Network Working Group   T. Henderson
Internet-Draft   The Boeing Company
Expires: May 26, 2007   P. Nikander
                                          Ericsson Research NomadicLab
                                                     November 22, 2006

Using HIP with Legacy Applications
draft-ietf-hip-applications-00.txt

Status of this Memo

Copyright Notice

Abstract

The Host Identity Protocol and architecture (HIP) proposes to add a cryptographic name space for network stack names. From an application viewpoint, HIP-enabled systems support a new address family (e.g., AF_HOST), but it may be a long time until such HIP-aware applications are widely deployed even if host systems are upgraded. This informational document discusses implementation and API issues relating to using HIP in situations in which the system is

HIP-aware but the applications are not.

Table of Contents

1. Introduction

The Host Identity Protocol (HIP) [1] is an experimental effort in the IETF and IRTF to study a new public-key-based name space for use as host identifiers in Internet protocols. Fully deployed, the HIP architecture will permit applications to explicitly request the system to connect to another named host by expressing a location-independent name of the host when the system call to connect is performed. However, there will be a transition period during which systems become HIP-enabled but applications are not.

When applications and systems are both HIP-aware, the coordination between the application and the system can be straightforward. For example, using the terminology of the widely used sockets API, the application can issue a system call to connect to another host by naming it explicitly, and the system can perform the necessary name-to-address mapping to assign appropriate routable addresses to the packets. To enable this, a new address family (e.g., AF_HOST) could be defined, and additional API extensions could be defined (such as allowing IP addresses to be passed in the system call, along with the host name, as hints of where to initially try to reach the host).

This note does not define a native HIP API such as described above. Rather, this note is concerned with the scenario in which the application is not HIP-aware and a traditional IP-address-based API is used by the application. To use HIP in such a situation, there are a few basic possibilities: i) allow applications to use IP addresses as before, and provide a mapping from IP address to host identity (and back to IP address) within the system, ii) take advantage of domain name resolution to provide the application with either an alias for the host identifier or (in the case of IPv6) the host identity tag (HIT) itself, and iii) support the use of HITs directly (without prior DNS resolution) in place of IPv6 addresses. This note describes several variations of the above strategies and suggests some pros and cons to each approach.

When HITs are used (rather than IP addresses) as peer names at the system API, they can provide a type of "channel binding" (Section 1.1.6 of [2]) in that the ESP association formed by HIP is cryptographically bound to the name (HIT) invoked by the calling application.

2. Terminology

Host Identity Tag: A 128-bit quantity formed by the hash of a Host Identity. More details are available in [1].

Local Scope Identifier: A 32- or 128-bit quantity locally representing the Host Identity at the IPv4 or IPv6 API.

Referral: An event when the application passes what it believes to be an IP address to another application instance on another host, within its application data stream. An example is the FTP PORT command.

Resolver: The system function used by applications to resolve domain names to IP addresses.

3. Approaches for supporting legacy applications

This section provides examples of how legacy applications, using legacy APIs, can operate over a HIP-enabled system and use HIP. The examples are organized by the name used by an application (or application user) to name the peer system: an IP address, a domain name, or a HIT.

While the text below concentrates on the use of the connect system call, the same argument can also be applied to datagram-based system calls.

Recent work in the shim6 group has categorized the ways in which current applications use IP addresses [3]. These uses include short-lived local handles, long-lived application associations, callbacks, referrals, and identity comparisons. Each of the below mechanisms has implications on these different uses of IP addresses by legacy applications.

3.1. Using IP addresses in applications

Consider the case in which an application issues a "connect(ip)" system call to connect to a system named by address "ip", but for which we would like to enable HIP to protect the communications. Since the application or user does not (can not) indicate a desire to use HIP through the standard sockets API, the decision to invoke HIP must be done on the basis of host policy. For example, if an IPsec-like implementation of HIP is being used, a policy may be entered into the security policy database that mandates to use or try HIP based on a match on the source or destination IP address, or other factors. The mapping of IP address to host identity may be implemented by modifying the host operating system or by wrapping the existings sockets API, such as in the TESLA approach [4].

