Paper

Automatic multicast IPsec by using a proactive IPsec discovery protocol and a group key management

Thorsten Aurisch, Tobias Ginzler, Peter Martini, Roger Ogden, Trung Tran, and Hartmut Seifert

Abstract— Internet protocol based networking is gaining ground in armed forces, leading to a concept described by the NATO as network centric capabilities (NCC). The goal is to enable state-of-the-art, affordable and powerful electronic information services to the troops. A tighter connection of the forces is expected to further enhance the joined strike capabilities. Providing secure information exchange within groups of armed forces is one aspect of the NCC concept. Such group communication is enabled by the multicast feature of the IP technology. Security requirements are met by using the IP security (IPsec) architecture. IPsec enables secure communication between secure private networks via an unsecured public text network. While secure unicast transmission with IPsec is common, only few achievements have been made to secure multicast transmissions. The protection of multicast data traffic of a group in an automated way is described in this document. We utilize an automatic detection of IPsec devices and an efficient key management protocol to reach our aim.

Keywords— secure group communication group key management, multicast IPsec, automatic IPSec device discovery.

1. Introduction

Establishment of keys and protocol parameters is necessary to enable IP security (IPsec) for securing network traffic. Such a set of keys and parameters is called security association (SA) in IPsec terminology. Manual configuration of the IPsec capable nodes is a time consuming task. Moreover deploying pre-shared keys for network encryption renders all IPsec nodes insecure if a single node is compromised.

To overcome these caveats in point-to-point communication the Internet key exchange protocol (IKE) has been created. It provides an automated configuration of IPsec devices. IKE enforces a set of security policies (SPs) such as minimum key length or maximum key life-time and configures security associations at the IPsec devices.

For multicast IPsec on the other hand, no automated way for configuration exists yet. It is required to have both automated key and IPsec parameter establishment for group communication, too. In many situations group communication needs to be secure, but trained computer specialists might not be available for manual configuration. Our two-parted approach addresses both requirements: easy to maintain and secure. One component of our concept is the IPsec discovery protocol (IDP) [10]. It discovers IPsec capable devices and configures them in an automated way. The second component is the multicast Internet key exchange protocol (MIKE) [5, 9, 18]. It negotiates and establishes a group key in a secure manner. By combining the two, an automatically secured network is possible. Only a one-time configuration of the affected network nodes is necessary in our approach. After that, the system self-maintains its security properties, even if nodes join or leave the group.

This paper is organized as follows. Section 2 gives account of previous work done in the realm of key management and IPsec discovery protocols. Our target use case is described in Sections 3 and 4. Section 5 goes into the details of the protocol communication. Our performance analysis deployment is described in Section 6. Finally, Section 7 summarizes and gives a further outlook.

2. Previous work

In this section, we want to give an overview of existing key management and IPsec discovery solutions. We want to motivate, why MIKE and the IDP were chosen to automate network encryption.

The IPsec discovery protocol clients run at the borders of secure (“red”) networks and un-secure public (“black”) networks. Their task is to discover red enclaves and interconnect them over the black network. A global discovery protocol for IPsec devices is described in [1]. It describes an infrastructure with a worldwide hierarchy of servers in a domain name system (DNS)-like manner (client-server discovery – CSD). Although different balancing schemes are discussed, the architecture relies on a single root server, which is considered to be a single point of failure.

Another possibility to locate IPsec routers is to reserve special IP addresses in each network prefix for them. It eliminates the need for a discovery protocol at all. This implicit peer enclave discovery protocol (IMPEPD) called solution scales well, but offers little flexibility. Another mechanism is called multicast discovery (MD), reaching neighbor routers via a multicast router request. Because of its reactive nature it may impose latency especially in dynamic environments. All of these proposals are expected to be included or are already part of high assurance Internet protocol encryptor interoperability specification (HAIPE IS) Version 3, the U.S. approved version of IPsec.

An approach which overcomes most of the limitations of the existing IPsec discovery mechanisms is the IPsec discovery protocol. IDP relies on IP multicasting within the black network. Because of its proactive nature and because it needs no special servers, it is best suited for medium scale networks. Larger scale networks can be enabled by partitioning them and introducing IDP security gateways. IDP is able to configure IPsec security associations and enforces a common IPsec security policy within the domain. Certificates are distributed by IDP, too.

