A Security Architecture for Mobility-Related Services

Robert C. Chalmers (robertc@cs.ucsb.edu) and Kevin C. Almeroth (almeroth.cs.ucsb.edu)
Department of Computer Science, University of California, Santa Barbara, California 93106

Abstract. In future wireless networks, mobility-related services, such as Candidate Access Router Discovery, will play a significant role in realizing truly ubiquitous, seamless connectivity. In order for these services to be realized, however, their particular security concerns must be addressed. Moreover, the security solution must be flexible and highly configurable in order to meet the demands of inter-domain roaming agreements. In this paper, we explore a number of alternatives and present a general architecture, iARSec, that provides both authentication as well as explicit authorization for services running between neighboring access routers.

Keywords: Security, Mobility management, Smooth handovers

1. Introduction

With the success of current cellular systems, mobile users have come to expect ubiquitous, seamless access to telephone services. Cellular systems have succeeded in providing seamless mobility due, in part, to the heavily engineered architecture of the local access network. In cellular access networks, a number of agents work in concert to track the current location of a mobile device, and to subsequently alter the routing state such that voice calls experience limited disruption or degradation despite frequent handovers [27, 4].

As the nature of mobile computing evolves to include more data-centric applications like e-mail and web browsing, future cellular architectures are targeting packet-based access networks which will employ the next generation Internet Protocol (IPv6) [7] and integrate directly with the larger Internet. The guiding philosophy of the Internet, however, advocates limiting complexity within the network itself [24]. Rather, complexity is pushed to end-devices, freeing the network to simply route packets.

This end-to-end philosophy has expressed itself in the design of Mobile IP [22], the standard solution for mobility management in the Internet. In Mobile IP, the location of a particular mobile node (MN) is maintained by an agent in the mobile's home network. As the mobile node moves through an access network, end-to-end signaling is required between the mobile and its home agent. This approach differs from that of current cellular architectures where signaling remains primarily within the local access network. The end result is that a system based on Mobile IP cannot attain the same level of seamlessness that users have come to expect from mobile phones [5, 17]. To overcome this inherent problem with Mobile IP, a number of proposals [18, 16, 19] are borrowing from the experience of cellular architectures by introducing local services into the access network in an effort to improve the seamlessness of MN handovers.

Security is a fundamental problem faced by each of these new proposals, especially when services span administrative domains. Messages exchanged between access routers (AR) must be protected to ensure authenticity and integrity. Without such protection, malicious nodes could impersonate neighboring routers or modify protocol messages in transit, directly affecting the support offered to mobile nodes. Most, if not all, protocol proposals to date assume that security associations (SA) between neighboring routers already exist, setup by some external means. In other words, it is not the responsibility of the individual services to prepare keys or enforce particular security policies. As is common with many new technologies, the necessity of security is recognized, but an actual solution is considered to be out of scope. In order for these services to become a reality, however, we must address this specific issue of establishing security associations (SA) between neighboring access routers.

Many authors of these proposed services mention the use of static shared keys as one possible solution. This solution may be applicable within a single domain since an administrator could configure each router with the appropriate shared key. For exchanges between domains, however, a static shared key presents a serious configuration challenge and is simply not a viable solution. The next logical approach to establishing SAs might be to employ digital certificates. Certificates are popular specifically because they ease administrative demands, especially for large, disperse configurations. Authentication based on digital certificates, however, generally requires a common Public-Key Infrastructure (PKI) in order to validate the individual certificates [26], and efforts to deploy a global PKI have met with limited success.

Whether or not a global PKI were available, authentication is only one component of an applicable solution to the problem of establishing SAs between cooperating access routers. To authenticate a peer is only part of the problem. Services must also be capable of ensuring that a given peer is indeed authorized to participate. Within a single domain, possession of a shared key may inherently provide such proof, but for more advanced relationships, authorization must be explicit.

Access networks already provide authorization services for mobile nodes. Separate domains form *roaming agreements* which allow mobile nodes from one domain to use the network and services of another. As part of these roaming agreements, the peer domains must establish at least one security association between the specific network entities that perform authorization. We propose to leverage this single security association to dynamically generate SAs for peering access routers. Our architecture, *iARSec* – inter-AR Security, provides not only authentication but explicit authorization for services operating between access routers. The primary component of iARSec is a module running on each participating access router which offers a common interface to all inter-AR services. This interface allows services to dynamically initiate and maintain security associations with peer routers.

The remainder of the paper is organized as follows. We begin in Section 2 with a more general look at the problem of establishing SAs between neighboring ARs. We define a generic architecture that models the relevant solution space in Section 3. In Section 4, we present our prototype implementation, iARSec. We compare alternative solutions in Section 5 in terms of the total delay introduced by authorization and authentication. We conclude the paper in Section 6.