There are a number of ways that HIP could be used in such a scenario.

Manual configuration:

Pre-existing SAs may be available due to previous administrative action.

Opportunistically:

The system could send an I1 to the Responder with an empty value for Responder HIT.

Using DNS:

If the responder has host identities registered in the forward DNS zone and has a PTR record in the reverse zone, the initiating system could perform a reverse+forward lookup to learn the HIT associated with the address. Alternatively, the HIT could be stored in some type of HIP name service such as a DHT, keyed by IP address. Unless secured with DNSSEC, the use of the reverse DNS map is subject to well-known security limitations (an attacker may cause an incorrect IP address to FQDN binding to occur).

These types of solutions have the benefit of better supporting applications that use IP addresses for long-lived application associations, callbacks, and referrals. They have weaker security properties than the approaches outlined in Section 3.2 and Section 3.3, however, because the binding between host identity and address is weak and not visible to the application or user. In fact, the semantics of the application's "connect(ip)" call may be interpreted as "connect me to the system reachable at IP address ip" but perhaps no stronger semantics than that. HIP can be used in this case to provide perfect forward secrecy and authentication, but not to strongly authenticate the peer at the onset of communications. DNS with DNSSEC, if trusted, may be able to provide some additional initial authentication, but at a cost of initial resolution latency.

Using IP addresses at the application layer may not provide the full potential benefits of HIP mobility support. It allows for mobility if one is able to readdress the existing sockets upon a HIP readdress event. However, mobility will break in the connectionless case when an application caches the IP address and repeatedly calls sendto().

3.2. Using DNS

In the previous section, it was pointed out that a HIP-enabled system might make use of DNS to transparently fetch host identifiers prior to the onset of communication. For applications that make use of DNS, the name resolution process is another opportunity to use HIP. If host identities are bound to domain names (with a trusted DNS) the following are possible:

Return HIP LSIs and HITs instead of IP addresses:

The system resolver could be configured to return a Local Scope Identifier (LSI) or Host Identity Tag (HIT) rather than an IP address, if HIP information is available in the DNS that binds a particular domain name to a host identity, and otherwise to return an IP address as usual. The system can then maintain a mapping between LSI and host identity and perform the appropriate conversion at the system call interface or below. The application uses the LSI or HIT as it would an IP address.

Locally use a HIP-specific domain name suffix:

One drawback to spoofing the DNS resolution is that some applications actually may want to fetch IP addresses (e.g., diagnostic applications such as ping). One way to provide finer granularity on whether the resolver returns an IP address or an LSI is to distinguish by the presence of a domain name suffix. Specifically, if the application requests to resolve "www.example.com.hip" (or some similar suffix), then the system returns an LSI, while if the application requests to resolve "www.example.com", IP address(es) are returned as usual. Caution against the use of FQDN suffixes is discussed in [5].

Since the LSI or HIT is non-routable, a couple of potential hazards arise, in the case of referrals, callbacks, and long-lived application associations. First, applications that perform referrals may pass the LSI to another system that has no system context to resolve the LSI back to a host identity or an IP address. Note that these are the same type of applications that will likely break if used over certain types of NATs. Second, applications may cache the results of DNS queries for a long time, and it may be hard for a HIP system to determine when to perform garbage collection on the LSI bindings. However, when using HITs, the security of using the HITs for identity comparison may be stronger than in the case of using IP addresses.

It may be possible for an LSI or HIT to be routable or resolvable, but such a case may not have the level of security in the binding to host identity that a HIT has with the host identity. For example, a special IP address that has some location invariance is the identifier-address discussed in [6]. In general, LSIs and HITs considered to date for HIP have been non-routable.

3.3. Connecting directly to a HIT

The previous two sections describe the use of IP addresses and and

LSIs as local handles to a host identity. A third approach, for IPv6 applications, is to configure the application to connect directly to a HIT (e.g., "connect(HIT)" as a socket call). Although more cumbersome for human users (due to the flat HIT name space) than using either IPv6 addresses or domain names, this scenario has stronger security semantics, because the application is asking the system to connect specifically to the named peer system.