Table 1 summarizes the comparison of the different discovery mechanisms. The capability to handle a large number of networks, robustness against the loss of parts of the infrastructure, flexibility and imposed latency is taken into account.

Table 1
Comparison of discovery architectures

Discovery mechanism	Scalability	Robustness	Flexibility	Latency
IMPEPD	Good	Good	Worst	Best
MD	Average	Good	Good	Worst
CSD	Best	Average	Average	Good
IDP	Good	Good	Good	Good

A key management service is necessary to enable multicast IPsec. The key management establishes a common group key at all IPsec devices. The key establishment takes place over the black network, so it has to be done in a secure manner. Public key cryptography and certificates are often used to accomplish this task. Once the common key is established, a confidential and authenticated communication through the black network is possible. Additionally, a key management system should be able to support users to join and leave the group without sacrificing security.

Different protocols and key management architectures exist. An overview of existing approaches is given in Fig. 1. For security reasons adding and excluding members of a group enforces a key change. Algorithms which support the frequent change of group keys are called "dynamic" key management approaches. The pre-distributed group keys (PGK) method is the only listed system which uses a static key and does not support frequent changes. For this reason it is only practicable for small, static groups.

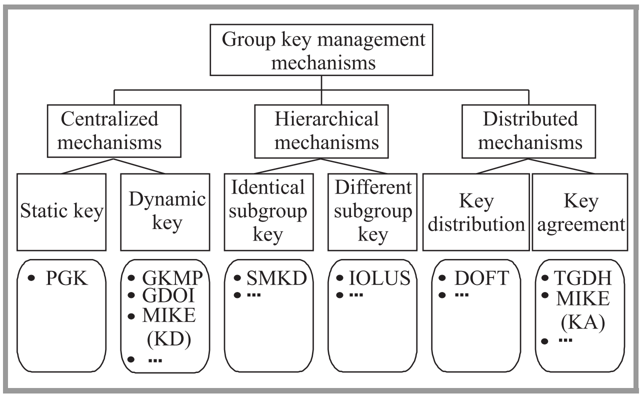


Fig. 1. Overview of group key management mechanisms.

Most of the algorithms rely on a centralized or hierarchical structure. Centralized systems like group key management protocol (GKMP) [11] or group domain of interpretation (GDOI) [12] contain a server. On the one hand a server is considered as a single point of failure. On the other hand, building a hierarchical system often involves investing in a costly infrastructure (scalable multicast key distribution – SMKD [16]) or doing expensive de- and reencryption (IOLUS [15]). At first sight, distributed key management mechanisms seem to offer both efficiency and robustness. But most of the distributed systems depend on a reliable or ordered delivery of messages (e.g., tree-based group Diffie-Hellman (TGDH) [13]). Such a service is not offered by IP multicast. Some algorithms only utilize distributed systems to spread the key, while the calculation is done by a single instance (distributed one-way function tree (DOFT) [14]). An attacker would concentrate on this point.

To overcome the limitations of the existing key management mechanisms, the group key management system MIKE in the key agreement (KA) mode has been developed. Even though MIKE offers a distributed key management, it does not depend on ordered or reliable message transport. MIKE operates on top of the standard UDP/IP layer. A transaction manager (TM) spreads topology and status information to the clients. After that, every authenticated member is able to calculate the group key itself. Any device can hold the transaction manager status. The MIKE protocol defines which IPsec device is the current transaction manager. Mechanisms exist to elect a new TM if a TM becomes unreachable. Scalability and performance have been evaluated. Utilizing the key agreement operation mode of MIKE, approximately 50 IPsec devices can be managed easily. Larger networks are supported in the centralized operation mode called key distribution (KD). Key agreement mode offers robustness against failure of IPsec devices and network errors, while maintaining fair scalability.

Based upon a medium scale scenario described in the next section, MIKE and IDP seem to be the best solution for automated network encryption. IDP detects IPsec capable devices, while group key management is done by MIKE. We want to analyze how the combination of these two components is going to perform.

3. Scenario

An international research project sketched a typical network, as it may be deployed within a coalitional operation [2]. The following requirements and conditions have been identified:

- nations want to protect their red national networks;
- the national local area networks (LANs) are interconnected by public wide area networks (WANs);
- nations want to exchange information seamlessly at the tactical level;
- strict WAN/LAN separation;
- no security guarantees are given by the WAN operator.