2. Background

The general problem of network security has received significant attention in both academia and industry. While most academic studies consider specific aspects of security, such as cryptography, industry is concerned, in most part, with integrating security services in order to effectively administer operational networks. Our work attempts to bridge this gap by taking an analytical view of the problem of integrating security protocols.

In this paper, our scope is limited to enabling security for access routers that offer mobility-related services to mobile devices. Example services include Fast Mobile IP [18], Context Transfers [16] and Candidate Access Router Discovery (CARD) [19, 28]. This collection of services exhibit unique properties that make security difficult. In particular, relationships between ARs are usually initiated by the handover behavior of mobile nodes. For instance, in CARD, access routers collect information describing which services are available in the immediate neighborhood in order to aid mobile nodes in selecting the best path through the network. In order to accomplish this, access routers exchange capabilities describing their own offered services [28]. This implies that peering ARs must be able to trust one another and protect the signaling between them. The problem of securing inter-AR communication can be decomposed into two primary components: 1) an administrative infrastructure which allows two domains to enforce authorization policies and exchange security parameters; and 2) a protocol operating between peering ARs that leverages the information gathered through the former infrastructure in order to mutually authenticate and authorize each peer.

The administrative infrastructure mentioned above is often referred to as an Authentication, Authorization and Accounting (AAA) service. For mobile access networks, AAA services are usually focused on authentication and authorization for mobile devices. In cellular systems, the Home Location Register (HLR) maintains a mobile device's profile, including all pertinent security material. As a mobile moves into a visited network, the local Visited Location Register (VLR) communicates across domains with the HLR in order to authenticate the mobile, as well as authorize use of the local network [3]. Security between the HLR and VLR is a bi-product of the *roaming agreement* between the two network providers.

The Internet Engineering Task Force (IETF) recently standardized Diameter [6], the next generation AAA architecture for IP-based services. Diameter's predecessor, Radius, has seen significant deployment in IP-based access networks, such as Internet Service Providers (ISP) supporting dial-up users [20]. As future cellular systems transition to an all-IP infrastructure, Diameter and IP-based security will begin to play an ever larger role in authenticating and authorizing mobile users. Many researchers have advocated integrating Mobile IP [22] with a AAA infrastructure in order to enable authorization of MNs moving through foreign IP networks [23, 11]. As one might expect, the proposed architectures are similar to the use of HLRs and VLRs in current cellular systems.

Regardless of which specific AAA architecture is implemented, our goal is to leverage the existing infrastructure in order to provide the necessary security parameters to ARs so that they can authorize and authenticate one another directly, as well as negotiate an appropriate session key. A number of alternatives have been proposed for key negotiation protocols. The simplest is a Diffie-Hellman (DH) exchange where two peers exchange public values that are derived from individually chosen private keys [8]. A single shared key can then be computed independently by each peer.[1] DH exchanges, however, are anonymous. They cannot be used for authentication since the public values are not associated with the identity of either peer.

[1] Diffie-Hellman exchanges can be computationally intensive since they involve exponentiating large numbers.

The Internet Key Exchange (IKE) protocol [10] is currently the standard key negotiation protocol for IP-based services. IKE employs an initial DH exchange to protect the key negotiation, but employs other means to perform authentication such as shared keys, public keys or certificates. In recent years, IKE has received growing criticism for being overly complex. A new version of IKE, IKEv2, is currently being developed in the IETF [15]. Other protocols have also been proposed in the literature that attempt to reduce the complexity of establishing SAs. Just Fast Keying (JFK) [2], for instance, provides the minimal functionality necessary to negotiate a key, but does not provide mechanisms to re-key or generate additional SAs.

2.1. Problem Definition

To date, the literature has not addressed the specific problem of providing security for mobility-related protocols operating between access routers. Security mechanisms within individual protocols have been addressed [25], but protocol designs traditionally assume that an initial security association already exists between peering ARs. This assumption is due primarily to the fact that these protocols, and the issue of IP mobility in general, are still in a developmental stage. The majority of focus has been on early deployment within a single domain where initial security relationships can be manually configured.

In this paper, we address the problem of providing trust between two ARs which may be located in different administrative domains. The distinguishing characteristic of our problem stems from the fact that mobility-related protocols at the AR act on behalf of mobile nodes. In protocols such as CARD, the mobile node provides the address of the neighboring router as part of its signaling with the current AR [28].[2] Two cooperating access routers may have no prior knowledge of one another. In a sense, they first *meet* in response to the hand-over behavior of mobile nodes. Thus, a dynamic, inter-domain solution is necessary.

