Anonymous IPsec with Plug and Play: a prototype of IPsec with IKE using IPv6 temporary addresses and anonymous public-key certificates

Kazuomi Oishi * Haruyuki Kitawaki *

Abstract— An IPv6 device can use temporary addresses to have its activities not to be linked by a third party on the Internet. We study a case where an IPv6 device using a temporary address wants to execute IPsec with IKE (Internet Key Exchange) hiding its identity even to the communication partner. Ordinary certificates that include identity of the certificate holders must not be used, so that the authenticity of the device needs to be confirmed by other method to prevent the man-in-the-middle attack. We examine the problem from the viewpoints of IPsec and IKE specifications and anonymity, and propose a solution. Such a solution is referred to as *Anonymous IPsec*. We study role of the Certification Authority (CA) and consider deployment of the CAs. In our solution, a CA issues to an IPv6 device an X.509v3 anonymous public-key certificate that includes the IPv6 device's temporary address as subjectAltName. We explain a prototype of the proposed protocol. It is developed with FreeBSD, KAME SNAP, racoon, and OpenSSL. When the IPv6 device is turned on, it automatically requests and receives an anonymous public-key certificate from the CA and then it becomes ready to communicate with another IPv6 device via IPsec with IKE using a temporary address; hence, it is plug-and-play.

1 Introduction

IP version 6 (IPv6) is a new version of the Internet Protocol (IP) [RFC2460]. An IPv6 address has 128-bit length and the plug-and-play feature will release IPv6 device users from the device configuration with respect to networking. Therefore, future IPv6 devices will include a large number of, and a wide variety of devices in addition to personal computers; e.g., sensors, refrigerators, video recorders, cameras, printers, cellular phones, cars, etc.

IPv6 devices must support IPsec [RFC2401]. IPsec provides security services (confidentiality and integrity of communication over the Internet) at the IP layer. The default automated management protocol for IPsec is IKE (Internet Key Exchange) [RFC2409]. When an IPv6 device communicates with another device using IPsec with IKE, a pre-shared key or devices' certificates need to be used. It is obviously impossible for any pair of devices to share in advance a pre-shared key, so that IKE using certificates will be one general solution with scalability.

An IPv6 device can use temporary addresses to have its activities not to be linked. A typical IPv6 address consists of a 64-bit subnet prefix and a 64-bit interface identifier (ID) of the interface. Since a temporary address contains a seemingly random interface ID, a third party on the Internet may not be able to decide whether multiple activities using different temporary addresses are made by the same IPv6 device.

We consider how to realize IPsec using temporary addresses. If an entity (an IPv6 device or the user) reveals its identity to the communication partner (peer), then IKE using temporary addresses and certificates is a trivial solution. However, there are cases where an entity needs to hide its identity (i.e., be anonymous) even to the peer. For example, an e-commerce customer wants to be anonymous to avoid receiving junk mails, phishing attacks make people doubtful that a peer across the Internet is untrustworthy, so that they are reluctant to tell their personal data to the peer, and a whistle blower never reveal his or her identity to avoid potential danger. Simultaneously, an e-commerce company may not want to possess privacy-related data of customers because possession of such sensitive data increases the cost of operation. Thus, in some cases, being anonymous to the peer can be preferable (even for both peers). In such a case, ordinary certificates that include identity of the certificate holders must not be used, and the authenticity of the entity needs to be confirmed by other method to prevent the man-in-the-middle-attack.

In this paper, we examine the problem from the viewpoints of IPsec and IKE specifications and anonymity, and propose a solution to the problem. Such a solution is referred to as *Anonymous IPsec*. We consider that identification-based authentication is not necessarily required at the IP layer. Instead, an X.509v3 anonymous public-key certificate that includes an IPv6 address as subjectAltName is used in our solution. The certificate conveys authenticity of an anonymous IKE initiator, and that makes IKE responder possible to decide whether the access is allowed or not. We study role of the Certification Authority (CA) and consider deployment of the CAs. In addition, we explain a prototype of certificate issuing protocol of the proposed anonymous IPsec. It is developed with FreeBSD, KAME SNAP, racoon, and OpenSSL. When the IPv6 device

* Canon Inc. 30-2, Shimomaruko 3-Chome, Ohta-ku, Tokyo 146-8501, Japan

is turned on, it automatically requires and receives an anonymous public-key certificate from the CA and then it becomes ready to communicate with another IPv6 device via IPsec with IKE using a temporary address; hence, it is plug-and-play. We show with our prototype that anonymous IPsec does works over the Internet.

In section 2, basic mechanisms of IPv6 and notions of anonymity are reviewed. In section 3, the problem and solution are considered. In section 4, proposed protocol for anonymous IPsec is explained. In section 5, our prototype is explained, and we discuss our prototype experiment in section 6. Conclusions are presented in the final section.

2 Preliminaries

In this section we review IPv6, temporary addresses, IPsec, and IKE. Next, we introduce some notions with respect to anonymity and cryptography.

2.1 Extended IPv6 Stateless Address Autoconfiguration

An aggregatable global unicast address usually consists of a 64-bit subnet prefix and the 64-bit interface ID of the interface. Interface IDs are used to identify interfaces on a link and an interface ID is usually constructed in Modified EUI-64 format from its IEEE 802 48 bit MAC identifier.

