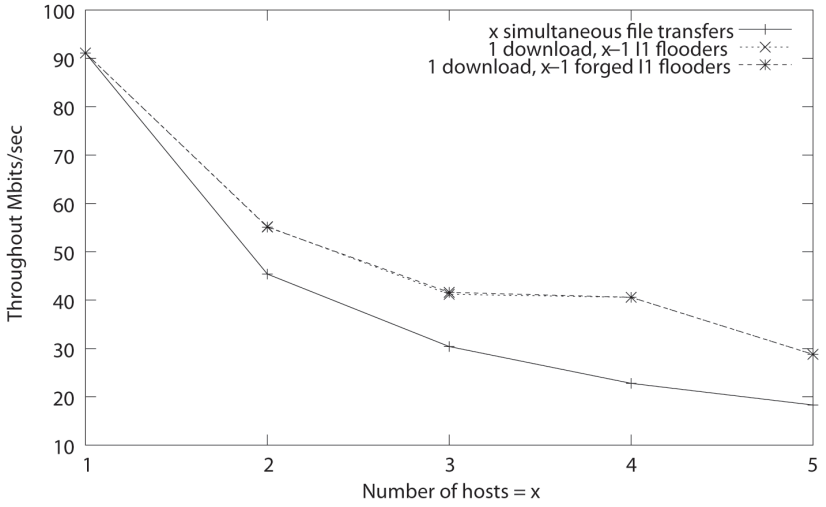


Figure 7. Simultaneous flooding and data transfers



measurements with a larger number of attackers and in gigabit networks.

DISCUSSION

The firewall implementation presented in this paper is scalable according to our measurements. However, we have identified a number of other challenges. The use of flat identifiers for access control introduces side-effects for identity and network management.

Additional Security Issues

The HIP base exchange establishes security associations and keying material between two end-hosts. The firewall tracks the SPIs of the ESP packets. An attacker can thus send ESP packets with valid SPIs through the firewall.

Naturally, the end-hosts discard these packets, but this introduces a possibility of flooding attacks directed towards the end-hosts. These issues are discussed in more detail in (Heer et al., 2009).

Multiple Identities per Host

A HIP-enabled operating system can support multiple Host Identities (Karlsson, 2005). One practical problem arises from the fact that users may have multiple affiliations, and also may want to use anonymous identifiers for privacy reasons.

The privacy management problem has been tackled by introducing a privacy management interface for MAC, IP and HIP layers (Lindqvist and Takkinen, 2006). A user can choose whether the identifiers in these layers are anonymous or public. However, this requires user expertise, and we cannot assume that all users can make decisions on this.

The complexity of the issue arises from the possibility to have multiple public identifiers. Typically, client side network applications do not care about the source identifier and just select the first one from the stack. The problem, in this case, is that an application can choose an identifier that is not configured to a firewall that protects the network. A tempting solution for the user is to communicate all identifiers to the firewall administrator, but this violates the privacy of the user as part of the identifiers are anonymous. In order to solve this, the client host has to have a local policy to enforce the selection of the correct source identifier for a given destination identifier.

Rulesets

A typical firewall today includes a rather large and complex set of filtering rules and setting up this ruleset is a challenging task. One problem is that one rule may be overlap with another. A service may be blocked due to a new badly formatted rule. Rules may leave unwanted access open in the firewall. Serious consequences may occur if the firewall rules are set up accidentally in a wrong order.

Typically, the outcome is that either certain service become out of reach of users because the administrator made a small change in the existing ruleset, or the firewall fails not filter all the intended traffic. Moreover, if users need access to additional services, inside or outside the firewall, an administrator must manually make a change in the rules; typically this is a long road, and the company security policy may not even allow it.

With our firewall design, rulesets is rather simple and supports mobile clients and renumbering of entire networks. Each client has a fixed firewall rule indepently of the physical location of the client. The rules of compromised clients can be revocated indepently of other clients. This allows also better protection against attacks internal to the corporation.

Management Issues

Although the HIP firewall can reduce administration effort, for example, with network renumbering, the management of HIs can be cumbersome. Further work is needed to find out ways how the public keys are introduced to the firewall. With HIP, the end-host creates and manages its own private keys. A problem in this approach is how to submit the public keys to the administration of the firewall. Next, we present two straw-man solutions to clarify the problem.

In the first solution, the administration is responsible for the installation of the operating system for e.g. a laptop. This is the usual case in many enterprise settings. However, granting access to the private key of the end-host introduces a possibility for privacy violations.

In the second solution, the user creates its own public/private key pairs, but the problem here is that how the keys are communicated to the firewall in a trustworthy way. Perhaps the user could hand the right key on a USB stick to administratation, but that is rather cumbersome in networks with thousands of users. Hence, a network-based transfer with a user-friendly

interface would seem more scalable here. An enterprise network could adopt a similar way to configure the HIP firewall as the EasyVPN (Benvenuto & Keromytis, 2003) approach. The EasyVPN approach uses a WWW server and TLS connections for configuring IPsec VPN gateways to clients. For HIP-enabled firewall, a similar management system could be implemented.