2.2. Solution Requirements

With the problem well defined, our next step is to enumerate the requirements of an acceptable solution. The first and most apparent requirement is that the solution must provide a means for two access routers to negotiate a security association that, in turn, allows them to protect subsequent protocol signaling. The level of protection should be configurable, including authenticity, integrity, privacy and replay

[2] Network-controlled signaling is also possible, but in this case neighboring routers are likely to have an existing security association.

protection. The solution should support multiple methods for mutual authentication, including shared keys, public keys and certificates. In addition to authentication, the solution must provide explicit authorization for each peer to engage in a particular service. This is crucial since these services exchange possibly sensitive information concerning the AR's own capabilities and the state of the mobile node.

Possibly the most important requirement to this specific problem is that the process of deriving a security association between two peering ARs must not depend on any existing direct relationship between the peers. In other words, the two routers are assumed to be strangers. What we do require, however, is that the two domains in which the ARs reside have some form of service-level or roaming agreement that can be directly or indirectly leveraged in order to dynamically build an SA between the two access routers. In the case that both peers are within the same domain, this requirement is trivial. For the more general inter-domain case, a number of solutions are possible. We pursue these solutions further in the following section with the presentation of a general model.

3. A General Model

The fundamental issue is how to best leverage a high-level relationship between domains in order to determine dynamically which ARs can peer. We can approach the problem as one of *transitive trust*. In our general model, each domain authorizes its own access routers to peer with specific neighboring domains. The trust between the two domains, as expressed through the roaming agreement, allows us to translate local authorization into mutual authorization. In other words, since each domain trusts their local router and the domains trust one another, the two ARs can, in turn, establish a mutual trust. Once trust is achieved, the access routers can authenticate each other directly.

To better encapsulate this concept of local authorization, we introduce a data structure specific to each access router, the *Security Profile*. The Security Profile determines with which neighboring domains the AR is authorized to peer for each requested service. In addition to this authorization state, the Security Profile records which authentication methods are allowed, including any necessary keys.

By formulating the model in this way, we now have freedom in where we place the Security Profile for individual access routers. For instance, we could maintain all profiles for local ARs within a domain-wide database, or we could choose to replicate specific profiles in neighboring domains. Next, we present three general scenarios for the placement of profiles. Each scenario differs not only in where profiles are maintained, but also in the effect that placement has on the form of the security relationship between the two domains.

Distributed Scenario: In the *distributed* scenario, each domain maintains its own database of Security Profiles which we will refer to as the *Security Repository*. When an access router wishes to peer with another AR in a neighboring domain, it first authorizes itself with the local Security Repository, passing along the identity of the desired peer. The repository, in turn, contacts the Security Repository in the peer's domain to ensure that the peer is also authorized, as well as to exchange necessary keys. This scenario requires that the security relationship between the two domains take the form of an actual security association between the two Security Repositories, and that communication between the two domains occurs in-line as part of the authorization process.

Replicated Scenario: The *replicated* scenario employs replication to eliminate the necessity of in-line communication. In this scenario, the security relationship between the domains is used to exchange Security Profiles for all pertinent access routers off-line, prior to any authorization requests. Authorization is handled locally within each domain. This improves the worst-case delay of any request, but it also increases maintenance costs and reduces the dynamic nature of the system since changes in the local domain must be replicated out to all neighboring domains affected by the change.

Certified Scenario: In the *certified* scenario, profiles are again maintained locally, but they are first certified. In other words, each profile is signed by the neighboring domain to ensure its authenticity. With this technique, the profiles can be exchanged directly between peers as part of the authentication process. Each peer can guarantee authorization by simply verifying the signature on the profile. This again changes the form of the security relationship between domains. Now, the relationship must take the form of cross-certification or an agreement on a common third-party to perform such certification. With this technique, we eliminate the requirement of a Security Repository, but the result is to fragment the Security Profile since each certified profile becomes specific to a particular domain. Moreover, changes to local configuration must be re-certified by all neighboring domains.