The sketched network is shown in Fig. 2. IPv6 is used to benefit from the advanced security and multicast capabilities. The IPsec labeled components are border routers with IPsec capabilities. We want to test if the IDPMIKE solution deployed on these devices will enable seamless interconnection of the national LANs.

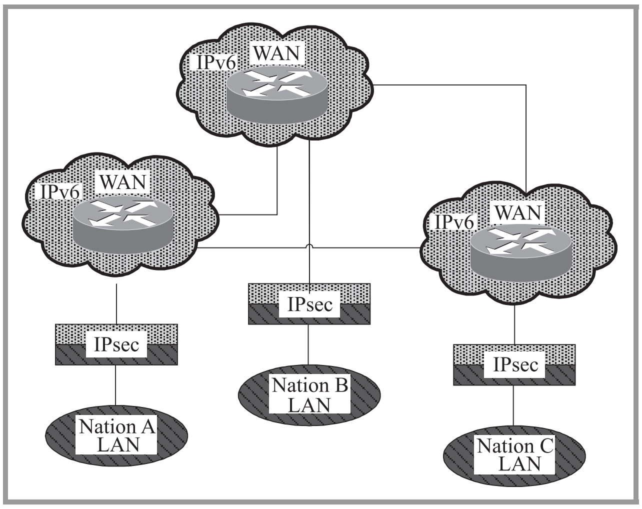


Fig. 2. Network setup of a coalitional operation.

In a typical coalition network, the number of nations (and national LANs) does not exceed a few dozens. So it is not necessary to use IDP security gateways. Furthermore MIKE can be used in key agreement mode.

Certificate distribution is possible with IDP, but it has been deliberately kept out of the scope of this paper, since our focus is on feasibility, scalability, and performance of the network configuration. We consider certificates to be already deployed at the IPsec devices. The IDPMIKE solution is prepared for various kinds of public key infrastructure (PKI) concepts and policy guidelines.

4. Test-bed

According to the sketched network in the section before, a test-bed was built. The crucial components are the border IPsec devices, drawn as two-parted rectangles in Fig. 2. These devices are COTS hardware products, namely PCs running a Fedora Core Linux kernel. IDP Version 3.1.4 runs in combination with MIKE Version 0.8 on these IPsec devices. The realized test-bed is shown in Fig. 3.

Overall ten IPsec devices were involved. For simplification, a switch was used to interconnect the black routers. In a real deployment this is done by a public WAN. All connections were realized as 100 Mbit/s Ethernet. The upper three routers are border routers of the black network. Open shortest path first (OSPF) or the protocol independent multicast (PIM) [8] multicast routing protocol run between these routers. The IPsec devices are drawn as triangles; they contain the IDPMIKE solution. The IPsec devices divide the network into the red network domain and the black network domain. The lower routers are in the domain of the national LAN, providing secure network access to clients.

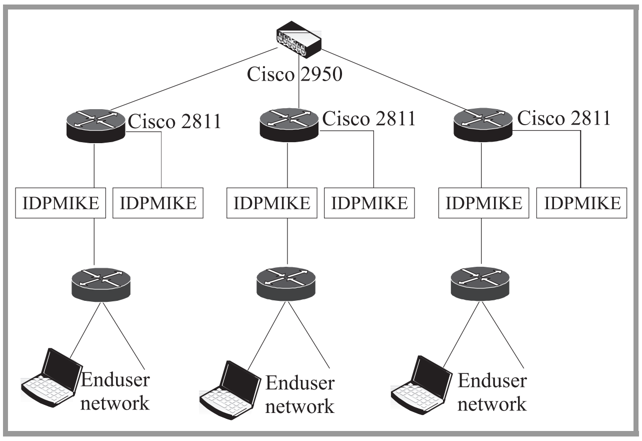


Fig. 3. The realized test-bed.

The realized test net provides not the functionality of the ideal network sketched before. Linux kernel does not support IPv6 multicast routing through IPsec yet. It is not possible to route red IPv6 multicast traffic through IPsec and across the black network to another red network. The setting of the IPsec SA and SP according to the scenario was correct. In independent experiments multicast routing was proved to work correctly as well as IPsec but not both in combination. The reason is that the Linux IPv6 multicast routing procedure bypasses the netfilter API and IPsec stack [3].

