Host Identifier Revocation in HIP
draft-zhang-hip-hi-revocation-01

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on April, 2010.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in
Abstract

This document mainly analyzes the key revocation issue with host identities (HIs) in the Host Identity Protocol (HIP), which has not attracted enough attention from the HIP community yet. As a key functionality of key management mechanisms, key revocation is critical for security systems especially which are expected to operate for a long period. Apart from that, this document also discusses the potential challenges that the designers of HI revocation mechanisms have to encounter and propose candidate solutions.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RFC2119].

Table of Contents

1. Introduction ................................................... 2
2. Key Management ................................................ 3
3. HI Revocation in HIP ............................................ 4
   3.1. Implicit HI Revocation in HIP ............................. 5
   3.2. Explicit HI Revocation without Third Parties .......... 5
   3.3. Explicit HI Revocation with Third Parties .............. 6
4. Issues with the Introduction of Trusted Third Parties .......... 7
5. Conclusions .................................................... 8
6. IANA Considerations ............................................ 8
7. Acknowledgments ................................................ 8
8. References ..................................................... 8
   8.1. Normative References ...................................... 8
   8.2. Informative References .................................... 9
Authors’ Addresses ................................................ 9

1. Introduction

In HIP architectures [RFC5201], a host needs to select a permanent public key pair before it attempts to communicate with other hosts. The public key is used as its HI while the private key is kept...
securely. When initiating a communication (e.g., a TCP session), the communicating peers adopt their HI key pairs to perform mutual authentication and distribute secrets for subsequent usage. Thus, the security of HI key pairs is the foundation of the security of HIP architectures. After compromising the HI key pair of a host, an attacker can easily impersonate the victim to carry out malicious attacks. However, a cryptographic public key pair is in more dangers of compromise after having been employed for a certain period. As time elapses, attackers can collect more materials (e.g., encrypted data, signatures and associated plain texts, etc.) and obtain more time to compromise the key pair. It is widely recognized that keys should have reasonable valid periods [Recommendations]. Therefore, in security systems expected to be executed for a long time, the cryptographic keys need to be revoked when they are not secure enough any more.

In current HIP architectures, the key revocation issues with transient (session) keys have been well discussed. HIP allows two communicating hosts to update their transient keys at run time. However, the key revocation issues with permanent keys (i.e., HIs) have not been carefully analyzed yet. In current HIP architectures, there is no facility provided for HI revocation either. Because key revocation is a core component of key revocation mechanism, this document first generally introduces key management mechanisms. After this, the security issues with HI revocations in typical scenarios are discussed in detail respectively.

2. Key Management

Key management aims at guaranteeing the security of cryptographic keys during the period of their application and includes all of the provisions made in a security system design which are related to generation, validation, exchange, storage, safeguard, application, and replacement of cryptographic keys. Appropriate key management is critical to security systems. Generally, a key management mechanism can be broken down into five sub-systems: key generation, key maintenance, key distribution, key assessment, and key revocation.

Key generation systems concern the issues related with key generating processes. Key maintenance systems deal with the issues with key storage. Key distribution systems deal with the issues with regard to distributing cryptographic keys for users. Key assessment systems take in charge of verifying the legality of cryptographic keys according to certain policies. Key revocation systems cover the issues with regard to discarding and updating cryptographic keys.
In the sub-systems of a key management mechanism, key revocation is the most complex and may cover most functions of the other four sub-systems. In a typical key revocation process, antique keys need to be abandoned and securely discarded; new keys need to be generated, assessed and distributed; the users supposed to be affected by the change of keys need to be notified. A key revocation process may be triggered in different scenarios. For instance, when a host changes its private key it needs to inform all other communicating partners of the change of its key. Also, when a host, due to misbehavior or end of validity period, is removed from the system, other partners which it is communicating with (or it intends to contact) need to be informed efficiently.

Generally, there are two types of key revocation approaches: implicit key revocation and explicit key revocation. Indicated by the name, implicit key revocation does not need any additional operations to revoke a cryptographic key. For instance, in a public key infrastructure (PKI), a Certificate Authority (CA) can issue a certificate for a user to assert the relationship between the user and its public key. The certificate is associated with a life period. When the period expires, the user’s public key is revoked as well. However, in order to revoke a key, an explicit key revocation mechanism needs to allocate an entity (e.g., the owner the key) to explicitly deliver notifications. For instance, in X.509 [RFC2459] based systems, an issuer can generate a list of certificates, which were revoked but not expired yet, for users to consult.