In short, by moving profiles closer to the individual access routers, we reduce the time required to establish new security associations at the cost of manageability. The *distributed* scenario is the most manageable

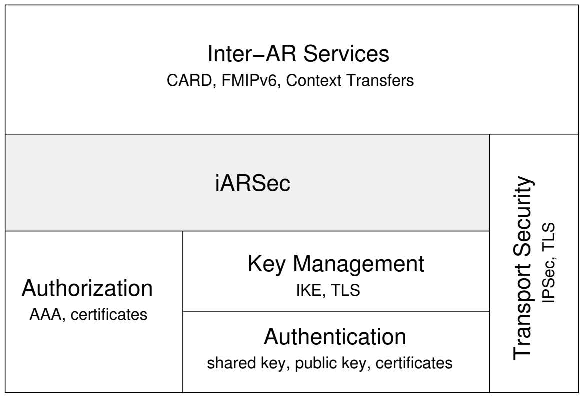

Figure 1. Software components involved in the iARSec architecture.

since each domain maintains it own Security Repository. Any changes, due possibly to reconfiguration of the access network, can be made locally without requiring synchronization with neighboring domains. The cost of this increased manageability, however, is an increased delay experienced by individual requests to authorize a peer; explicit signaling must occur between the two repositories. We take a closer look at the issue of delay in Section 5.

4. Architecture

In this section, we present the iARSec architecture as a specific instance of the general model described in the previous section. Although the architecture supports all three scenarios for placing Security Profiles, our presentation herein focuses on the *distributed* scenario in an effort to keep this section concise. We return to a discussion of all three scenarios in Section 5. In our prototype, we represent Security Repositories as AAA servers which were developed using version 1.0.2 of the Open-Diameter library [21]. To perform authentication and key negotiation, we employ the FreeS/WAN 2.04 implementation of IKE [9], slightly modified FreeS/WAN to support anonymous Diffie-Hellman exchanges.

Figure 1 presents the software components of our prototype architecture. The individual services running at each participating access router are represented by the topmost layer. As mentioned previously, many possible services exist, such as Fast Mobile IP and Context Transfers [18, 16]. Each of these services is specified to require secure transport for messages exchanged between neighboring access routers, specifically message authentication, integrity, replay protection, and possibly encryption. Moreover, each services must be capable of verifying that the peer AR is a valid, authorized participant. Otherwise, malicious nodes could abuse the trust implicit in the services themselves to extract information private to the participating domains, as well as disrupt the services being provided.

Rather than force each individual service to manage the complexity of security associations, iARSec exports a standard interface to all services. Through this interface, a service requests a session with a particular peer, passing the peer's IP address, a service identifier, and optional service parameters such as the desired transport protocol, ports, etc. The end product is a session-level security association established for the particular service which initiated the request. Since multiple services may be operating simultaneously between any two peers, iARSec maintains a cache of active security associations. If a subsequent request is received for an already associated peer, a new session-level SA can be generated directly without performing a separate round of authorization and authentication.

4.1. Identification

For mobility-related services, peers are initially identified an IP address. Using an IP address as an identifier for the purpose of authorization is possible but problematic since each AR can be identified by multiple addresses, at least one per interface. For services that rely on mobile nodes to identify their peers, as is the case for most mobility-related protocols, the reported address is dependent upon the MN's point of attachment (i.e., access point). Different mobiles may report different addresses for the same access router. Employing IP addresses for the purpose of identification requires that all addresses be used interchangeably. This, in turn, results in an unwanted growth in configuration overhead and increases the chance of misconfiguration.

Rather than rely on IP addresses to identify an access router, we assign each AR a unique Network Access Identifier (NAI) [1] which is a text-based identifier specifying both a user and a realm. For example, the NAI, *iarsec@secure.net*, identifies *iarsec* as the user or service in this case, and *secure.net* as the realm. The realm portion of the NAI is used to route messages between AAA servers located in different domains.

With this approach, the first task is to map the IP address to a unique NAI that can be used to perform authorization through AAA. To do this, the initiating peer, AR_i, performs an initial IKE exchange with the responding peer, AR_r using an anonymous Diffie-Hellman exchange (see Figure 2a). As part of this exchange, the two peers include their respective NAIs. The anonymous Diffie-Hellman key pro-

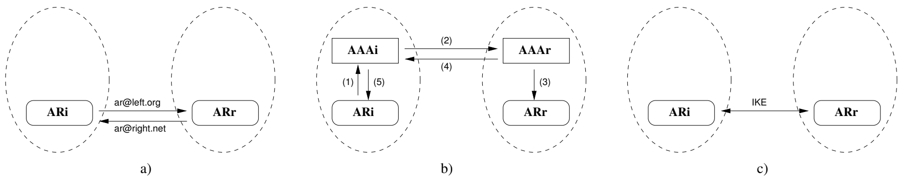

Figure 2. Messages exchanged in iARSec: a) peers initially exchange identities; b) AAA messages are exchanged with local AAA server and between neighboring servers; c) a security association is established directly between peers. Dotted ellipses represent distinct domains.

vides protection for the identities from passive eavesdropping, but does not guarantee against an active attacker collecting or modifying the identities. A modified identity, however, will be discovered and rejected later during the authentication process.