When a node (a device implements IP) is turned on or rebooted, the node (either host or router) begins stateless address autoconfiguration process [RFC2462]. It first generates a link-local unicast address and confirms the uniqueness on the link, then multicasts Router Solicitation message (RS) to obtain a prefix of global address. A router periodically sends Router Advertisement messages (RA) which may contain Prefix Information options. If a router receives RS sent from a host, then the router returns RA, so that a host may quickly get RA. Upon receiving the RA, then the host obtains a subnet prefix from the option (if any), generates a global address by appending the interface ID to the prefix, and sets the router as its default router. Therefore, the host, using the global address and the default router, can communicate with other nodes on the Internet.

There is a privacy issue in the stateless address autoconfiguration, that is, an interface ID constructed in Modified EUI-64 format from an embedded IEEE 802 48 bit MAC identifier is always the same; hence, it can be used to monitor activities of the same node (or its user). A privacy extension scheme [RFC3041] may solve the issue. The approach is to change the interface ID portion of an IPv6 address over time and generate new addresses from the randomized interface IDs. In the extended stateless address autoconfiguration, since a node has a Modified EUI-64 format interface ID and a set of randomized interface IDs, two kinds of global addresses are generated. They are called *public address* and *temporary address*, respectively. Since determining whether two temporary addresses correspond to the same node is supposed to be difficult, temporary addresses can be given preference over public addresses when the node initiates outgoing communication. The default frequency at which temporary addresses are regenerated is one day.

2.2 IPsec

IPsec [RFC2401] is a security architecture for IP (both IPv4 and IPv6). Specifically, IPsec provides confidentiality and integrity of communication over the Internet at the IP layer and IPv6 nodes must support IPsec. IPsec has two protocols; AH (Authenticating Header) and ESP (Encapsulating Security Payload), and two modes; transport and tunnel.

A fundamental concept in IPsec is Security Associations (SAs). A SA is a simplex connection that affords security services to the traffic carried by it. A SA is uniquely identified by a triple consisting of a Security Parameter Index (SPI), an IP destination address, and a security protocol (AH or ESP) identifier. Negotiating SA(s) is necessary for nodes to communicate with IPsec. Key management is also fundamental to IPsec. Authentication and confidentiality offered by IPsec depend on a key shared by the communicating nodes, so that the cryptographic key used by the nodes must be securely shared.

2.3 IKE

The SA and key management can be done manually or automatically. For widespread deployment and use of IPsec, scalable automated management protocols are required. The default automated management protocol selected for use with IPsec is IKE.

In IKE, Diffie-Hellman key-exchange protocol is executed by the communicating nodes to share a secret-key, so that each node needs to confirm the authenticity of the other node to prevent the man-in-the-middle-attack. Four different authentication methods are defined; pre-shared key, digital signature, or two forms of authentication with public-key encryption.

It is obviously impossible for any pair of nodes to share in advance a pre-shared key, so that pre-shared key based authentication works for small groups only. Other three methods may have scalability but they additionally need authentication of public-keys. A solution to the pubic-key authentication in IKE is public-key certificates. The format is specified in [RFC3280] and referred to as X.509v3.

2.4 Notions of Anonymity

For later discussion, it is necessary to classify notions of anonymity from the viewpoints of protocol and levels. Suppose a protocol is executed by multiple entities, for example, a consumer, a shop, and a bank in an online check system conduct a payment protocol on the Internet. We consider privacy of a target entity, i.e., a consumer, against adversary (non-target entities), i.e., shops and the bank. We assume that the adversary can monitor every data sent to and received from the target entity during protocol executions.

If a target entity is not identified by adversary after a run of the protocol, then we call that anonymity (of

target entity) is preserved in the protocol. If the protocol is executed multiple times and it is not possible for adversary to determine whether a target entity participated in a run of the protocol and a target entity participated in other run of the protocol are the same entity or not, then we call that unlinkability (of target entity) is preserved in the protocol. Unlinkability and anonymity together may be referred to as untraceability.

In addition, there are two levels in anonymity and unlinkability. If adversary with limited computational power cannot computationally identify a target, then anonymity is computationally preserved. If adversary with unlimited computational power cannot identify a target, then anonymity is unconditionally preserved. The two levels are similarly defined for unlinkability. Note that the security of typical public-key cryptography is based on computational infeasibility of a problem, e.g., RSA depends on the infeasibility of factoring a large composite number.

2.5 Computational Assumptions

From previous research on cryptography, the following assumptions are considered to be reasonable [OMO].

Discrete Logarithm Problem (DLP)
Let G be a finite group and g be a generator of G. The discrete logarithm of $v \in G$ with respect to g is denoted by $\log_g v$. It is infeasible to compute the discrete logarithm if the order of G is sufficiently large.

Comparing Discrete Logarithms Problem (CDLP)
Let r and s be random elements of G. Given g, $v = g^s$, $g' = g^r$, and $v' = g^{r \cdot s}$, if the order of G is sufficiently large and r and s are unknown, it is infeasible to determine whether $\log_g v$ is equal to $\log_{g'} v'$ or not.

3 Problem and Solution

In this section we consider how to realize IPsec using temporary addresses. We examine the problem from the viewpoints of IPsec and IKE specifications and anonymity, and consider the solution.