5. IDPMIKE interoperation

The interaction process between IDP and MIKE is divided into three phases (Fig. 4). First the discovery process detects a change in its internal connectivity tables. For example an IPsec device joins or leaves the network. The MIKE service is informed about the change by an UDP datagram (Fig. 4, Step 1). In the second phase MIKE establishes a new group key and returns it via a datagram answer to IDP (Fig. 4, Step 2). In the last phase IDP configures the IPsec devices automatically (Fig. 4, Step 3). The phases are now described in detail.

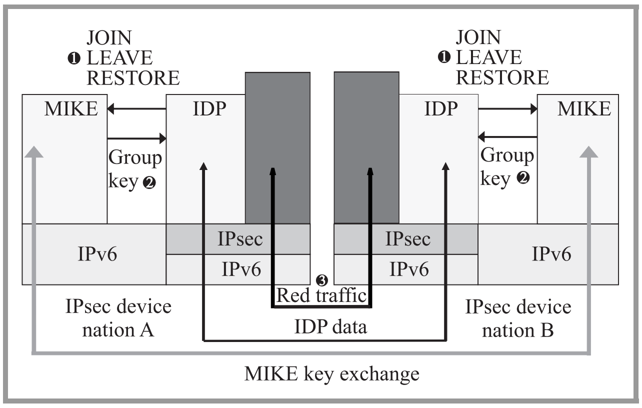


Fig. 4. Interacting sequence of IDP and MIKE.

When an IDP instance starts up, it multicasts a *hello message* periodically. Simultaneously it listens to any incoming IDP message. All IDP instances receiving a *hello message* add the sending IPsec device to their connectivity table if it is not already added. Then they establish a unicast IPsec SA to this device. This connection is secured by a pre-shared key. The SA is used by the IPsec devices to announce its IP network prefix to the newly discovered instance periodically. The prefixes are evaluated by the receivers and the routing is updated accordingly. When no change in the network is detected within four times of the *hello interval*, the announcing of prefixes stops and only *hello messages* are continued to be sent. If an IPsec device stops sending *hello messages* it is considered down and all other IPsec devices delete the according IPsec SAs, SPs and routing entries. The pre-shared key of the SA is used for advertisement of the prefixes only. In an improved version of the integration this pre-shared key can be replaced by the dynamic MIKE key.

After IDP detects a change in the network, MIKE is informed about it. There are three IDP-to-MIKE messages defined. One indicates a JOIN of an IPsec device. IDP reports a newly discovered instance on every IPsec device, so the joining device reports itself, too. The second indicates a LEAVE event and the third event informs MIKE about a RESTORE of an IPsec device. A RESTORE occurs if an IPsec device is reconnected after its network connection temporary failed. A RESTORE is distinguished from a restart by the *hello messages* originating from an already known device. All *hello messages* carry a timestamp of the startup time of the IDP instance. If the timestamp has changed, a restart must have occurred, otherwise a RESTORE event occurred. Restarted IPsec devices are treated as joining instances.

According to the indication reported by IDP, MIKE executes one of the following steps. If a JOIN event was triggered, MIKE checks the IP address of the joined member. If the IP address of the joining instance matches the local IP address, the JOIN is executed and the IPsec device becomes a member of the secure MIKE group. If the addresses differ, no action is performed. It is necessary to check the addresses because only the newly discovered instance can add itself to the MIKE.

The MIKE reacts to a RESTORE event similar as to a JOIN event. Even if the instance is already in the MIKE group, the join is executed (rejoin). This is necessary because the group key may have changed while the IPsec device was temporarily disconnected. A forced rejoin renews the key and assures the proper distribution to all IPsec devices.

While JOIN and RESTORE event are executed by the newly (re)discovered instance, the LEAVE event is only processed by the current MIKE transaction manager. If the TM is supposed to leave the group itself, it hands over the TM status to another IPsec device before leaving. After the handover, the new TM excludes the old one. To put it all together: a LEAVE detected by IDP triggers the exclusion of that member out of the MIKE group, thus rendering the key of the excluded member useless.

