Limitations of IPsec Policy Mechanisms

Jari Arkko and Pekka Nikander

Ericsson Research NomadicLab,
02420 Jorvas, Finland
{Jari.Arkko,Pekka.Nikander}@nomadiclab.com

Abstract. IPsec, while widely implemented, is rarely used for end-to-end protection of application protocols. Instead, it is mainly used today as an "all or nothing" protection for VPNs. In this paper we discuss the structure and shortcomings of the IPsec security policy mechanisms as partial reasons for this situation. We describe our experiences in using IPsec in a number of situations, including IPv6 control protocols, mobility protocols, network management, and multimedia protocols. We conclude that more often than not, the existing policy mechanisms are inadequate. While IPsec is quite effective in authenticating the peer and establishing assurance about its identity, the lack of attention to authorization questions is a root cause of the existing inadequacies. We also claim that the problems are more fundamental than the lack of suitable APIs and management tools. Finally, we present some potential architectural modifications which could improve the situation, and discuss the practical challenges in achieving these modifications.

1 Introduction

Today, all the major operating systems and a growing number of devices include a standards compliant IPsec [16] implementation. This is occurring later than anticipated – perhaps due to the slow standardization process, or the widespread availability of the SSH [26] and TLS [10] protocols [13]. But the current wide support is significant, since it would allow IPsec to be used as a universal security solution. However, IPsec is today typically used only as an "all or nothing" protection for VPNs, connecting, for instance, two sites privately together or remote users to a central site.

Today, IPsec is rarely used for end-to-end protection of application protocols. It is also rarely used for protecting the various control protocols of the Internet. This is unfortunate, as the original goal of IPsec was to enable the protection of all types of communications between Internet nodes. It is even required that all IPv6 nodes implement IPsec. At the same time, there are a number of situations where there are no credible alternatives to the use of IPsec. For example, TLS and SSH are limited to TCP- and SCTP-based protocols. Furthermore, IPsec offers certain security benefits over the alternatives, such as better resistance to Denial-of-Service attacks than TLS.

In the IETF, all new protocol specifications have to describe how the protocols are to be secured. Often there is a mandatory requirement that all implementations must support a particular security solution [23]. However, many of the older specifications suggest using IPsec without providing sufficient information on how exactly to use IPsec. Many of the newer protocols rely on TLS. Some of newer protocols require IPsec, but the details are often relatively complicated in practice [1, 7]. More importantly, there is lack of users that actually turn *on* the specified and implemented security mechanisms, even when they are there and readily available. In some cases we have even found evidence that the originally intended secure behavior fails in unexpected ways and has significant, undocumented limitations.

The purpose of this paper is to share our experiences on the reasons that have led to the rare of use IPsec outside of the VPN applications. One of our conclusions is that quite often the existing IPsec policy mechanisms turn out to be inadequate to the tasks at hand. While IPsec is quite effective in authenticating the peers and establishing assurance about the *identity* of the peers, we claim that the lack of attention to *authorization* questions is a root cause for many of the existing inadequacies. We also think that the problems are more fundamental than the lack of suitable APIs, a common complaint against IPsec [8, 13].

The rest of this paper is structured as follows. In Section 2 we illustrate our experiences in using IPsec in a number of situations, including IPv6 control protocols, mobility protocols, network management, and multimedia protocols. In Section 3 we analyze these results and categorize the observed limitations. Finally, in Section 4 we present some potential architectural modifications which could improve the situation, and discuss the practical challenges in achieving these modifications.

2 Practical Experiences

2.1 IPv6 Neighbor Discovery

The Neighbor Discovery (ND) protocol is an integral part of IPv6. The protocol is defined in RFCs 2461 and 2462 [21, 25], implemented in all IPv6 devices. Neighbor Discovery is used by IPv6 nodes to discover other nodes on the same local link, to determine the link-layer addresses of these nodes, to find routers, and to maintain reachability information about the active neighbors. The specifications call for the use of IPsec to protect the ND messages, though with the details and limitations have been largely left unspecified.

In this particular application IPsec can only be used with manual keying. The currently standardized key management protocols, i.e. IKE, cannot be used, since ND uses multicast, and multicast is not supported by IKE.