3.1 Issues on IPsec and IKE specifications

With temporary addresses an entity (IPv6 device or the user) can be *anonymous*. An anonymous entity can be unlinkable by changing the temporary addresses over time. This feature is beneficial to protecting privacy but may be abused by attackers to impersonate node or substitute key of node.

The Internet, a distributed open packet-switched network, does not guarantee that IP datagrams with the same source address originate from the same entity using the address. Different entities may use the same source address at different moment and IP datagrams can be easily modified on the route. As an issue, it is important to emphasize that IPsec authenticates neither remote entities nor IP addresses. It authenticates the IP datagrams as originating from an entity who knows a particular key and it allows to confirm whether integrity of the IP datagrams is not compromised on the route. Entity authentication is achieved by not IPsec but IKE.

If an entity reveals its identity to the communication partner, then IKE using temporary addresses with certificates is a trivial solution. IKE allows the communicating entities to authenticate themselves to each other using public-key encryption or digital signature as an authentication method, and the authenticity of the public-keys used is assured by the certificates. Neither impersonating entity nor substituting key of entity is successful even if an attacker and the target are in the same link. However, because we consider a case where an entity needs to hide its identity even to the peer, ordinary certificates that include identity of the certificate holders must not be used.

[RFC2407] defines the following ID Types; an IP address, a fully-qualified domain name string, a fully-qualified username string, a range of IP addresses, the binary DER encoding of an ASN.1 X.500 Distinguished Name, the binary DER encoding of an ASN.1 X.500 General Name, and a vendor-specific data. For anonymity, a username and a distinguished name must not be used. The possible choice is an IPv6 address, but it is a temporary address and changes over time. This brings another issue; how the CA can assure that a temporary address is really assigned to an entity?

3.2 Problem Reconsidered

Taking the above into account, we reconsider the problem as follows. Due to the Internet structure it is not possible for a remote CA to confirm that a temporary address is really used by an entity. But the primary purpose of authentication in IKE is to disable the man-in-the-middle-attack. If the CA can guarantee that a temporary address is bound to a specific anonymous entity and only the entity can utilize the fact, then that is sufficient for our purpose.

From these observations, it is considered that an anonymous entity with a temporary address needs to be able to show its *consistency* as to using the same temporary address, and it must be impossible for other entities to succeed in neither impersonating the entity nor substituting the entity's key while the entity uses the same temporary address. In other words, a temporary address needs to be bound to an anonymous entity in such a way that another remote entity (peer) can confirm both the binding and the entity's consistency.

3.3 Exclusive Guarantee of Address

The guarantee can be achieved by a CA issuing X.509v3 public-key certificates. The CA issues an X.509v3 public-key certificate that includes a public-key of an entity, a date and time as validity period, and a temporary address as subjectAltName. Identity of the entity in the subject field must be empty. Until the expiration date and time comes, the CA never issue any other certificate including the same temporary address. This guarantees that a temporary address is exclusively bound to

a public-key of an anonymous entity for a period of time and only the entity having the corresponding private-key can utilize the certified public-key. The temporary address is used, as ID type, in the authentication in IKE, that is, authenticity of the address is verified not only by the certificate but also by confirming that it is used as the source address of IP datagrams in IKE. Consequently, the subsequent IPsec communication using the temporary address is tightly linked with the preceding IKE through the address embedded certificate. This linkage enhances the robustness of whole processing of IPsec with IKE using a temporary address.

3.4 Issues on Anonymity

Even if an IPv6 temporary address is bound to an anonymous entity's public-key, the single certificate cannot be a solution since the public-key is unique, i.e., it lacks unlinkability. Therefore, a certificate with anonymity should be used only once so that unlinkability of public-keys in the certificates is preserved. This necessitates that a (unlinkable) public-key must be generated each time a certificate is requested and generation of public-key and issuing of certificate should be as efficient as possible.

In addition, CA's ability should be minimized in order to make solution as sound as possible. That is, CA must not be able to compute certificate users' private-keys and it can be proven that CA generated false certificates of certificate users if it does so.

Furthermore, while anonymity is preserved, an entity should be discouraged from transferring its private-key to the others. This can be satisfied by that if a certificate brings a dispute and the fact is judged by CA as reasonable, then the corresponding user is correctly identified and imposed on a penalty.

3.5 Comparison of Previous Schemes

There are several certificate-based techniques that offer authentication and anonymity; e.g., group signatures [CvH], pseudonymous self-certified keys [PH], anonymous public-key certificates [OMO].

In a group signature scheme with group manager [CS], first a group member receives a membership certificate from the manager, then the member produces group signatures by himself or herself. In order to bind a temporary address to an entity (member), it is the manager who must put the temporary address into the member's certificate. However, the membership certificates are not designed to allow such data inclusion and main purpose of the scheme is to allow members to show their membership without showing the membership certificates. Therefore, such group signature schemes are not suitable for a solution.