4.2. Authorization

Once AR_i has discovered the identity of AR_r, it initiates a AAA exchange with its local server, AAA_i, as depicted in Figure 2b. The local server maintains a database which determines which access routers are authorized to peer with which neighboring domains for different services. The database indicates which possible authentication modes are suitable, as well as specific shared or public/private keys that should be used when communicating with a particular domain. The AAA server validates the authorization request with respect to its database, ensuring that AR_i is indeed authorized to peer with AR_r's domain.

If local authorization succeeds, the server sends a message to the neighboring AAA server (message 2 in Figure 2b), with the NAIs and IP addresses of the two access routers, a set of proposed services, and any necessary keying material, such as the public key of AR_i. This proposal leverages the security association between the two domains. Thus, the receiving server, AAA_r, can ensure that the request to peer originates from a valid domain, and that AR_i has already been authorized to participate. AAA_r performs a similar authorization check for AR_r, and determines which of the proposed services will be allowed. If successful, the server issues two messages: 3) a local notification to AR_r with the details of the peering relationship, and 4) a reply to AAA_i with the accepted set of services. Finally, the local AAA server replies to the initiating access router with the necessary details to proceed with authentication.

4.3. Authentication

Once authorization completes, both peers have the necessary keys to initiate authentication and form the primary security association. At this point authentication is fairly straightforward. An existing key establishment protocol, such as IKE, can be used without modification. The NAIs should be used for identification to ensure the original exchange was not corrupted by an active attacker.

When selecting which keys to use for authentication, administrators have three primary options: shared keys, public keys, and certificates. Shared keys may be useful within a domain where iARSec is deployed with static configurations rather than explicit AAA messaging. Public keys or certificates are a more scalable solution for inter-domain authentication. Public keys are distributed as part of the AAA exchange. Certificates, however, can be exchanged directly between ARs. For this to work, however, the two domains must share a common Certificate Authority (CA) or cross-sign each other's certificates [26].

5. Discussion

Although the prototype architecture, as presented in the previous section, focuses on the *distributed* scenario, iARSec does support all three of the general scenarios described in Section 3. In this section, we take a closer look at how the alternate scenarios are achieved in iARSec and compare the three alternatives in relation to the total delay required to establish an initial security association. We conclude the section with a discussion of the impact caching has on the overall delay experienced by individual services.

5.1. Delay Components

The first step is to identify the source of delay in each of the alternative scenarios. We define six variables in Table I to represent the delay introduced by the different messages exchanged (i.e., round-trip times), as well as processing delays incurred due to cryptographic operations. This classification makes an implicit assumption that similar messages in differing domains incur similar delays. For example, one would expect that a request to a local repository would experience similar delays (r_l) for both peers since signaling remains within each local domain. Signaling between repositories, on the other hand, must traverse domains and should result in more significant delays (r_r).

Table II provides estimated values for each of the delays listed in Table I. Inter-domain round-trip times were measured between ma-

Table I. Round-trip and processing delays introduced by messages between components of the architecture.

Delay	Description and Endpoints
r_l	round-trip delay between AR and local Security Repository
r_n	round-trip delay between neighboring Security Repositories
r_p	round-trip delay between peering ARs
p_h	processing delay for Diffie-Hellman exponentiation
p_e	processing delay for public-key encryption
p_d	processing delay for public-key decryption

chines located in UC Santa Barbara and Georgia Tech. It should be noted that round-trip times are difficult to estimate since exact values depend on a number of factors, including the internal topology of each domain, the distance between domains in terms of backbone networks, as well as the load along each path. The round-trip values in Table II represent ball-park figures that should be considered only in terms of their relative differences rather than as absolute values. The key point is that signaling between domains (r_r and r_p) can incur significantly more delay than signaling within a single domain. In this case, the two differ by more than two orders of magnitude. Also notice that r_p is represented as an inter-domain delay since inter-AR signaling is generally assumed to occur across the backbone network rather than the wireless link.

Processing delays for cryptographic operations were measured from our prototype. To perform the measurements, we instrumented our IKE implementation which ran between two machines, a 1.7 GHz P4 with 768 MB of memory and a 1.6 GHz P4 laptop with 500 MB of memory,

Table II. Measured round-trip and processing delays in milliseconds averaged over ten trials.

Delay	Average	Std Dev	Min	Max
r_l	0.231	0.072	0.276	0.488
r_n	55.042	0.256	54.800	55.365
r_p	55.150	0.158	54.866	55.309
p_h	7.494	0.387	7.095	8.640
p_e	7.882	0.167	7.669	8.098
p_d	0.295	0.006	0.290	0.307

both running Linux 2.6.5. Each test employed a 1024-bit *modp* group (second Oakley Group [10]) for DH exponentiation, as well as 1024-bit RSA keys. As can be seen, public-key encryption is far more costly than decryption. RSA encryption and DH exponentiation, however, have similar costs.