After MIKE has finished its operation it informs the IDP service about it. A short status report is sent, containing a pointer to the new group key. In a final step, IDP writes a multicast IPsec SA with the group key MIKE reported. Establishing the unicast SA and SPs and the multicast SA is done in parallel, as soon as a change is detected. The multicast policy is left unchanged, because it has already set at startup time.

During a bilateral workshop the sketched scenario and interoperation functionality was proved within the test-bed [4]. Mainly two test cases were examined:

1) successive addition of new members,

2) successive leave of members.

After each addition or leave the security associations were checked at the IPsec devices. Joining was performed by starting up IDP. A leave was simulated by shutting down an IDP instance. It was possible to produce the same results by physically disconnecting IPsec devices and reconnect them. It shows the robustness and applicability of the protocol combination in tactical networks.

The IDP waits four times of the *hello interval* before considering an instance to be down. This is the reason why it handles flipping connections well. If a connection changes between the up and down state periodically, this connection is considered valid. If the connection stays down for longer than four times of the *hello interval*, it is considered down. This prevents unnecessary key updates due to lost *hello* datagrams.

6. Measurement setup

To analyze the scalability of our solution, measurements were conducted. It had to be determined how long it takes from connecting a network to finishing the setup of the new group key. The measured time span reached from the detection of other members of the intercoalition LAN to the setup of the IPsec SAs at the new device is finished (Fig. 5).

In a real deployment the IPsec devices are interconnected through black routers. After an IPsec device is physically connected to a router, it may take some time until all other black routers have established routes to the newly connected network. This delay is not taken into consideration, as is relies heavily on the used routing protocol and its timeout parameters.

It is also noted, that communication between different IPsec devices is only possible, if the SAs are set up at both IPsec devices. The difference between the set-up times at the different IPsec devices is called dispersal. The dispersal depends mainly on network delay and computational speed of the IPsec devices. It was shown, that dispersal in a lo-

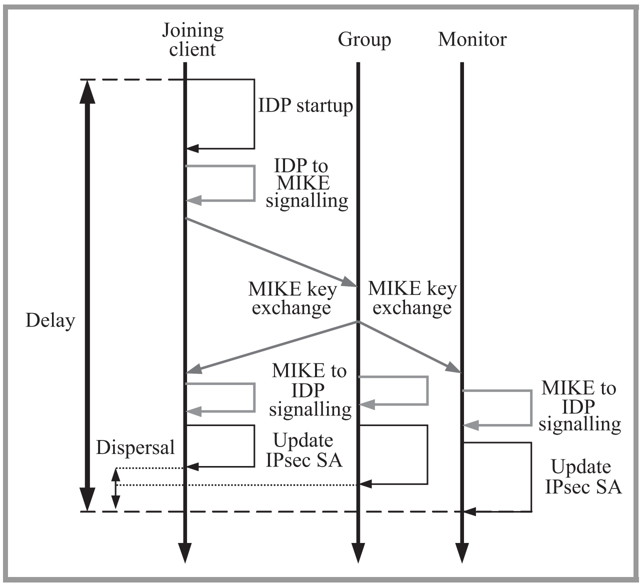


Fig. 5. Time sequence of a measurement step.

cal 100 Mbit/s Ethernet network with homogenous IPsec device hardware is minimal for MIKE [5]. Since the additional delay imposed by IDP is constant, dispersal in the IDPMIKE scenario may be also constant.

The test-bed shown in Fig. 3 was adopted for performance measurements. All devices were interconnected by a switch, no routers were involved. This simplification was necessary to minimize routing protocol side effects and dispersal. The number of clients was increased to ten IPsec devices to better show scaling effects. The devices were interconnected to each other, but IDPMIKE was not started at this point in time. It simulated, that the devices were not able to communicate. In a preparation step, two IDPMIKE IPsec devices were started up. The devices discovered each other and built up their IPsec SAs. One of them was determined as monitor. The monitor measured for each IDP operation how long it takes from detecting a member to IDP writing the multicast SA. The second IPsec device was responsible for holding the MIKE transaction manager status before the actual measurement started. It is important to have the same device being TM each time a measurement is conducted, to ensure identical conditions for each measurement step.

Writing the unicast SAs and multicast SAs is done simultaneously, when a change in the network is detected. For our performance evaluation we assured the multicast SA to be written first. This is done because the focus of our work is in the establishment of multicast IPsec SAs.