A public-key certification scheme is proposed in [PH]. The keys issued by the scheme are referred to as *pseudonymous self-certified keys*. The keys are generated as the blind signatures [Ch] of Schnorr signature scheme, so that it is called pseudonymous (anonymous). Since the usage itself of the key in the cryptographic communication certifies the authenticity of the key, it is referred to as a self-certified key. However, it is not possible for the CA to include a temporary address in the certified key. In addition, since the user is unconditionally anonymous even to the CA, nothing prevents the users from transferring their private-keys to other users; hence, pseudonymous self-certified keys are not suitable for a solution.

Anonymous public-key certificates [OMO][Oi] can protect privacy of the public-key certificate users from third parties. An anonymous public-key certificate of a user includes a random public-key of the user but no information about the user identity. Anonymity and unlinkability are computationally preserved in the scheme of [OMO] and unconditionally preserved in the scheme of [Oi]. Only the user with private-key can utilize the corresponding certificate, and if a certificate is abused and a dispute occurs, then the CA can identify the corresponding user and prove the identification by zero-knowledge proof. Thus, transferring private-keys is suppressed. It is possible for the CA to include in the certificate a temporary address as well as the identity of the CA, its expiration dates, etc., and the issuing process is more efficient than previous schemes in terms of the computation required by the user. If the CA generates a false certificate of an entity, then the entity can prove by zero-knowledge proof that it is a false one generated by the CA. So, the CA's abuse is prevented as well. Thus, anonymous public-key certificates are suitable for a solution.

3.6 Solution

Accordingly, the solution is as follows. A CA issues an X.509v3 entity an anonymous public-key certificate that includes a temporary address as subjectAltName. An entity requests a certificate from the CA by sending a certificate signing request (CSR) that includes the temporary address it uses. A CSR is a digital signature made by the entity on a certificate request message that includes entity's identity and public-key and attribute information such as temporary address. In the IP packet transporting CSR to the CA, the source address is the temporary address. The CA generates and sends an X.509v3 anonymous public-key certificate to the temporary address. The certificate is sent to and confirmed by a peer in IKE. The validity period can be short (e.g., one day), so that revocation mechanism such as certification revocation list can be unnecessary.

3.7 Deployment of CAs

The remaining issue is overlap of different CAs' domains. A model with single CA on the Internet is not practical. If a temporary address simultaneously belongs to different CAs' domains, then the CAs need to cooperate when a certificate including the temporary address is issued because a temporary address must not be bound to different entities at the same moment. If the domains do not overlap, each CA can control issuing all by itself.

IPv6 global unicast addresses format [RFC3587] allows deployment of CAs without overlap. The format is shown in Figure 1.

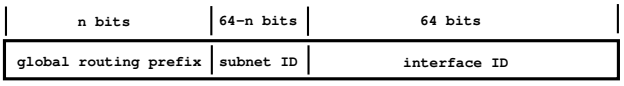

Figure 1: IPv6 global unicast address format

The global routing prefix is designed to be structured hierarchically. Hierarchical structure is necessary to permit aggregation of routing information by ISPs (Internet Services Providers) and to limit the expansion of Internet routing tables [RIPE-267]. This condition is utilized to design reliable hierarchical CA structure as follows.

An exemplified hierarchical structure of the global routing prefix is [RFC2374] which proposes rules for Top-Level Aggregation Identifiers (TLA IDs) and Next-Level Aggregation Identifiers (NLA IDs). TLA IDs are the top level in the routing hierarchy and are assigned to organizations. NLA IDs are used by organizations assigned a TLA ID to create an addressing hierarchy and to identify sites. The sites are administrated by individual organizations assigned NLA IDs. In this structure, an organization assigned a TLA ID can have a CA that issues certificates to the subordinate individual organizations assigned NLA IDs. The global routing prefix, in this example, is organized into two levels, but more levels can be defined as well. Moreover, the structure and arrangement can be extended. Subnet IDs (corresponds to SLA IDs in [RFC2374]) can be used by an individual organization assigned a global routing prefix (corresponds to a NLA ID) to create its own local addressing hierarchy and to identify subnets within the site, so that each individual organization assigned a global routing prefix can similarly have a subordinate CA that issues certificates to the subnet administrators who are assigned subnet IDs.

In this design, the CA hierarchical structure is parallel to the hierarchical structure of the global routing prefix (and subnet ID). The aggregation of routing information properly works if and only if assignments of global routing prefixes (and subnet IDs) to routers at each level are properly managed. Thus, it is possible to assume that each CA at all levels can be managed similarly by a responsible administrator (who may be the same administrator that is responsible for managing a router). Therefore, it is possible to design and allocate CAs in a hierarchical manner so that domains of the CAs do not overlap.

In addition, since the domains correspond to subnet prefixes, if a subordinate CA issues a certificate associated with wrong domain (subnet), the fact is easily detected as follows. In the hierarchy, a CA issues certificates to its subordinate CAs, which also issue certificates to their subordinate CAs, and so on. Thus, certification paths are formed. A certificate includes its subnet prefix and the subnet prefix is inherited from the parent certificate in the certification path. The inheritance is easily confirmed along the certification path by comparing the leftmost bits of the prefix in the certificates. So, a subordinate CA cannot issue a certificate associated with wrong domain without being detected. This inhibits a CA administrator from irresponsible management; hence, reliable management of CAs can be expected.