Depending on how HITs are ultimately defined, it may be hard for a system to distinguish between a HIT and a routable IPv6 address. Elsewhere it has been proposed [7] that HITs be precluded from using highest-ordered bits that correspond to IPv6 addresses, so that at least in the near term, a system could differentiate between a HIT and an IPv6 address by inspection.

Another challenge with this approach is in actually finding the IP addresses to use, based on the HIT. Some type of HIT resolution service would be needed in this case.

A third challenge of this approach is in supporting callbacks and referrals to possibly non-HIP-aware hosts. However, since most communications in this case would likely be to other HIP-aware hosts (else the initial connect() would fail), the problem may be instead if the peer host supports HIP but is not able to perform HIT resolution for some reason.

4. Security Considerations

In this section we discuss the security of the system in general terms, outlining some of the security properties. However, this section is not intended to provide a complete risk analysis. Such an analysis would, in any case, be dependent on the actual application using HIP, and is therefore considered out of scope.

The three outlined scenarios differ considerably in their security properties. There are further differences related to whether DNSSEC is used or not, and whether the DNSSEC zones are considered trustworthy enough from an application point of view.

When IP addresses are used to represent the peer system, the security properties depend on the the configuration method. With manual configuration, the system's security is comparable to a non-HIP system with similar IPsec policies. The security semantics of an opportunistic key exchange are roughly equal to current non-secured IP; the exchange is vulnerable to man-in-the-middle attacks. However, the system is less vulnerable to connection hijacking attacks. If the DNS is used, if both maps are secured (or the HITs stored in the reverse MAP) and the client trusts the DNSSEC signatures, the system may provide a fairly high security level. However, much depends on the details of the implementation, the security and administrative practises used when signing the DNS zones, and other factors.

Using the forward DNS to map a DNS name into an LSI is a case that is closest to the most typical use scenarios today. If DNSSEC is used, the result is fairly similar to the current use of certificates with TLS. If DNSSEC is not used, the result is fairly similar to the current use of plain IP, with the exception that HIP provides protection against connection hijacking attacks.

If the application is basing its operations on HITs, the connections become automatically secured due to the implicit channel bindings in HIP. That is, when the application makes a connect(HIT) system call, the resulting connection will either be connected to a node possessing the corresponding private key or the connection attempt will fail.

5. Acknowledgments

Jeff Ahrenholz, Miika Komu, Teemu Koponen, and Jukka Ylitalo have provided comments on different versions of this draft.

6. References

[1] Moskowitz, R., "Host Identity Protocol", draft-ietf-hip-base-06 (work in progress), June 2006.

[2] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, January 2000.

[3] Nordmark, E., "Shim6 Application Referral Issues", draft-ietf-shim6-app-refer-00 (work in progress), July 2005.

[4] Salz, J., Balakrishnan, H., and A. Snoeren, "TESLA: A Transparent, Extensible Session-Layer Architecture for End-to-end Network Services", Proceedings of USENIX Symposium on Internet Technologies and Systems (USITS), December 2003.

[5] Faltstrom, P., "Design Choices When Expanding DNS", draft-iab-dns-choices-04 (work in progress), October 2006.

[6] Bagnulo, M. and E. Nordmark, "Level 3 multihoming shim protocol", draft-ietf-shim6-proto-06 (work in progress), October 2006.

[7] Nikander, P., "An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers (ORCHID)", draft-laganier-ipv6-khi-05 (work in progress), September 2006.

Authors' Addresses

Tom Henderson
The Boeing Company
P.O. Box 3707
Seattle, WA
USA

Email: thomas.r.henderson@boeing.com

Pekka Nikander
Ericsson Research NomadicLab
JORVAS FIN-02420
FINLAND

Phone: +358 9 299 1
Email: pekka.nikander@nomadiclab.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.