After the preparation has finished, IDPMIKE starts up on the remaining IPsec devices. The start up is time triggered. All time triggered events were pre-calculated before the measurements. The monitor logs the time after the SA was written. After that, the second IP-sec device takes over the TM status. The group size is increased by starting up another IDPMIKE device. This is done until all IPsec devices have started up. The difference between the startup time and the time the SA is plotted in Fig. 6. Ten repetitions are done resulting in 80 values.

All IPsec devices are time synchronized using the network time protocol (NTP) [17]. Update interval and an adopted kernel are utilized to keep time synchronization error below 1 ms [6].

7. Results

The delay imposed by IDP and MIKE before a secure connection is established is about 400 ms. In Fig. 6 the measured delays are drawn. As a second result it is observed that the delay hardly increases when the group becomes larger. Indeed the delay increases with a small linear factor with the group size due to the MIKE key establishment algorithm. But the growth is too little to be visible is the diagram. The performance of MIKE was analyzed in [5] in detail. The measure data is reprinted in Fig. 7. As for a group size of ten, the delay imposed by MIKE is about 50 ms and will not exceed 80 ms at a group size of 50 members.

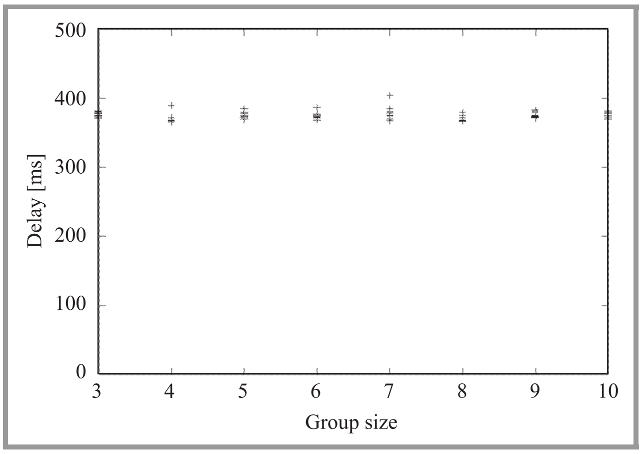


Fig. 6. Latency of IDPMIKE.

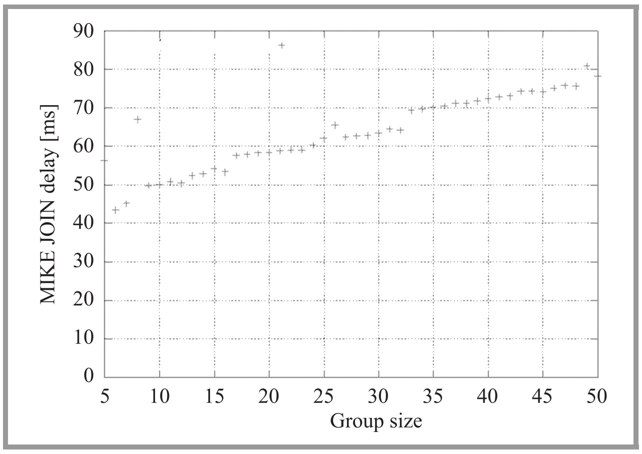


Fig. 7. Delay of MIKE in larger groups.

To further investigate the composition of the delay, two measurement points were analyzed in more detail (Fig. 8). The communication overhead for signaling changes to MIKE and the way back is very small. So the interprocess communication is not a bottleneck.

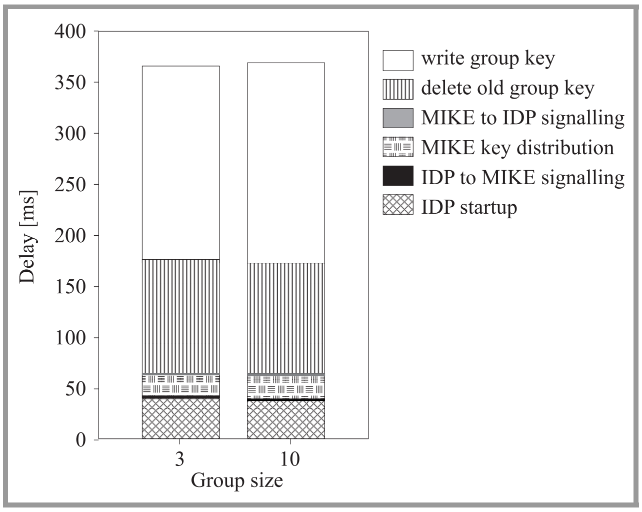


Fig. 8. Delay splitted up.

