Securing Host-Based Mobility and Multi-Homing Protocols against On-Path Attackers

Georg K. Hampel
Bell Labs/Alcatel-Lucent, Murray Hill, New Jersey, USA
Email: georg.hampel@alcatel-lucent.com

Vladimir Kolesnikov
Bell Labs/Alcatel-Lucent, Murray Hill, New Jersey, USA
Email: vladimir.kolesnikov@alcatel-lucent.com

***Abstract*—Host-based mobility and multi-homing protocols allow hosts to migrate ongoing transport sessions between networks or network interfaces. While such protocols can facilitate vertical mobility in a cost-efficient and access-agnostic manner, they are hard to secure when strong authentication between end points is not available. We present a balanced security solution which protects these protocols against redirection- and DoS attacks performed by on-path adversaries, while demanding only insignificant processing overhead on the end nodes. The solution is based on proof of session ownership using secret/answer chains as well as temporal separation and routability tests. It creates a level of protection that requires more (in some cases, significantly more) effort to break than conducting corresponding attacks through existing Internet signaling protocols. We discuss how this solution can strengthen the security of Multi-path TCP. We further show how it improves the security of route-optimized Mobile IPv6 while permitting operation without home agent.**

***Index Terms*— Communication System Security, Internet, Mobile communications, Mobile Radio Mobility Management, Security**

I. INTRODUCTION

The surge in mobile Internet traffic caused by the latest generation of smart phones forces research communities to rethink the existing concepts of IP mobility and multi-homing.

Fundamentally, mobility and multi-homing involve the matter of security, since the associated signaling messages can be misused by adversaries to redirect transport sessions for malicious purposes. Such redirection attacks allow the adversary to hijack a session and continue it on behalf of one of the end points. They can also be used for distributed Denial-of-Service (DoS) attacks where the adversary steers high volume traffic toward a victim host.

The available mechanisms to protect against such redirection attacks depend on the specific mobility technology used. The current 3G and 4G mobility standards such as W-CDMA, EVDO, LTE and WiMAX use network-based mobility. These technologies employ network-side anchors to relay all traffic between the Mobile Node (MN) and its peers. Since the mobility-related signaling messages are exchanged between MN and network, they can be secured using trust relationships that exist between subscriber and mobile network operator (MNO). When the MN moves to another network, the trust relationship is extended through roaming agreements between the MNOs of both networks. In case network operators do not support roaming agreements, vertical mobility is generally not supported. To support roaming from 3G/4G networks to untrusted WLANs, the 3GPP standards body introduced a solution, where all traffic is routed via an IPsec tunnel between MN and MN's home network [1] (Fig. 1). Such a solution introduces inefficiencies in routing, bandwidth- and network-resource utilization.

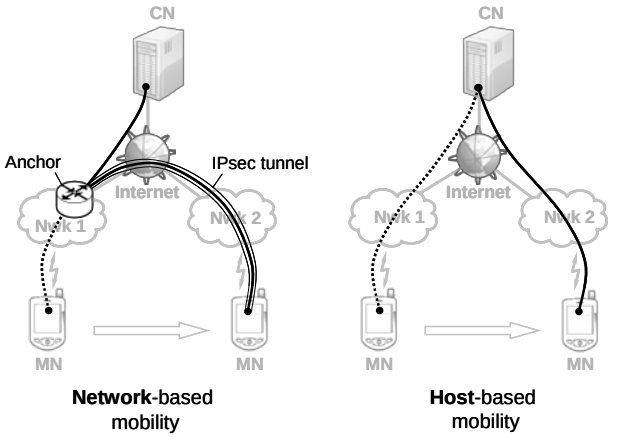

Figure1. Vertical mobility between networks in absence of roaming agreements

In contrast to network-based mobility, *host-based* mobility permits the MN to directly update the Correspondent Node (CN), when it changes its IP address (Fig. 1). In response, the CN sends its packets to MN's new address. Avoiding the need for anchors, host-based mobility is more cost-effective, versatile and scalable, and this has motivated multiple efforts to establish such protocols. Examples include the Host-Identifier Protocol (HIP) [2], route-optimized Mobile IPv6 (MIPv6) [3], TCP-R [4] and EMIPv6 [5]. A review comparing the various mobility models has been provided by [6].

Some host-based protocols support multi-homing scenarios, where the host announces multiple IP addresses to its CN as alternative routing paths. This

Manuscript received November 11, 2011; accepted January 21, 2011.

doi:10.4304/jcm.6.1.101-114

permits session migration in a make-before-break manner, and increases session reliability due to the availability of additional fallback paths. Protocols that support such scenarios are SCTP [7], multi-homed TCP [8] and SHIM6 [9]. Another protocol, referred to as Multipath TCP (MPTCP), splices traffic along multiple transport paths, which improves the aggregate session throughput [10]. Since IP mobility and multi-homing address highly related issues, we mostly refer to them as *mobility* when addressing both phenomena.

Since the Internet does not enforce strong authentication between hosts, such mechanisms cannot be used to secure host-based mobility protocols. There are exceptions where such methods are applied to protect confidential information, e.g., in on-line banking and VoIP calls. The majority of traffic, however, is not secured using strong authentication because it is not considered necessary or the associated overhead cannot be justified economically.

Frequently, the user applies implicit forms of authentication based on her recognition of patterns in the data stream such as the visuals on a webpage, the voice of a well-known speaker or the expected environment shown on a video stream. While these forms of implicit authentication are by no means "strong", they are often hard to gauge and create a significant burden for an adversary to overcome. Since such methods are usually applied when the session starts, the adversary may prefer instead to hijack the session at a later point in time.