More fundamentally, a chicken-and-egg problem [2] appears when attempting to use IKE [12] before ND is operational. The problem is easy to explain through an example. Let us assume that Alice wants to communicate with Bob over the local link. Since all she initially has is Bob's IP address, she must first find Bob's link-layer address. To do so, she must run the Neighbor Discovery protocol. Now, if all traffic between Alice and Bob is expected to be secured with IPsec, this would imply that even the messages used for finding Bob's link-layer address would have to be secured with IPsec. In order to secure these messages, a pair of IPsec security association between Alice and Bob must either exist or needs to be established. Since we are assuming that manual keying is not used, such security associations must be created. This requires IP packets to be sent, such as the IKE UDP packets. However, in order to send such packets, the link-layer address of Bob would have to be known to Alice. Hence, a chicken-and-egg problem exists.

Even if manual keying can be used with Neighbor Discovery, the number of security associations needed is quite large [4]. The number of security associations on *each* node is proportional to the number of IP addresses assigned to *all* nodes attached to the link. This is impractical. Even if keys shared between all members of a group were acceptable, using manual keying becomes impossible on links where the IP addresses – which are decided by the nodes themselves in IPv6 – are not known beforehand.

More importantly, the use symmetric security associations does not prevent authenticated hosts from masquerading as routers for other hosts, given that the same key is known to all nodes when multicast is used. Similarly, a host can forge messages that appear to come from another host's address.

Finally, the lack of detailed instructions in RFC 2461 on how to set up the necessary security associations and policy entries creates a burden for administrators, and may limit interoperability.

As a result, IPsec is rarely, if ever, used for protecting Neighbor Discovery messages.

Similar problems about lack of IP addresses in the initial stages make the use of IPsec hard with DHCP [11]. Multicast problems appear also in other applications that use multicast or broadcast, such as routing protocols.

2.2 Secure IPv6 Neighbor Discovery

The problems in protecting Neighbor Discovery became apparent a few years ago. More recently, new proposals have appeared for solving some of these problems [5, 6, 18]. Even if this is still ongoing work, we discuss some of the experiences gained in these attempts to extend IPsec to make Neighbor Discovery protection easier.

The approach chosen in [6] involves the definition of a new IPsec transform, based on asymmetric cryptography. The new transform is designed to be compatible with multicast, avoid the chicken-and-egg problems, authorize routers and hosts separately, and be capable of telling which addresses belong to which hosts. IPsec AH [15], adopted to use public key cryptography, is used to protect the relevant Neighbor Discovery messages. However, a few problems still remain. A minor problem is that the IPsec policy mechanisms do not allow selection based on ICMP message types. Another issue is that some of the Neighbor Discovery solicitations are sent with an unspecified source address, and the "real" address appears only at the application layer. This prevents IPsec from ensuring that the host has used the right address in the message. These problems can be solved, by extending the IPsec policy mechanisms and by modifying the source address rules of the Neighbor Discovery protocol.

But there are additional problems that cannot be solved easily. The most interesting problem relates to the gradual deployment of secure Neighbor Discovery on links that have both secure and non-secure nodes. While it is possible to arrange this with IPsec, it would come at the cost of using different addresses for the two groups of nodes. For instance, each node would have to have two IPv6 link-local address, one secure and one insecure, and the address prefixes announced by the routers would have to be duplicated in order to make it possible to use both secure and insecure nodes at the same link.

In contrast, a security mechanism integrated to the Neighbor Discovery protocol can be made optional, allowing easy co-existence with older equipment [3]. The secure Neighbor Discovery implementation has immediate knowledge about the security status of any received message and can make specific checks with regards to the acceptance of the message. For instance, a securely communicated router advertisement would override any previously received insecure advertisement, but not vice versa. This level of control is hard to achieve with IPsec.

In addition, given the stateless nature of IPsec (in an environment that does not have a separate key management phase), replay protection can be easily arranged only through timestamps. In an solution integrated with Neighbor Discovery, request - response pairs can be easily matched using a nonce. Since the use of nonces is desirable, an IPsec-based security solution would need components both from the Neighbor Discovery and IPsec layers, thereby complicating the analysis of the mechanisms.