A CA that issues certificates to its end entities (IPv6 devices or the users who use temporary addresses) may be a node assigned a prefix of 32-bit, 48-bit, or 64-bit (or an intermediate length), and the node may be a dedicated node or a router. Taking these into account, the following may be a typical deployment. A root CA issues certificates to organizations assigned a 32-bit prefix and each certificate includes the assigned prefix. Each organization assigned a 32-bit prefix manages a subordinate CA, which issues certificates to organizations assigned 48-bit prefixes and each certificate includes the assigned prefix. Each organization assigned a 48-bit prefix manages a subordinate CA, which receives certificate requests from its end entities in the subnet and issues anonymous public-key certificates to the end entities, and each anonymous public-key certificate includes the temporary address used by the end entity. In this deployment, the certification path includes four certificates (and a self-signed certificate of the root CA).

In a certification path, only a certificate issued to an end entity must be an anonymous public-key certificate. Ordinary public-key certificates and ordinary certificate management schemes can be applied to the other certificates in a certification path.

4 Proposed Anonymous IPsec

In this section, we propose an anonymous IPsec. First, an anonymous public-key certificate scheme is explained. Next, a protocol in which a host requests and receives an anonymous public-key certificate from the CA is explained. The security is discussed.

4.1 Anonymous Public-key Certificates with DSA

An anonymous public-key certificate scheme can be implemented with public-key cryptosystems that depend on the intractability of DLP such as ElGamal, DSA (Digital Signature Algorithm), and Schnorr. Our prototype uses DSA as the public-key signature scheme used by IKE initiator.

Let p, q be primes such that $q|p-1$, g be a generator of order q, and $\mathcal{H}$ be a one-way hash function. They conform to DSA and are shared by users and CA.

Step-0 An entity U_i randomly chooses its private-key $s_i \in \{1, \ldots, q-1\}$, computes the corresponding public-key $v_i = g^{-s_i} \bmod p$, and submits with the CA its identity, the public-key v_i, and other necessary data (e.g., temporary address and expiration date and time). After proper identification[1] and confirmation, the CA accepts the identity, public-key, and other necessary data. The public-key v_i is referred to as *seed public-key*. The CA registers the accepted set of data on its database.

[1] In our prototype, an entity is an IPv6 device and the ordinary public-key certificate issued by the device manufacturer is used by the CA to identify the entity at this step.

Step-1 The CA U_{ca} chooses a random number $r \in \{1, \ldots, q-1\}$ and computes $g' = g^r \bmod p$ and $v_i' = {v_i}^r \bmod p$. Next, U_{ca} makes a signature sig_{ca} on X with its signature scheme, where X includes identity of U_{ca}, expiration date and time, p, q, g', v_i', and the temporary address. Finally, U_{ca} sends to U_i the anonymous public-key certificate (X, sig_{ca}) in X.509v3 format.

Step-2 The entity U_i that received an anonymous public-key certificate (X, sig_{ca}) confirms whether it is a valid signature of U_{ca} on X by the public-key of U_{ca}.

Step-3 U_i makes a signature on a message using DSA.

For a message $m \in Z$, U_i chooses a random number $k \in \{1, \ldots, q-1\}$, and computes $sig_1 = ({g'}^k \bmod p) \bmod q$ and $sig_2 = k^{-1}(\mathcal{H}(m) + s_i \cdot sig_1) \bmod q$. The signature on m made by U_i is (m, sig_1, sig_2). We call $(X, sig_{ca}, m, sig_1, sig_2)$ an *anonymous signature* on a message m.

Step-4 The **Step-1** to **Step-3** is repeated at any time if necessary. Random numbers should be different at each time.

Step-5 If a verifier U_v receives an anonymous signature on a message $(X, sig_{ca}, m, sig_1, sig_2)$, U_v confirms if sig_{ca} is a valid signature of U_{ca} on X by the public-key of U_{ca}. With certified p, q, g', v_i', then U_v confirms

$$sig_1 \stackrel{?}{=} ({g'}^{\mathcal{H}(m) {sig_2}^{-1}} \cdot {v_i'}^{sig_1 {sig_2}^{-1}} \bmod p) \bmod q.$$

If these checks succeed, U_v verifies that the anonymous signature on the message is made by an anonymous entity who is authorized by U_{ca}.

4.2 Proposed Protocol

The proposed protocol for anonymous IPsec is based on the above scheme. In the protocol, a host requests and receives an anonymous public-key certificate from the CA, and the host initiates IKE using a temporary address. We assume that the host knows the CA's IPv6 address[2] and certification paths are already formed outside of the protocol. For simplicity, encryption of CSR sent from a host to a CA is omitted.[3]

1. (At factory) A host generates its private-key and public-key and submits its identity and public-key to the manufacturer. After proper identification, the manufacturer registers the identity and public-key on its database and issues an ordinary public-key certificate to the host. The public-key is referred to as *registered public-key*.

2. (At user's site) When the host is turned on, it generates a seed public-key only for the first time, and it generates a temporary address and assigns it to the interface.

3. The host creates a certificate request message that contains its identity, registered public-key, seed public-key, the temporary address, and its expiration date and time, and generates a signature on the message using the private-key (corresponding to the registered public-key). The message and signature is referred to as CSR (Certificate Signing Request).

4. The host sends to the CA the CSR along with its ordinary public-key certificate. It uses the temporary address as the source address.