3. HI Revocation in HIP

The revocation of transient keys has been considered in the design of HIP. In the basic handshaking protocol, communicating hosts employ the authenticated Diffie-Hellman algorithm to distribute a symmetric key which is used to generate new cryptographic keys adopted in the following communications. After the handshake, the hosts can refresh their transient keys and the corresponding associations, by exchanging update packets. The revocation issue with HIs, however, has not been carefully considered yet. In current HIP architecture, hosts are able to update their identifiers arbitrarily without notifying others. This loophole can be taken advantage of by attackers to, for instance, escape auditing or bypass ACLs. In this section, related issues and candidate approaches are discussed.
3.1. Implicit HI Revocation in HIP

The basic principle of an implicit HI revocation mechanism is to associate a key with a valid period and use cryptographic methodology to prove the binding between the key and its valid period. In practice, the valid period of a key can be transported in HIP headers either using a parameter or a certificate. The issuer of the key can calculate a signature for the key and its valid period. By verifying the signature, users can assert whether the key is still valid.

Although the principle is intuitive and there have been lots of successful practices, several practical issues still need to be further explored in order to achieve implicit HI revocation in HIP. For instance, in current HIP implementations, hosts have a full control over its permanent key pairs. A host can arbitrarily assign, reset, or extend the life time of its HIs, which may cause serious security issues. Assume that a lazy manager of a host sets an extremely long life period for the HI of the host to avoid frequent update of the key pairs. Thus, the attackers are able to obtain enough time to compromise the key. A practical solution to this problem is to allocate a trusted third party to manage the life period for each HI. The assertion of the third party should be able to be verified by the users. However, the introduction of trust third parties will cause big modifications on HIP architectures, and the related issues are discussed in section 4.

Compared with explicit key revocation mechanisms, implicit key revocation mechanisms usually introduce less overhead, since on additional information is required to notify the revocation of antique keys. However, implicit key revocation mechanisms are incapable in the scenarios where keys need to be revoked for some reasons (e.g., compromise) before they expire. In many security architectures (e.g., X.509 [RFC2459]), both implicit and explicit key revocation are supported.

3.2. Explicit HI Revocation without Third Parties

In a scenario where there is no assistance from trusted third parties, a host intending to refresh its HI needs to notify the update to its collaborating peers. Cryptographic methodologies must be used to ensure that the notification messages cannot be foiled, modified, replayed so as to prevent adversaries from using these messages to perform attacks. An intuitive approach is to define a new type of update packet. When a HI is revoked, the host carries out basic handshakes with its peers, establishes security channels
and then transports the update message through the channel to notify the revocation. If a host is communicating with its peers at the time when the revocation is performed, the host can use the established security channels to transport update packets directly.

If a host only has a small group of collaborating partners and the relationship between the host and its partners is stable, this solution can be effective. However, if the number of the partners of a host is large or the partners changes frequently, the host has to spend large efforts to maintain the information of its partners and to inform its partners of the revocation. In addition, the job of maintaining the partner information normally needs the involvement of human beings which are error-prone. Any omission of candidate partners will cause difficulties or failures in future communications, not to even mention that candidate partners are unpredictable on many occasions. Moreover, this type of solution requires all the partners to be available during the period of revocation. Otherwise, the host will miss the update information. This requirement, however, is difficult to fulfill, especially when hosts communicate in an asynchronous mode.

3.3. Explicit HI Revocation with Third Parties

The notification approach mentioned above uses a "push" model, if a host needs to communicate with a large number of peers, the host may suffer from the overhead introduced by the revocation of its HI. In this case, a "pull" model shows its advantage. That is, to deploy a trust third party for hosts to upload their updated HI information and for users to query whenever they need assurance about the freshness of the key. DNS can be a good candidate to implement such a model. When a host refreshes its HI, it can update its Resource Record (RR) on a DNS server. Other communicating parties then can use the host’s FQDN to find out its latest HI. Security extensions of DNS have been developed, which can ensure that only authorized hosts can update their RRs. In practice, the effectiveness of DNS servers as a looking-up solution has proved. However, there are some inherent issues of DNS servers which must be carefully considered. For instance, the long delay caused by propagating the updated information throughout DNS servers may result in the inconsistency of HI information stored in different DNS servers. Therefore, it is difficult for a user to guarantee that the HI it obtains is the latest one. Moreover, in some scenarios, the service of mapping from antique HIIs to latest HIIs is required. For instance, DNS allows a FQDN to be mapped to multiple hosts in order to facilitate load balance. Each time when receiving a query about the FQDN, the DNS server returns the addressing information about a different host.
Therefore, when a user sends a DNS server with the FQDN of the desired host again and obtains a HIT different from the previous one, it is difficult for the host to find out whether the HI it gains is the updated HI of the desired host or the HI of another host. The later condition may result in errors as the host indicated by the HI does not have the state information maintained by the desired host. It is possible to store a list of HIs that a party used to adopt in its RR. But the performance of this solution becomes bad, when the length of the list increases.