Given these issues, the IETF recently decided to abandon IPsec, and design the future Neighbor Discovery security solution in such a way that it is integrated with the actual protocol.

2.3 IP Mobility

An initial approach to secure Mobile IPv6 route optimization relied on IPsec and its key management. It was suggested to use IPsec to protect the so called Binding Updates, which are messages sent by the mobile nodes to their peer nodes in the Internet. While thas approach would work well on a small scale – for instance, between a user's laptop and her home server – it fails on the global scale.

An obvious reason is that we do not have (and most probably will never have) a global authentication infrastructure which could help the two hosts to set up security associations between them. More interestingly, even if such an infrastructure would exist, identity based authentication of Binding Updates would be almost worthless. Simple identity based authentication would verify that a claimed identity, e.g., an e-mail address or a Distinguished Name, corresponds to the used public keys. Binding Updates, however, operate on IP addresses. Thus, unless the used authentication infrastructure were to track all the addresses assigned to users (even with dynamic assignments) and to provide this information in a secure way, the authentication information would not help in authorizing operations on IP addresses.

As a result, it was necessary to develop new security mechanisms for Mobile IPv6 Route Optimization. The new mechanisms rely on the routing infrastructure to provide some level of the authorization of IP addresses [14].

Interestingly, IPsec is still used in Mobile IPv6 to protect the signaling between the mobile nodes and their home agents. In a way, protecting these message is an easier problem, since there must be an a priori relationship between a mobile node and its home agent. However, the current specifications create a hard binding between static IP addresses, security associations, and IKE endpoints [7]. This creates some undesirable complexity and limitations to the protocol. For instance, current protocols do not easily allow dynamic allocation of home addresses, as security policies are tied to the addresses.

2.4 Network Management Protocols

Interactions and conflicts between the application layer and the IPsec layer are also problematic for using IPsec as a form of protection for network management protocols. IPsec can easily provide a base-level security for all management traffic in a large network. However, typically there is a requirement to differentiate management users from each other. For instance, one user may be allowed to only view data (e.g. SNMP GET) while another would also be allowed to make modifications. A third user might only be allowed to change particular variables.

It is hard to model this in the IPsec security policy database, as the policies are not expressive enough to be able to understand which operations are being performed at the application layer. At the same time, the lack of standardized or deployable APIs between the IPsec and application layer prevents the application from learning the identity of the user from IPsec. As a result, either the offered security level resembles "all nodes are equal" VPNs, or other security solutions must be adopted.

If the IPsec implementation in a management station supports only machine level authentication and not user authentication, it can be difficult to distinguish two users with different privileges when that station communicates with a managed node. Similarly, problems arise if the implementation supports user authentication but does not ensure that only the traffic from this particular user is protected by security associations created for that user – achieving this in a kernel mode implementation may be hard. For these reasons, some protocols provide machine authentication at the IPsec level, but add their own user authentication mechanisms inside the application layer. Examples of such protocols include iSCSI [1] and L2TP [19].

2.5 Streaming Multimedia

Currently the selection of outgoing IPsec policy entries is based on the examination of IP addresses, the upper layer protocol identifier, and port numbers. This approach works well with protocols that use stable IP addresses and meaningful port numbers. Not all applications are like that. For instance, the streaming multimedia applications utilizing UDP and RTP [24] are quite problematic from this perspective. Both the source and destination port numbers used for RTP can be dynamically assigned, making it hard to define IPsec policies to select the appropriate UDP streams to protect.

3 Analysis of the Limitations

The limitations we have experienced fall under the following categories:

- Lack of expressive power in policy specifications.
- Lack of application control over policies.
- Lack of support for authorization, and the inability to link authorization decisions with security processing.

In the following, we discuss each of these issues separately.

3.1 Expressive Power

In some cases the expressive power of the security policy entries needs to be increased to cover more than the pair of addresses, the upper layer protocol identifier, and the ports. For instance, recent new requirements include SCTP addresses and port numbers, or ICMP message types.

More generally, IP addresses no longer carry semantic information to the extent they used to do. Given the widespread use of dynamic address assignment, nomadic users, and the growing use of mobility, privacy addressing [20], and multihoming, an IP address does no more identify the host. Yet, the IPsec policies and security association parameters are tightly bound with IP addresses. This limits the freedom nodes have in choosing the addresses they use for communications.