Surprisingly the vast majority of time is consumed by deleting the old IPsec multicast SA and writing the new one. If the duration for writing the unicast SA is taken into account, the delay is about 200 ms longer. The Linux `setkey` command can be used to manually create IPsec SAs. The time it takes for writing an IPsec SA can be determined by calling the `time` command in conjunction with `setkey` as shown in Fig. 9. The manual creation of an IPsec SA took about 200 ms. The result was verified on different machines. The execution duration is dependant on the CPU power of the machine.

```
> time setkey -f addMulTun_0

real    0m0.201s
user    0m0.004s
sys     0m0.000s

> cat addMulTun_0

add  2001::1  ff0e::1  esp  0x1  -m  tunnel  -E
3des-cbc "012345678901234567890123" -A hmac-
sha1 "this_is_the_test_key";
```

Fig. 9. Performance of `setkey` on a 2.53 GHz P4.

In the available, preliminary version of IDP `setkey` is called via the `system` directive. The usage of the PF_KEY API [7] instead of `system` may lead to smaller delays.

8. Summary

We showed automated network encryption to be manageable by combining the discovery algorithm of IDP and the key management of MIKE. In the test deployment, a full intercoalition network with all involved components such as protocol independent multicast routers and IPsec devices was proven to be working. Our approach enables dynamic and automated multicast traffic protection. This minimizes setup times for intercoalition networks. In a later enhancement of the protocol, the pre-shared key used for IDP prefix advertisement traffic can be replaced by the common group key supplied by MIKE. We were able to show that adding a multicast network to an existing coalition network can be performed within 400 ms with strong security guarantees. The obtained measurement results indicate that IDP in combination with MIKE scales well with increasing network size. The delay is expected to be smaller, once a different interface to set IPsec security associations is used.

References

[1] G. Nakamoto, L. Higgins, and J. Richter, "Scalable HAIPE discovery using a DNS-like referral mode", MITRE Corporation, Aug. 2005.

[2] H. Seifert *et al.*, "Interoperable networks for secure communications II (INSC II) Task 2", Final Rep., Aug. 2006, p. 6f.

[3] A. Faul, C. Zänker, and M. Zeller, "PMIDP-implementation in LINUX", IABG, March 2007.

[4] A. Faul, T. Ginzler, and M. Zeller, "IDP and MIKE interoperation on LINUX", IABG and FGAN, July 2007.

[5] T. Ginzler, "Bewertung und Implementierung von Schlüsselmanagentsystemen in Rechnernetzen", Diploma thesis, University of Bonn, 2006.

[6] V. Smotlacha, "One-way delay measurement using NTP", CESNET, 2003.

[7] D. McDonald, C. Metz, and B. Phan, "RFC 2367 – PF_KEY key management API", Version 2, The Internet Society, 1998.

[8] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas, "Protocol independent multicast – sparse mode (PIM-SM)", The Internet Society, Aug. 2006.

[9] T. Aurisch and C. Karg, "A deamon for multicast internet key exchange", in *Proc. 28th Ann. IEEE Int. Conf. Loc. Comput. Netw. LCN'03*, Königswinter, Germany, 2003, p. 368ff.

[10] T. H. Tran, "Proactive multicast-based IPsec discovery protocol and multicast extension", in *Proc. IEEE MILCOM 2006 Conf.*, Washington, USA, 2006.

[11] H. Harney and C. Muckenhirn, "Request for comments 2093: group key management protocol (GKMP) architecture", IETF, 1997.

[12] M. Baugher, B. Weis, T. Hardjono, and T. Harney, "Request for comments 3747: the group domain of interpretation", IETF, 2003.

[13] Y. Kim, A. Perrig, and G. Tsudik, "Simple, and fault-tolerant key agreement for dynamic collaborative groups", in *7th ACM Conf. Comput. Commun. Secur.*, Athens, Greece, 2000, pp. 235–244.