5. Upon receiving the CSR and ordinary certificate from the host, the CA checks if the ordinary certificate is valid (as a signature). Then, with the certified registered public-key, it also checks if the CSR is valid. If the ordinary certificate is not valid or the CSR is not valid, the CA terminates. If they are valid, next the CA checks if the message in CSR is correct. In particular, the CA checks if the temporary address is in the CA's domain and if it is already used by another node by looking up the database. If it is not listed on the database, the CA writes it on the database and generates an X.509v3 anonymous public-key certificate in which the temporary address is included as subjectAltName and its expiration date and time is set to the same one in the CSR. The CSR and certificate may be recorded on the CA's database. If the CSR message is not correct, the CA terminates.

6. The CA sends the X.509v3 anonymous public-key certificate to the host by using the temporary address as destination address.

7. The host receives an anonymous public-key certificate and checks if it is valid and correct. If the check is not confirmed, the host terminates.

8. The host as IKE initiator begins IKE using the temporary address. Using DSA, an anonymous signature is made and sent to the IKE responder.

9. When an IKE responder starts IKE with an anonymous IKE initiator and receives an anonymous public-key certificate (in an anonymous signature) from the peer, then the IKE responder confirms the validity of the anonymous signature and its certification path.[4] If the temporary address used as the source address in the IKE packet and the one contained as subjectAltName in the anonymous public-key certificate are not the same, the IKE responder terminates. If the confirmation is completed, the peers begin IPsec using the temporary address.

[2] In our prototype RS and RA are extended to deliver CA's IPv6 address from a router to a host.

[3] Once a host knows the CA's address, it can inquire the CA's public-key and confirm the authenticity, so that encrypting a message sent to a CA can be easily done.

[4] The certification path may be discovered and validated by a trusted server with DPD (Delegated Path Discovery) and DPV (Delegated Path Validation) [RFC3379]. Then, the IKE responder needs to confirm only the validity of an anonymous signature.

10. If the expiration date and time comes, the host generates a new temporary address and repeats the steps 3 to 7.

11. The CA deletes the temporary address from the database if its expiration date and time comes.

4.3 Security of The Proposed Protocol

Security of the proposed protocol depends on the underlying cryptographic schemes and protocol. Anonymity, unlinkability, and unforgeability of signature and certificate must be assured throughout the protocol. The proposed protocol consists of requesting phase, issuing phase, and IKE phase.

A CSR is a digital signature made by an entity on a certificate request message. The temporary address the entity uses, the registered public-key, the seed public-key, and the entity's identity are included in the message, so that they are bound to each other through the signature. So, no one else can forge the CSR as long as the signature scheme is secure; hence, impersonating an entity (by substituting identity), substituting a public-key, or substituting a temporary address are infeasible in the requesting phase. If a CSR is not encrypted by the CA's public-key, the identity of the entity and the temporary address it uses may be known to third party. This compromises the anonymity of the entity.

The scheme of 4.1 used in the issuing phase is supposed to be secure as long as DLP and CDLP are infeasible. That is, anonymity and unlinkability are preserved and it is infeasible to forge a certificate. Even if an anonymous public-key certificate is sent to a wrong address and a wrong entity receives it, the wrong entity cannot obtain the private-key corresponding to the public-key in the certificate as long as DLP is infeasible. It is infeasible for anyone except CA to identify certificate user from a certificate because of CDLP. So, abusing somebody else's certificate does not occur and anonymity is preserved in the issuing phase.

The DSA used by IKE initiator is supposed to be secure as long as DLP is infeasible. So, authentication in IKE is secure and the man-in-the-middle-attack is prevented. Therefore, IKE phase is secure as well.

Anonymity is preserved if an anonymous public-key certificate is used only once. As long as DLP and CDLP are infeasible, unlinkability is preserved if the requesting, issuing, and IKE phases are correctly repeated. Consequently, the proposed protocol is secure.

5 Prototype

We developed a prototype of the proposed protocol to confirm whether the protocol was viable on the Internet.

5.1 Components

The components of the prototype are shown below.

Hardware : IBM PC compatible

Operating System : FreeBSD 4.7-RELEASE

IPv6 & IPsec : KAME SNAP [KAME]

IKE : racoon (included in KAME SNAP)

Cryptography Library : OpenSSL 0.9.6g [OpenSSL]

Certificate Format : X.509v3

Signature Scheme : DSA

There are multiple OS candidates that support IPv6 temporary addresses and IPsec; e.g., FreeBSD, NetBSD, and Windows XP. Windows XP SP1 provides neither ESP (except NULL encryption) nor IKE for IPv6. IPv6 and IPsec of both FreeBSD and NetBSD are based on KAME and it is released in two ways. Official releases of the BSDs are stable, tightly integrated IPv6-ready operating systems and they provide IPv6-ready IPsec. KAME SNAP, as diffs to the BSD release versions, comes with more experimental protocols/APIs support and userland programs; e.g., racoon. Comparing the BSDs, we concluded that FreeBSD was relatively easy to use because lots of useful information are easily available.

KAME SNAP includes the IPv6-ready IKE program, racoon, and KAME's IPv6-ready IPsec and racoon depend on OpenSSL for all cryptographic processing.