Furthermore, there is no support in the current (or planned) IPsec key management protocols for controlled address agility. For instance, it would be useful to change the address associated with a particular IKE phase 1 security association, or be able to use multiple addresses instead of just one.

Finally, decisions cannot always be made based on port numbers, given that not all protocols use static port numbers. Furthermore, port numbers are not universally available in security gateways when fragmentation occurs.

As a result of the above, there are scenarios where *none* of the available selectors can be used, for instance when both dynamic addresses and dynamic port numbers are used.

3.2 Application Control

Application control of policies is required in two cases: First, it may be necessary to use of IPsec in cases where there are no distinguishable selectors. Consider, for instance, the dynamic address and port number case discussed above.

Secondly, policies need to be configured by applications. As we have found out, significant protocols can become widely deployed without security ever being turned on. A partial reason for this is that users find it unreasonable that they have to configure IPsec policy entries, and that not all networks have professional network managers whose task is to ensure correct policies.

On the other hand, the end-host applications could set up the necessary entries since in many cases the expected policies are standardized. Some standard specifications even require that the application is aware of the underlying security mechanisms, or at least whether security is turned on or off.

3.3 Authorization

Not all peers have the same rights. For instance, the administrator of a host, connecting from a remote device, has different rights than an unknown peer connecting through opportunistic IPsec. Even peers whose identities are authenticated within a specific trust domain may still have different rights. For example, in a corporate network, network managers have different rights than employees.

In order to use certificates for authorization, it is typically necessary to either use attribute certificates, create a separate certificate infrastructure for each different application, or introduce standardized extensions to the certificate formats to be able to express the authorization that the certificate represents. The traditional alternative, using local access control lists is not a viable option in a large network.

Finally, there are no standard requirements for IKE to take in account the authorization information from certificates (phase 1) in accepting a specific request for new security associations (phase 2). And, taking this account can be problematic for the responder as it may need to use a specific identity in phase 1, but does not know for what purpose the security associations will be requested in phase 2.

More importantly, the security processing decisions often require information from the application layer, or vice versa. In the current IPsec architecture all security processing is expected to be performed at the IPsec layer, and the application typically does not even know if IPsec was applied. This makes it hard to accommodate finer grained decisions, for example, such as those required in co-existence of secure and non-secure Neighbor Discovery.

4 Potential Improvements

There appears to be two possible paths for the future development of security solutions in the context of protocols discussed in this paper. The first path involves focusing IPsec as a protocol solely for VPNs, and using other solutions for securing application protocols. The second path involves an attempt to make IPsec friendlier for application protocols.

In the following we discuss the required improvements for both of these approaches.

4.1 Application-Specific Security

This appears to be the current default assumption for most protocol developers. An advantage of this approach is that no additional requirements are placed for current IPsec implementations, and that tailored security solutions can be provided for the application protocols.

A disadvantage is that a tailored security solution is always needed, at least for those protocols that cannot employ TLS or S/MIME. This approach seems also at odds with the current IPv6 requirement of mandating IPsec for every node. (For instance, a requirement to support TLS for applications and secure Neighbor Discovery for control signaling might provide a better security / cost ratio.)

If this approach is chosen, it would be desirable to have a common security solution for those protocols that cannot employ TLS or S/MIME. It may not be possible to create a separate protocol or a separate "layer" due to the very different nature of the applications. However, it may be possible to provide tools in the form of common security object formats and a library to support them. The library should allow application programmers to integrate the following features to their protocols:

- Liveness, denial-of-service, and address validity testing procedures.
- Retrieval and processing of certificates and certificate chains.
- Passing of signed and encrypted data objects.
- Passing information about the peer to the application, and responding to authorization questions from the application.
- Cryptographically generated address (CGA) processing [22].

While some such libraries exist today, they are typically tailored for traditional applications, and lack mechanisms (such as address tests) required in control protocols.