Another problem is that the identifiers in HIP are flat. IP addresses are hierarchical and contain a network prefix, which can be used for grouping IP addresses. There is no similar mechanism for the flat public key based identifiers in HIP. Public keys matching a certain pattern are computationally difficult to create and might potentially introduce further security problems.

One additional drawback of the current architecture is that it does not support easy grouping of services and administrators require a high-level management interface. The firewall implementation provides an interface that allows extending the management to e.g. web based management interface. The management interface could be used to attach semantic data to the flat identifiers. The data could consist of groups, user names and expiration dates for access control.

Combined Firewall and Rendezvous Service for Mobile Nodes

We have presented a solution where the firewall requires being located on the path. When a mobile host moves outside of the corporation, the firewall cannot filter its traffic anymore. NAT traversal extensions for HIP (Komu et al., 2009) introduce a new type of rendezvous service that forwards all HIP control traffic. Such service could be used to implement off-path filtering when a firewall service is coupled with the new rendezvous service. The mobile hosts would just have to deny all incoming base exchanges from other hosts than the new rendezvous server. This type of firewall would protect also communications for a mobile node that moves e.g. from an enterprise network to an public network at the cost of triangular routing.

CONCLUSION

We have presented a firewall architecture that allows both users to benefit of end-to-end mobility and multihoming in a secure way and allows the enterprise network management to centrally enforce their corporate network security policies.

Our initial measurements indicate that the overhead introduced by the firewall architecture is relatively small as the existence of the firewall in the network adds a delay of under 1 ms. The approach does not require an additional firewall control protocol when both the client and server support HIP. Despite that the HIP architecture seems promising for establishing the identifier/locator split to the Internet and providing seamless secure mobility and multihoming, it is not without tradeoffs. The management of the flat identifiers used in HIP introduces new challenges to operating systems and network management. We have proposed and implemented viable approaches to solving these problems. The possibility to use public keys as secure service identifiers and cryptographically secure authentication provided by our architecture is not available in the current Internet architecture or previous proposals.

ACKNOWLEDGMENT

The authors thank Sanna Liimatainen, Laura Takkinen, Kristian Slavov, Samu Varjonen and Antti Ylä-Jääski for their comments on the paper. The authors are also grateful to N. Asokan for fruitful discussions and suggesting the service identifier approach.

REFERENCES

Aura, T., Roe, M., & Mohammed, A. (2005). Experiences with host-to-host IPsec. *Security Protocols, 13th International Workshop.*

Benvenuto, M. C., & Keromytis, A. D. (2003). EasyVPN: IPsec Remote Access Made Easy. *17th USENIX Large Installation Systems Administration (LISA) Conference.*

Candolin, C., Komu, M., Kousa, M., & Lundberg, J. (2003). An implementation of HIP for Linux. In *Proc. of the Linux Symposium 2003*, Ottawa, Canada.

Casado, M., Garfinkel, T., Akella, A., Freedman, M. J., Boneh, D., McKeown, N., & Shenker, S. (2006). Sane: A protection architecture for enterprise networks. In *Proc. of 15th USENIX Security Symposium.*

Heer, T., Hummen, R., Komu, M., Götz, S., & Wehrle, K. (2009). End-host authentication and authorization for middleboxes based on a cryptographic namespace. In *ICC2009 Communication and Information Systems Security Symposium.*

Ioannidis, S., Keromytis, A. D., Bellovin, S. M., & Smith, J. M. (2000). Implementing a distributed firewall. In *Proceedings of the 7th ACM conference on Computer and communications security.*

Jokela, P., Moskowitz, R., & Nikander, P. (2008). *Using the encapsulating security payload (ESP) transport format with the host identity protocol (HIP).* RFC 5202, IETF.

Jokela, P., Nikander, P., Melen, J., Ylitalo, J., & Wall, J. (2003). Host identity protocol: Achieving IPv4 - IPv6 handovers without tunneling. In *Proc. of Evolute workshop 2003: "Beyond 3G Evolution of Systems and Services*

Karlsson, N. (2005). *Enabling Multiple HIP Identities on Linux*. Master's thesis, Helsinki University of Technology, Telecommunications Software and Multimedia Laboratory.

Kitamura, H. (1999). Entering the IPv6 communication world by the SOCKS-based IPv6/IPv4 Translator. In *Proc. of INET99*.

Komu, M., & Henderson, T. (2008). *Basic Socket Interface Extensions for Host Identity Protocol (HIP).* IETF.

Komu, M., Henderson, T., Tschofenig, H., Melen, J., & Keranen, A. (2009). *Basic hip extensions for traversal of network address translators.*

Laganier, J., & Eggert, L. (2008*). protocol (HIP) rendezvous extension*. RFC 5204, IETF. Host identity

Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., & Jones, L. (1996). *Socks protocol version 5.* RFC 1928, IETF.