OpenSSL is a full-featured, open-source toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols as well as a full-strength general purpose cryptography library [OpenSSL]. OpenSSL supports PKCS # 10 certificate requests, X.509v3 certificates, and DSA.

5.2 Prototype Specification

There are lots of choices in designing a prototype. It is important to implement necessary functions for user friendliness and sufficient functions for protocol to be viable. We want to modify the underlying components as little as possible. So, we make assumption, simplification, and extension as follows.

We make the following assumptions. An IPv6 device has it's own private-key and public-key, and the manufacturer issues and installs a public-key certificate on the device. The manufacturer's public-key certificate may be issued by other CA, and the manufacturer's certificate and the other CA's certificate may be installed on the device as well. These keys and certificates are securely managed in the device and the user does not need to take care of them.

We make simplification as follows. From the above assumptions, identity of an IPv6 device can be physically confirmed and assured by the manufacturer and an ordinary public-key certificate is issued; hence, the identity (i.e., the certificate) can be verified by a third party with the manufacturer's certificate. Thus, anonymous public-key certificate issuer CA does not need to confirm physical identification of the IPv6 devices.

There may be two CA hierarchies in the solution; one for CAs to authenticate devices and issue ordinary public-key certificates, another for CAs to guarantee exclusive use of IPv6 temporary addresses and issue anonymous public-key certificates. Racoon does not support certification path validation and one CA can

essentially represent a CA hierarchy from the viewpoint of its function. Therefore, we replace two CA hierarchies with two CAs.

We make an extension as follows. To the best of our knowledge, current IPv6 RFCs do not provide a standard method to inform host of CA address. Since a router can be configured manually by its administrator or automatically by ISP via DHCPv6 (Dynamic Host Configuration Protocol for IPv6) with prefix delegation [RFC3633], it is reasonable to assume that a CA address would be set on a router. We extended RA and RS so that a router can automatically inform a host of a CA address. This is an extension of IPv6 plug-and-play feature and contributes to user friendliness.

There are multiple standards or proposals that are designed to manage public-key certificates; e.g., PKCS # 10, PKCS # 7, CMP, CRMF, CMC, CMS, SCEP. From the viewpoints of function, purpose of prototype, and OpenSSL support, PKCS # 10 [RFC2986] is adopted as a certificate request format and X.509v3 is adopted as a certificate format.

5.3 Implementation Design Issues

Although the two CAs take different roles, their functions are essentially same with respect to certification. In addition, racoon does not support multiple CAs. Thus, we have, in our prototype implementation, one and only CA that issues to the IPv6 devices both ordinary public-key certificates and anonymous public-key certificates.

In addition, a registered public-key is used as a seed public-key. So, the public-key parameters such as p, q, g, and $\mathcal{H}$ are commonly used in ordinary public-key certificates and anonymous public-key certificates.

These simplifications reduce the implementation complexity originating from two CA hierarchies but does neither lose nor change the functional essence of the protocol.

We define a new option type in both RA and RS, and put a CA address in RA.

TCP is used as the transport protocol between CA and end entity because certificate request and certificate should be delivered reliably. We investigate ICMPv6 (used by RS and RA) and UDP (IKE) but they are less reliable than TCP and not suitable for transporting large data.[5] Encrypting CSR by the CA's public-key is omitted, too.

IPv6 temporary address must be embedded in both certificate request and certificate. OpenSSL 0.9.6g can embed only IPv4 address in them as subjectAltName. We modify it to embed IPv6 address in them.

5.4 Programs developed

We implemented four programs; `rtsold`, `rtadvd`, `apcreqd`, and `apcresd`. The first two are extended from the KAME's original daemons `rtsold` (for RS) and `rtadvd` (for RA) to deliver a CA address from a router to a host, respectively. `apcreqd` is run on a host to request anonymous public-key certificate from CA and `apcresd` is run on the CA to issue an anonymous public-key certificate to the host. `apcreqd` and `apcresd` use TCP.

[5] We should expect that a message to issue certificate may be large, since it may contain multiple certificates (because of certification path).

5.5 Experiment Environment

We prepared two hosts and one CA. Programs of our prototype implementation were installed on the CA and one host that was an IKE initiator with temporary address. Another host is an IKE responder. A registered public-key, its corresponding private-key and certificate, and security policy were installed and racoon was configured to automatically run on each host. CA's public-key and corresponding private-key were installed on the CA. For saving space, the CA worked also as a router. We created two sites across the Internet and located the router & CA and one host in a site and another host in another site. One site was connected with an ADSL link to the Internet and another site was connected with Metro Ethernet link to the Internet. The local subnets in the sites were connected with 10/100-base T. The certificate request and certificate were delivered in a local subnet at a site (between the router and the host) and the two hosts communicated via IPsec with IKE over the Internet. The environment is shown in Figure 2.