4.2 Enhanced IPsec

The following improvements appear necessary for the wider use of IPsec in securing applications:

1. Provide a mechanism for applications to control policies. We recommend that specific applications automatically provide at least a default configuration for IPsec. This can be provided through an API. That would help ensure that security is not accidentally turned off, improve interoperability, and it might help in dealing with the port number agility problem. The drawback of the approach is an additional burden on the applications. However, there is a trade-off between a one-time effort in software construction and manual configuration effort by all users, making the application based solution beneficial in the longer run.
   Unfortunately, such default configurations may not always suffice. For instance, the default configuration may not be compatible with specific requirements for protecting all traffic from the node in question.

2. Allow applications to make authorization decisions. One approach to allow applications to control authorization is to create an application programming interface (API) between IPsec, IKE, and applications. The creation of such an interface is particularly easy when the "applications" are integral parts of the IP stack, such as in IPv6 control signaling. A standardized API would also make it possible for user space applications to rely on IPsec and IKE to provide meaningful security information to them.
   However, it is not enough to copy the functionality of the IPsec security policy database to the application layer; the application layer policy information has to be presented in "semantic" form. For instance, policies could be based on application concepts rather than port numbers.
   In order to make use of the API, applications need tools to deal with authorization questions. For instance, an application may have a need to perform certificate chain validation to a particular trusted root. Such needs can be satisfied either through the API or via a specific library, such as Keynote2 [9].
   The effective use of certificates for authorization purposes requires certificate mechanisms extensions so that it is possible to represent authorization information in them in an easy manner.
3. Reduce the reliance on IP addresses in the policies. This reduction applies in IPsec security associations, in policy entries, and in application layer policies. One approach to go to that direction is presently available in the form of the Host Identity Protocol (HIP) [17] proposal. Reducing reliance on IP addresses is necessary to allow for address agility.
   While the enhancement of the expressive power of IPsec policies appears necessary, it seems hard to decide exactly how much more expressive the policies should be, and from which point applications and interfaces to applications should be used instead. Currently, our opinion is that the policies should be extended to cover new transport protocols, ICMP message types, and address agility.

4.3 Conclusions

We have described our experiences in attempting to use IPsec to secure several types of protocols. Based on these experiences, we recommend either dedicating IPsec for pure VPN usage, or extending IPsec with APIs that can provide the required level of information and control for the applications.

As usual, the biggest challenge of the IPsec enhancements seems to be deployment. Implementing the suggested changes does not seem to have any larger technical or theoretical obstacles, though they can only be provided in native host implementations and not with Bumb-in-the-Wire implementations. While distributing the new implementations will be time consuming, the main problems are in the applications.

Handling authorization properly seems to be especially challenging. It requires attention both from security and application semantics. Integrating IPsec with generic authorization mechanisms, such as Keynote2 [9], and providing an API for applications so that they can query IPsec for both identification and authorization information, is only the beginning. The applications must also be changed so that they actually use this information in some meaningful way.

Our experience suggests that there is an apparent dichotomy between the security experts and application developers, at least at the IETF. Creating proper communication between these two groups of people seems to be really hard.

Most application developers have been trained to think about security in terms of users and user rights, i.e., access control lists. However, many future applications would probably benefit from different authorization models, such as relying on direct certificate based authorization instead of relying on locally configured username based access control lists. Application developers often also tend to think of security as an authentication and encryption mechanism, and fail to use their application expertise to develop requirements for the security solution. Thus, application developers need new mental models, and that will take time.

Many of the security experts are unwilling to consider any compromises to security, and may not perform a proper risk analysis. This, in turn, has led some applications developers to shun security.

Willingness to think in new ways is required both from application developers and from security experts. The application developers (and even some security experts) need to learn to think in new terms, paying much more attention to authorization than before. The security experts must accept that often "good enough" security is better than as-strong-as-possible security. For example, the key management procedures required by a full strength security solution might not be possible due to the planned deployment scenarios of the application. Both people need to pay much more attention to economics, to the interests of the key players, and to deployment scenarios.

Acknowledgments

The authors would like to thank Bernard Aboba, James Kempf, Steven Bellovin, and Stephen Kent for interesting discussions in this problem space.