The total delay can be decomposed into the two distinct phases of authorization and authentication. We start by computing the authorization delay (D_Z) for each scenario. Then, we include the authentication delay (D_C), and compute the total delay (D). For communication between access routers and their local Security Repositories, as well as between different Repositories, we assume that SAs have already been established. Thus, only symmetric cryptography is required during authorization.

5.2. Authorization Delay

For the *distributed* scenario, mutual authorization requires a total delay of

$$D_{Z1} = r_l + r_n. \tag{1}$$

The initial r_l term represents the round-trip between the initiating access router, AR_i, and its local AAA server, AAA_i (see Figure 2b). The exchange between the two AAA servers adds r_n. We do not include delay between the responding server, AAA_r, and its access router (message 3 in Figure 2b) since the message is sent in parallel with the response to AAA_i; we can safely assume that $\frac{1}{2}r_l \ll r_n + \frac{1}{2}r_l$ even though the two r_l terms may not be identical in each domain.

In the *replicated* scenario, the domains exchange Security Profiles off-line.[3] Authorization is performed independently by each AAA server, thus the delay is simply the time required for each access router to query its local server:

$$D_{Z2} = 2r_l. \tag{2}$$

In this case, the more costly inter-domain messages are unnecessary. The authentication delay could be further reduced to r_l if both access routers queried their local AAA servers in parallel. In our prototype, however, we chose to delay the responder's query until after the authentication phase has begun. This has two benefits: 1) the initiator is forced to commit significant resources to the authentication process before the responder performs a AAA query, and 2) the responder is not required to distinguish between *replicated* and *distributed* scenarios. In the *distributed* scenario, the appropriate security information will be

[3] The local, replicated copy could also be the result of caching a previous *distributed* request.

pushed to the responding AR. If authentication begins and the appropriate information is not available, as would be the case in a *replicated* scenario, AR_r queries its local AAA server to resolve the appropriate security parameters. This delays AR_rs commitment of resources which, in turn, reduces the efficacy of a Denial of Service (DoS) attack.

In the final, *certified*, scenario, Security Profiles are certified by each neighboring domain. This can be achieved in iARSec by encoding the authorized services as Certificate Policy extensions of X.509 certificates [26]. In this way, all necessary certificates can be present at each access router without requiring a domain-wide Security Repository. Thus, authorization becomes integrated with authentication, and no extra messages are necessary:

$$D_{Z3} = 0. \tag{3}$$

5.3. Authentication Delay

Once the necessary keys are in place, authentication occurs directly between the two peering ARs. Using IKE, authentication and the generation of an initial security association requires three two-way exchanges or six messages ($3r_p$). Also, each peer must perform two Diffie-Hellman exponentiations ($4p_h$): one to generate a private key, and another to compute the shared key. If public-keys (or certificates) are employed for authentication, four additional public-key operations are necessary ($2p_e + 2p_d$): two encryptions and two decryptions. In IKE, the initial SA generated is reserved for use by IKE itself. Service-specific SAs, referred to as *child SAs*, are generated in a subsequent exchanges that each requires three messages ($1\frac{1}{2}r_p$). Thus, authentication for an initial service requires

$$D_C = 4\frac{1}{2}r_p + 4p_h + k_u(2p_e + 2p_d), \tag{4}$$

where k_u is a binary value indicating whether public-key cryptography is employed.

One more source of delay must be addressed. Prior to authorization, the two access routers must first exchange identities. Establishing identity is important for both authorization and authentication, even in the *certified* scenario since the appropriate certificate must be selected based on the peer's domain.[4] A problem with IKE, however, is that identities are exchanged in the final message. This is too late if

[4] Literally speaking, identity is not necessary for authentication, but alternative techniques such as Return Routability [13] are not directly applicable to the problem of mutually authenticating two access routers.

we hope to properly authorize a peer and determine the appropriate authentication mechanism dynamically. Rather than change IKE, we chose to introduce an initial exchange in which the two peers simply share identities. In its simplest form this exchange adds r_p to D_C. If we want to protect the identities from passive listeners, we can employ a anonymous Diffie-Hellman key to encrypt the exchange. This, however, increases the number of messages. Using IKE, this requires three round-trips, increasing D_C by $3r_p$ rather than just r_p. We assume that the same DH keys can be reused for authentication.