Lindqvist, J., & Takkinen, L. (2006). Privacy management for secure mobility. In *Proceedings of the 5th ACM CCS Workshop on Privacy in Electronic Society.*

Lindqvist, J., Vehmersalo, E., Komu, M., & Manner, J. (2007, June 17-22). Enterprise Network Packet Filtering for Mobile Cryptographic Identities (extended abstract). In *USENIX annual technical conference poster session*, Santa Clara, CA.

Moskowitz, R., & Nikander, P. (2006). *Host Identity Protocol Architecture*. RFC 4423, IETF.

Moskowitz, R., Nikander, P., Jokela, P., & Henderson, T. (2008). *Host identity protocol.* RFC 5201, IETF.

Nikander, P., Arkko, J., Vogt, C., & Henderson, T. (2008). *End-Host mobility and Multi-Homing with the host identity protocol*. RFC 5206, IETF.

Nikander, P., & Laganier, J. (2008). *Host identity protocol (HIP) domain name system (DNS) extension.* RFC 5205, IETF.

Nikander, P., Ylitalo, J., & Wall, J. (2003). Integrating security, mobility, and multi-homing in a HIP way. In *Proc. of Network and Distributed Systems Security Symposium (NDSS'03)*, San Diego, CA, USA. Internet Society.

Rhea, S., Godfrey, B., Karp, B., Kubiatowicz, J., Ratnasamy, S., Shenker, S., et al. (2005). OpenDHT: A public DHT service and its uses. In *Proc. of ACM SIGCOMM'05,* Philadelphia, PA, USA: ACM Press.

Stiemerling, M., Tschofenig, H., Aoun, C., & Davies, E. (2008). *NAT/Firewall NSIS Signaling Layer Protocol (NSLP).*

Tschofenig, H., Nagaraja, A., Shanmugam, M., Ylitalo, J., & Gurtov, A. (2005a). Traversing Middleboxes with Host Identity Protocol. In *Proc. of ACISP'05.*

Tschofenig, H., Nagarajan, A., Torvinen, V., Ylitalo, J., & Grimminger, J. (2005b). *NAT and firewall traversal for HIP: draft-tschofenighiprg-hip-natfw-traversal-02.*

van Rooij, G. (2000). Real Stateful TCP Packet Filtering in Ipfilter. *2nd International SANE Conference*, Maastricht, The Netherlands.

Vehmersalo, E. (2005). *Host Identity Protocol Enabled Firewall - A Prototype Implementation and Analysis.* Master's thesis, Helsinki University of Technology, Telecommunications Software and Multimedia Laboratory.

Walfish, M., Stribling, J., Krohn, M., Balakrishnan, H., Morris, R., & Shenker, S. (2004). Middleboxes no longer considered harmful. In *Proc. of the 7th USENIX Symposium on Operating System Design and Implementation (OSDI 2004)*, San Fransisco, CA, USA: ACM Press.

Ylitalo, J., Salmela, P., & Tschofenig, H. (2005). SPINAT: Integrating IPsec into Overlay Routing. In *First International Conference on Security and Privacy for Emerging Areas in Communication Networks.*

Sams. Janne Lindqvist is a specialist researcher and PhD candidate with the Department of Computer Science and Engineering of Helsinki University of Technology (TKK). He received his MSc (Tech) on February 2005 from TKK. He will publicly defend his doctoral thesis on Practical Privacy Enhancing Technologies for Mobile Systems *on June 5, 2009. His research interests are in systems security and privacy, especially in building secure, privacy-preserving and usable systems.*

Essi Vehmersalo is a senior software engineer working at Nokia Corporation. She received her MSc in computer science from the Helsinki University of Technology (TKK) 2005 with thesis topic related to Host Identity Protocol enabled firewall technology. After that she has worked on mobility solution of the networking middleware of S60/Symbian smart phone platform. Currently she is working on Nokia music service.

Miika Komu graduated as MSc on 2004 and he is working on his postgraduate studies at Helsinki University of Technology. He is affiliated as a researcher at Helsinki Institute for Information Technology. He has been participating to various HIP related research, engineering, standardization and deployment activities since 2001.

Jukka Manner (born 1972) received his MSc (1999) and PhD (2004) degrees from the University of Helsinki. He is a full professor (tenured) and holds the chair of networking technology at the Department of Communications and Networking (Comnet) of the Helsinki University of Technology (TKK). He is also Adjunct Professor and a senior researcher at the University of Helsinki, and contributes to the research at the Helsinki Institute for Information Technology (HIIT). His research and teaching focuses on development of a future Internet, particularly in topics related to networking beyond IP, energy efficiency, mobility management, QoS, and transport protocols. He is the Academic Coordinator for the Finnish Future Internet research programme. He is an active peer reviewer and member of various TPCs. He has contributed to standardization of Internet technologies in the IETF for over 10 years, and is currently the co-chair of the NSIS working group. He has been principal investigator and project manager for over 15 national and international research projects. He has authored over 50 publications, including several IETF RFCs. He is also a member of the IEEE.