[14] L. Dondeti, S. Mukherjee, and A. Samal, "A distributed group key management scheme for secure many-to-many communication", Tech. Rep. PINTL-TR-207-99, Department of Computer Science, University of Maryland, 1999.

[15] S. Mittra, "IOLUS: a framework for scalable secure multicasting", in *Proc. ACM SIGCOMM'97 Conf.*, Cannes, France, 1997, pp. 277–288.

[16] A. Ballardie, "Request for comments 1949: scalable multicast key distribution", IETF, 1998.

[17] D. Mills, "Request for comments 1305: network time protocol (Version 3) specification and analysis", IETF, 1992.

[18] T. Aurisch, "Using key trees for securing military multicast communication", in *Proc. IEEE MILCOM 2004 Conf.*, Monterey, USA, 2004.

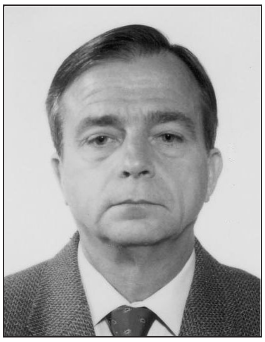
[19] T. Aurisch, "Optimization technique for military multicast key management", in *Proc. IEEE MILCOM 2005 Conf.*, Atlantic City, USA, 2005.

Thorsten Aurisch received his diploma degree in physics from the University of Bonn, Germany, in 1997. Since 1998 he works as scientist at the Research Establishment for Applied Science (FGAN). He has been involved in several national and international research projects related to military network design and planning. His research interests include security in wired and resource-restricted wireless networks.

e-mail: t.aurisch@fgan.de
Department Communication Systems
Research Establishment for Applied Science (FGAN)
Neuenahrer st 20
D-53343 Wachtberg-Werthhoven, Germany

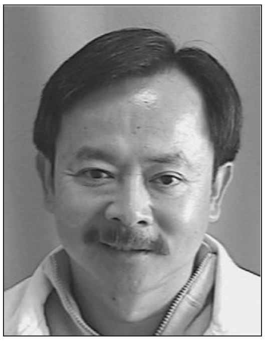
Tobias Ginzler received his diploma degree in computer science from the University of Bonn, Germany, in 2006. Since then he works as a scientist at the Research Establishment for Applied Science (FGAN) research facility. He is dedicated to network security and wireless technologies.

e-mail: ginzler@fgan.de
Department Communication Systems
Research Establishment for Applied Science (FGAN)
Neuenahrer st 20
D-53343 Wachtberg-Werthhoven, Germany



Roger Ogden has a B.Sc. degree in mathematics and a Masters degree in physics. He has been employed at SPAWAR Systems Center in San Diego, USA, and predecessor organizations since 1981. He has worked as a communications engineer in the area of Fleet network communications since 1996. He is the project manager of projects funded by the Office of Naval Research and the Office of the Secretary of Defense for the development of emerging networking protocols and implementations for tactical use especially in the area of secure, highly-mobile networks and also to better integrate legacy tactical data links with IP applications.

e-mail: roger.ogden@navy.mil
Space and Naval Warfare Systems Center
Hull st 53560
San Diego, CA 92152-5001, USA



Trung Tran has a B.Sc. degree in electrical engineering. He has worked at SPAWAR Systems Center in San Diego, USA, since 1989 and as a network communications engineer since 1992. His interests have been development of protocols and software applications for secure, highly-mobile tactical network communications. He was instrumental in the development of applications and implementations that allowed initial use of military tactical links for IP networking in the US Fleet. He holds two patents.

e-mail: trung.tran@navy.mil
Space and Naval Warfare Systems Center
Hull st 53560
San Diego, CA 92152-5001, USA

Hartmut Seifert received his diploma degree in communications technology from the Universität der Bundeswehr in Neubiberg, Germany, in 1979 and was serving afterwards up to 1987 as a technical officer within the German Airforce. Since 1987 he works as a scientist and as a program manager in several national and international research projects related to military network design and planning, including INSC, at IABG in Ottobrunn.

e-mail: seifert@iabg.de
Industrieanlagen-Betriebsgesellschaft mbH (IABG)
Einsteinstrasse 20
D-85521 Ottobrunn, Germany

Peter Martini – for biography, see this issue, p. 61.