5.4. Total Delay

Next, we compute the total delay for each of the three scenarios:

$$\begin{aligned} D_1 &= r_l + r_n + 7\frac{1}{2}r_p + 4p_h + k_u(2p_e + 2p_d) && (5) \\ D_2 &= 2r_l + 7\frac{1}{2}r_p + 4p_h + k_u(2p_e + 2p_d) && (6) \\ D_3 &= 7\frac{1}{2}r_p + 4p_h + 2p_e + 2p_d. && (7) \end{aligned}$$

One result is immediately obvious: the primary cause of delay is due to messaging between the peering ARs. In other words, authorization plays a small overall part in the total delay. We expect that r_l is much less than r_n, but r_n and r_p should be relatively similar since both messages cross domains. Even if the initial exchange of identities was free and we ignored the processing overhead, authentication using IKE is the major contributor to the overall delay.

By replacing IKE with a less heavyweight protocol, such as IKEv2 or JFK, we could reduce the authentication delay slightly. In particular, both IKEv2 and JFK require just two round-trips to create an initial association. Moreover, IKEv2 can also negotiate an initial *child SA* as part of the initial exchange. JFK does not support the concept of *child SAs*, thus all communication between the two entities are protected by a single SA. This is insufficient for our purposes since individual protocols may require different security properties, such as whether to employ encryption. Finally, neither IKEv2 nor JFK provide a mechanism for exchanging identities prior to authentication. So, iARSec would still require an initial DH exchange. Hopefully, future protocols will provide improved support for this rather simple requirement.

5.5. Optimizing Delay

So, how can we reduce the time required to establish a security association for new services? In iARSec, we take two complementary approaches to this problem: caching associations, and mapping addresses. Once an initial association has been formed between two peers, additional service-specific SAs can be created in IKE with just three messages using Quick-Mode. Moreover, rather than just authorize the service that initially requested the association, all authorized services are returned and stored in the cache. In this way, the next service requesting to peer with a given AR does not require reauthorization or reauthentication.

In addition to leveraging the initial association for subsequent requests for the same IP, iARSec maintains a list of alternate, equivalent addresses for each peer. As we stated earlier, one of the interesting properties of inter-AR services is that each access router can have multiple addresses, and peering is usually initiated on behalf of mobile nodes who have a varied view of the AR as an IP node. Therefore, once authentication completes, the two peers exchange a list of equivalent addresses. Requests to peer with one of these equivalent addresses are mapped to the existing association, and new service-specific SAs can be generated as previously described.

With these two approaches, all secondary requests for association can be fulfilled with a total round-trip delay of $D = 1\frac{1}{2}r_p$. This is a significant improvement, but what about the initial association? In fact, most services are very sensitive to delay since their signaling is intended to improve the handover performance of mobile nodes. A preferred solution is to implement a protocol similar to CARD, Candidate Access Router Discovery, that operates continually between an access router and each of it physical neighbors. In CARD, the initial security association can be delayed without adversely affecting the protocol. Once a peer has been *discovered*, other services can form new short-term associations quickly and efficiently.

6. Conclusion

In future wireless networks, mobility-related services will play a crucial role in realizing truly ubiquitous, seamless connectivity. In order for these services to be realized, however, their particular security concerns must be addressed. Specifically, access routers must be able to mutually authorize and authenticate one another prior to protocol signaling. The use of static shared secrets is not a scalable solution within a large access network, let alone between domains. More flexible solutions are necessary.

Although most handovers occur within a domain, we focus on the inter-domain case because it is the more difficult problem, and inter-domain solutions easily map to the intra-domain. As the Wireless Internet evolves, mobile users will encounter an expanding choice of providers offering competing services. Handover between domains should become more prevalent as the network becomes more heterogenous; users will demand the best service available whether or not it comes from their own provider [14, 12]. Seamless handovers between domains will be just as important as those within a single domain.

To ensure seamless handovers, inter-AR protocols must be able to establish security associations quickly and efficiently. Additional delay introduced by the security architecture adversely affects the handover latency experienced by supported mobile nodes. The solution is to amortize the cost of authorization and authentication over a number of services. By caching associations, we can generate service-specific SAs with minimal overhead and in minimum time. Moreover, by managing security associations independently of any particular service, new services can remain simple and be deployed with relative ease.