References

1. B. Aboba, J. Tseng, J. Walker, V. Rangan, and F. Travostino. Securing Block Storage Protocols over IP. Internet Draft draft-ietf-ips-security-19.txt (Work In Progress), IETF, January 2003.
2. J. Arkko. Effects of ICMPv6 on IKE and IPsec Policies. Internet Draft draft-arkko-icmpv6-ike-effects-01.txt (Work In Progress), IETF, June 2002.
3. J. Arkko SEcure Neighbor Discovery (SEND). Internet Draft draft-arkko-send-ndopt-00.txt (Work In Progress), IETF, June 2003.
4. J. Arkko, P. Nikander, T. Kivinen, and M. Rossi. Manual SA Configuration for IPv6 Link Local Messages. Internet Draft draft-arkko-manual-icmpv6-sas-01 (Work In Progress), IETF, June 2002.
5. J. Arkko, T. Aura, J. Kempf, V.-M. Mantyla, P. Nikander, and M. Roe. Securing IPv6 Neighbor Discovery. In *Wireless Security Workshop*, September 2002.

6. J. Arkko, J. Kempf, B. Sommerfeld, and B. Zill. SEcure Neighbor Discovery (SEND). Internet Draft draft-ietf-send-ipsec-01.txt (Work In Progress), IETF, June 2003.
7. J. Arkko, V. Devarapalli, and F. Dupont. Using IPsec to Protect Mobile IPv6 Signaling between Mobile Nodes and Home Agents. Internet Draft draft-ietf-mobileip-mipv6-ha-ipsec-06.txt (Work In Progress), IETF, June 2003.
8. S. Bellovin. Guidelines for mandating the use of IPsec. Internet Draft draft-bellovin-useipsec-00.txt (Work In Progress), IETF, October 2002.
9. M. Blaze et al. The KeyNote Trust-Management System Version 2. RFC 2704, IETF, September 1999.
10. T. Dierks and C. Allen. The TLS Protocol Version 1.0. RFC 2246, IETF, January 1999.
11. R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, M. Carney. Dynamic Host Configuration Protocol for IPv6 (DHCPv6). Internet Draft draft-ietf-dhc-dhcpv6-28.txt (Work In Progress), IETF, November 2002.
12. D. Harkins and D. Carrel. The Internet Key Exchange (IKE). RFC 2409, IETF, November 1998.
13. J. Ionnadis. Why don't we still have IPsec, dammit. NDSS 2003, February 2003.
14. D. Johnson, C. Perkins, and J. Arkko. Mobility Support in IPv6. Internet Draft draft-ietf-mobileip-ipv6-24.txt (Work In Progress), IETF, June 2003.
15. S. Kent and R. Atkinson. IP Authentication Header. RFC 2402, IETF, November 1998.
16. S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. RFC 2401, IETF, November 1998.
17. R. Moskowitz, P. Nikander, and P. Jokela. Host Identity Payload and Protocol. Internet Draft draft-moskowitz-hip-07.txt (Work In Progress), IETF, June 2003.
18. P. Nikander. Secure Neighbor Discovery using separate CGA extension header. Internet Draft (Work In Progress), IETF, June 2003.
19. B. Patel, B. Aboba, W. Dixon, G. Zorn, S. Booth. Securing L2TP using IPsec. RFC 3193, IETF, November 2001.
20. T. Narten and R. Draves. Privacy Extensions for Stateless Address Autoconfiguration in IPv6. RFC 3041, IETF, January 2001.
21. T. Narten, E. Nordmark, and W. Simpson. Neighbor Discovery for IP Version 6 (IPv6). RFC 2461, IETF, December 1998.
22. Greg O'Shea and Michael Roe. Child-proof authentication for MIPv6 (CAM). *Computer Communications Review*, April 2001.
23. E. Rescorla. Guidelines for Writing RFC Text on Security Considerations. Internet Draft draft-iab-sec-cons-03.txt (Work In Progress), IETF, January 2003.
24. H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson. RTP: A Transport Protocol for Real-time Applications. RFC 1889, IETF, January 1996.
25. S. Thomson and T. Narten. IPv6 Stateless Address Autoconfiguration. RFC 2462, IETF, December 1998.
26. T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, and S. Lehtinen. SSH protocol architecture. Internet Draft draft-ietf-secsh-architecture-12.txt (Work In Progress), IETF, January 2002.