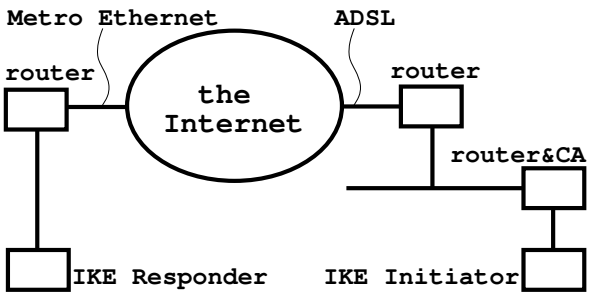

Figure 2: The experiment environment

6 Result

We confirmed that the prototype implementation worked. It is possible to configure the programs so that they automatically work when the host is turned on. Thus, when the host is turned on, it automatically requests and receives an anonymous public-key certificate and becomes ready to communicate with other host via IPsec with IKE using an IPv6 temporary address and the anonymous public-key certificate. This is considered to be a sufficient plug-and-play feature for anonymous IPsec. The upperlayer communication protocols confirmed include ping6 (ICMPv6), ssh (TCP), and Canon original protocol used by the network camera (WebView).

Attention should be paid to security policy. IPv6 nodes use the Neighbor Discovery (ND) protocol to actively keep track of which neighbors are reachable and which are not, and to detect changed link-layer addresses. If an IKE responder specifies the peer's address as `::/0` in the policy, then IPsec is applied to every communication between the node and any peer

including its default router; hence, the ND protocol may be interrupted unless the corresponding IPsec configuration is set on the router. An ad-hoc solution is to limit the peers' addresses.

7 Conclusion

We have considered how to realize IPsec with IKE using temporary addresses. We have examined the problem from the viewpoints of IPsec and IKE specifications and anonymity, and have proposed a solution that uses anonymous public-key certificates. Such a solution is referred to as *Anonymous IPsec*. In our solution, a temporary address to be used by an entity is included as subjectAltName in the entity's X.509v3 anonymous public-key certificate, and the temporary address will not be included in other certificates until it expires; hence, exclusive use of a temporary address by an entity is guaranteed by the CA and only the entity having the corresponding private-key can use the certified unlinkable public-key. The role and deployment of the CAs were discussed. In addition, we have developed a prototype of certificate requesting and issuing protocol of our anonymous IPsec. It is developed with FreeBSD, KAME SNAP, racoon, and OpenSSL. When the IPv6 device is turned on, it automatically requests and receives an anonymous public-key certificate from the CA and then it becomes ready to communicate with another IPv6 device via IPsec with IKE using a temporary address; hence, it is plug-and-play.

References

[Ch] D. Chaum, "Blind signatures for untraceable payments," *Advances in Cryptology – Proceedings of Crypto '82*, pp. 199–203, Plenum Press, 1982.

[CS] J. Camenisch, M. Stadler, "Efficient group signature schemes for large groups," *Advances in Cryptology – CRYPTO '97*, pp. 410–424, Springer-Verlag, 1997.

[CvH] D. Chaum, E. van Heyst, "Group signatures," *Advances in Cryptology – EUROCRYPT '91*, pp. 257–265, Springer-Verlag, 1991.

[KAME] `kame-20030113-freebsd47-snap.tgz` available at `http://www.kame.net/` (as of 2003).

[Oi] K. Oishi, "Unconditionally anonymous public-key certificates, " *The 2000 Symposium on Cryptography and Information Security*, SCIS2000-C32, 2000.

[OMO] K. Oishi, M. Mambo, E. Okamoto, "Anonymous public key certificates and their applications," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, **E81-A**, 1, pp. 56–64, 1998.

[OpenSSL] `http://www.openssl.org/`

[PH] H. Petersen, P. Horster, "Self-certified keys – concepts and applications," *Proc. Communications and Multimedia Security '97*, pp. 102–116, Chapman & Hall, 1997.

[RFC2374] R. Hinden, M. O'Dell, S. Deering, "An IPv6 Aggregatable Global Unicast Address Format," *RFC2374*, July 1998.

[RFC2401] S. Kent, R. Atkinson, "Security Architecture for the Internet Protocol," *RFC2401*, November 1998.

[RFC2407] D. Piper, "The Internet IP Security Domain of Interpretation for ISAKMP," *RFC2407*, November 1998.

[RFC2409] D. Harkins, D. Carrel, "The Internet Key Exchange (IKE)," *RFC2409*, November 1998.

[RFC2460] S. Deering, "Internet Protocol, Version 6 (IPv6) Specification," *RFC2460*, December 1998.

[RFC2462] S. Thomson, T. Narten, "IPv6 Stateless Address Autoconfiguration, " *RFC2462*, December 1998.

[RFC2986] M. Nystrom, B. Kaliski, "PKCS # 10: Certification Request Syntax Specification Version 1.7, " *RFC2986*, November 2000.

[RFC3041] T. Narten, R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6," *RFC3041*, January 2001.

[RFC3280] R. Housley, W. Polk, W. Ford, D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," *RFC3280*, April 2002.

[RFC3379] D. Pinkas, R. Housley, "Delegated Path Validation and Delegated Path Discovery Protocol Requirements," *RFC3379*, September 2002.

[RFC3513] R. Hinden, S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture," *RFC3513*, April 2003.

[RFC3587] R. Hinden, S. Deering, E. Nordmark, "IPv6 Global Unicast Address Format," *RFC3587*, August 2003.

[RFC3633] O. Troan, R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6," *RFC3633*, December 2003.

[RIPE-267] APNIC, ARIN, RIPE NCC, "IPv6 Address Allocation and Assignment Policy," Document ID: ripe-267, January 2003.