References

1. Aboba, B. and M. Beadles: 1999, 'The Network Access Identifier'. RFC 2486, Internet Engineering Task Force.
2. Aiello, W., S. Bellovin, M. Blaze, R. Canetti, J. Ioannidis, A. Keromytis, and O. Reingold: 2002, 'Efficient, DoS-Resistent, Secure Key Exchange for Internet Protocols'. In: *Proceedings of Conference on Computer and Communications Security (CCS'02)*. Washington DC, USA.
3. Akyìldìz, I. and J. Ho: 1996, 'On Location Management for Personal Communications Networks'. *IEEE Communications Magazine* **34**(9), 138–45.
4. Akyìldìz, I., J. McNair, J. Ho, H. Usunalioğlu, and W. Wang: 1998, 'Mobility Management in Current and Future Communications Networks'. *IEEE Network* **12**(4), 39–49.
5. Cáceres, R. and V. Padmanabhan: 1998, 'Fast and Scalable Wireless Handoffs in Support of Mobile Internet Audio'. *Mobile Networks and Applications* **3**(4), 351–63.
6. Calhoun, P., J. Loughney, E. Guttman, G. Zorn, and J. Arkko: 2003, 'Diameter Base Protocol'. RFC 3588, Internet Engineering Task Force.
7. Deering, S. E. and R. Hinden: 1998, 'Internet Protocol, Version 6 (IPv6) Specification'. RFC 2460, Internet Engineering Task Force.
8. Diffie, W. and M. Hellman: 1976, 'New Directions in Cryptography'. *IEEE Transactions on Information Theory* **22**(6), 644–54.
9. FreeS/Wan: 2003, 'Linux FreeS/Wan'. FreeS/Wan Project. http://www.freeswan.org.

10. Harkins, D. and D. Carrel: 1998, 'The Internet Key Exchange (IKE)'. RFC 2409, Internet Engineering Task Force.
11. Hasan, D. Singh, S. Zander, M. Kulbach, J. Jähnert, and S. B.: 2002, 'The Design of an Extended AAAC Architecture'. In: *Proceedings of IST Mobile and Wireless Telecommunications Summit*. Thessaloniki, Greece.
12. Henry, P. S. and H. Luo: 2002, 'WiFi: What's Next'. *IEEE Communications Magazine* **40**(12), 66–72.
13. Johnson, D., C. Perkins, and J. Arkko: 2003, 'Mobility Support in IPv6'. Technical Report draft-ietf-mobileip-ipv6-*.txt, Internet Engineering Task Force.
14. Katz, R. H.: 1994, 'Adaptation and Mobility in Wireless Information Systems'. *IEEE Personal Communications* **1**(1), 6–17.
15. Kaufman (Editor), C.: 2004, 'Internet Key Exchange (IKEv2) Protocol'. Technical Report draft-ietf-ipsec-ikev2-*.txt, Internet Engineering Task Force.
16. Kempf (Editor), J.: 2002, 'Problem Description: Reasons for Performing Context Transfers between Nodes in an IP Access Network'. Informational RFC 3374, Internet Engineering Task Force.
17. Koodli, R. and C. Perkins: 2001, 'Fast Handovers and Context Transfers in Mobile Networks'. *ACM Computer Communication Review* **31**(5), 33–47.
18. Koodli (Editor), R.: 2002, 'Fast Handovers for Mobile IPv6'. Technical Report draft-ietf-mobileip-fast-mipv6-*.txt, Internet Engineering Task Force.
19. Liebsch, M. and A. Singh (Editors): 2003, 'Candidate Access Router Discovery'. Technical Report draft-ietf-seamoby-card-protocol-*.txt, Internet Engineering Task Force.
20. Metz, C.: 1999, 'AAA Protocols: Authentication, Authorization, and Accounting for the Internet'. *IEEE Internet Computing* **3**(6), 75–9.
21. Open Diameter: 2003, 'Open Diameter Library'. Open Diameter Project. http://www.opendiameter.org.
22. Perkins, C.: 1997, 'Mobile IP'. *IEEE Communications* **35**(5), 84–99.
23. Perkins, C.: 2000, 'Mobile IP Joins Force with AAA'. *IEEE Personal Communications* **7**(4), 59–61.
24. Saltzer, J., D. Reed, and D. Clark: 1984, 'End-to-end Arguments in System Design'. *ACM Transactions on Computer Systems* **2**(4), 277–88.
25. Shim, E., J. P. Redlich, and R. Gitlin: 2003, 'Secure Candidate Access Router Discovery'. In: *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC'03)*. New Orleans, LA, USA.
26. Stallings, W.: 1999, *Network Security Essentials: Applications and Standards*. Upper Saddle River, NJ: Prentice Hall.
27. Tabbane, S.: 1997, 'Location Management Methods for Third-Generation Mobile Systems'. *IEEE Communications Magazine* **35**(8), 72–8,83–4.
28. Trossen, D., G. Krishnamurthi, H. Chaskar, R. Chalmers, and E. Shim: 2002, 'A Dynamic Protocol for Candidate Access-Router Discovery'. Technical Report draft-trossen-seamoby-dycard-*.txt, Internet Engineering Task Force.