In other resolution mechanisms which facilitate mapping from HIT to IP address (e.g., DHT), the mapping service from the old HI to updated HI is more important as users have no other namespace to rely on.

4. Issues with the Introduction of Trusted Third Parties

In the above discussion, we indicate that there are lots of practical issues which are difficult to be addressed without the assistance of trust third parties. However, it has been well known that trusted third parties may cause the issues in terms of scalability and single-point-of-failure, which limits their application in generating large-scale security mechanisms. In HIP the key management infrastructure is expected to serve all end systems attached to the Internet in a global scale. Currently, there have been already billions of hosts attached to the Internet. Considering other types of devices (e.g., mobile phones, network printers, and even human beings) which has been connected or are going to be connected to the Internet, the number of the users that the infrastructure needs to serve is even larger. To this end, no trusted-third-party dependent security infrastructure ever deployed has the capability to effectively support such a large amount of users. In addition, the Internet is comprised of multiple networks which belong to different countries or organizations. As there is no unique trust domain in the Internet, two hosts intending to communicate may belong to two different trust domains, and each host may not trust the assertion made by the security server in the other domain. Although there have been solutions of generating trust relationship across various trust domains, all these solutions impose additional communications and key exchanges, which will negatively influence the performance of HIP. Moreover, the HIP community argues that two HIP-aware hosts should be able to communicate without any additional security facilities. Actually, the only third party server introduced in the base-line HIP architecture is the Rendezvous Server (RVS)[RFC5204]. A RVS only relays messages for the hosts which attempts to communicate with
mobile hosts and provides little security functionality. The hosts intending to communicate still need to use the basic handshaking protocol to carry out authentication and exchange secrets. Consequently, in order to introduce trusted third parties, the designers of HIP revocation mechanisms have to modify current HIP architectures and may break some principles having been adopted in the design of HIP.

It needs to be clarified that we do not oppose integrating additional security infrastructures with HIP. However, before implementing such a security infrastructure, the concerns mentioned above need to be carefully investigated.

5. Conclusions

If HIP is only supposed to be an ID/Locator separating solution and security is dispensable, key revocation may not be a critical issue on many occasions. However, if HIP is expected to keep working securely for a relatively long period, proper key management mechanisms have to be provided. In fact, key management has been an active research area for a long period, and lots of successful key-management systems (e.g., PKI) are widely adopted in practice. However, because of many issues (e.g., scalability, lack of trust), it is unwise to employ these key-management systems for HIP directly. Based on the discussion of this document, we argue that the HI revocation mechanisms should be a comprehensive and able to perform the functions of both implicit and explicit key revocation. As different key revocation approaches have their advantages as well as limitations, the designers of HI revocation mechanisms have to combine them elaborately and make tradeoffs sometimes.

6. IANA Considerations

No such considerations.

7. Acknowledgments

Thanks Thomas. R. Henderson for his kindly revision and precious comments.

8. References

8.1. Normative References

8.2. Informative References

[Recommendations]

Barker, E., Barker, W., Burr, W., Polk, W., and M. Smid,
"Recommendation for Key Management-Part1-General(Revised)",
NIST Special Publication 800-57, March 2007,
<http://csrc.nist.gov/publications/nistpubs/800-57/SP800-
57-Part1.pdf>.

Authors’ Addresses

Dacheng Zhang
Huawei Technologies Co., Ltd
KuiKe Building, No.9 Xinxi Rd.,
Hai-Dian District
Beijing, 100085
P.R. China
Email: zhangdacheng@huawei.com

Dmitriy Kuptsov
Helsinki Institute for Information Technology
PO. Box 9800
TKK FI-02015
Finland
Email: dmitriy.kuptsov@hiit.fi