Such hijacking attacks can be thwarted by security measures that build on session ownership rather that host authenticity. For that reason, many host-based protocols have resorted to methods of weak authentication, such as random-nonce exchanges, key arrangement through Diffie-Hellman exchanges and routability tests [11]. We argue that the present solutions either provide insufficient protection, or they put an unnecessary processing burden onto the end nodes.

Our Contribution: We present a security solution, which sufficiently protects host-based mobility protocols when strong authentication is not available. Our solution focuses on on-path attackers that have passive control of the traffic stream or are of transient nature. We also consider situations where the attacker represents the end point of the traffic session itself to commit flood attacks.

Our solution is lightweight as it does not create a large burden in processing or state information to be held on either end node. The setup cost is very low so that mobility support can be justified even for short-lived connections. At the same time, our solution provides sufficient protection in the context of the existing security level of the Internet. That is, for the redirection attacks we consider, our protection constitutes a barrier at least as high as that provided by the underlying traffic protocols. From another angle, it is easier for the attacker to hijack a session or commit a flood attack using the existing traffic protocols rather than our secured mobility protocol.

Outline of the presentation: In the next section, we examine the value proposition for host-based mobility to emphasize the relevance of our work. In Section 3, we discuss the principal attacks achievable with mobility-related signaling, and how an adversary can accomplish the same malicious goals through other methods. Section 4 addresses related work. Section 5 presents our security solution for host-based mobility protocols. In section 6, we discuss how this security solution can protect the state-of-the-art protocols MPTCP and MIPv6. The conclusion in Section 7 summarizes the work.

II. The Case for Host-Based Mobility

While many host-based mobility solutions have been proposed in the past 15 years, none of them has gained substantial market traction. Instead, all commercial mobility networks use network-based protocols. Given the technical advantages of host-based mobility, the lack of market success seems to be a mystery.

The reasons for this have to be sought in the historical development of mobile data communications and IP-protocols. The present mobile data networks have their roots in cellular telephony, which relies on circuit-switched connections supported by a hierarchical network of centralized nodes holding host-related state information.

The Internet protocol is based on a different architecture. Applying the principle of stateless routing, it decentralizes network operations and keeps host-specific information confined to the network edge. The advantages such as scalability, reliability and low cost allowed the Internet to grow at rapid pace.

Host-based mobility is compliant with the Internet's principal paradigm of stateless routing, since all relevant state information is confined to the end-nodes. Mobility support, however, was not built into TCP/IP. Retrofitting the protocol with mobility at the present stage is possible but it encounters two major problems.

Firstly, a large number of hosts and services on the Internet will have to be upgraded. This makes an incremental deployment very difficult. Secondly, inter-host signaling messages are hard to secure in the absence of inter-host trust relationships and means of strong authentication. The IETF MIPv6 workgroup argued that the integrity of the entire Internet would be at risk if these security problems could not be appropriately addressed [12]. While the security issue is subject of this paper, we'd like to spend a few lines to dispel the deployment problem.

Deployment of host-based mobility protocols was problematic at a time when the Internet had grown to considerable size, but only few mobile hosts engaged in data services. Under such circumstances, it was easier to support mobile data traffic via anchors and leave the vast majority of stationary hosts unaffected. The recent growth of mobile data traffic, however, shifts the weight into the opposite direction, favoring solutions for host-based mobility. Further, only Internet services which

engage into data connections with extended session times need to provide host-based mobility support. Presently, this applies to streaming applications as well as real-time- and conversational services. Typical web browsing or email-server interactions are mostly too short-lived to justify vertical mobility maneuvers. Host-based mobility protocols can further alleviate the transition by incorporating the existing anchor-based technologies as a fallback solution.

Presently, the mere load on cellular networks has made MNOs seek for alternatives to offload traffic from their air-interface and core network. The 3GPP standard body has acknowledged the severity of this issue and introduced traffic offload scenarios to relieve the operators licensed spectrum and cellular infrastructure [13]. Some of these offload scenarios sacrifice mobility for the sake of scalability and lower cost. Other developments toward increased operator diversification and network neutrality will also play in the favor of host-based mobility.

While these reasons motivate the rationale for host-based mobility at present, such technology may facilitate a principal transformation of communication networks toward leaner and simpler mobility architectures in the future.

III. Security Threats in Context

Host-based mobility allows mobile nodes (MNs) to change their IP addresses underneath ongoing transport sessions and to directly inform their correspondent (CNs) about this change through signaling messages. These messages are commonly referred to as Binding Updates (BUs) since they carry information about the binding between session endpoint and the MN's current location.

When receiving a BU message, CN updates the binding cache it holds for the corresponding MN. At the same time, it may also activate one of the new bindings, i.e. steer outgoing traffic toward the corresponding destination address and ensure that packets received from this address reach the local session end point.

Note that this principal methodology applies to all host-based mobility or multi-homing protocols independently of their protocol layer of operation (e.g. network-, transport- or session layer), features and design details.

An adversary can use the signaling of such protocols to conduct redirection attacks, i.e. forge or replay BUs to redirect traffic for malicious purposes. One commonly differentiates among three goals such attacks may pursue [10]:

- Session interruption: The attacker interrupts an existing traffic session (DoS).
- Session hijacking: The attacker continues the session on behalf of the MN.
- Flood attack: The attacker creates a flood of traffic onto a victim host (distributed DoS).

The adversary can interrupt or hijack an existing transport session by sending a forged BU with MN's credentials to CN. In response, CN directs the traffic toward another location, which interrupts the traffic flow between CN and MN. In case the adversary has directed the traffic to its own location, he can potentially continue the session on behalf of MN.

To commit a flood attack, the adversary creates traffic sessions with a set of servers, which return a high volume flow of traffic in response. Then the adversary sends BUs to the servers to direct the traffic to a victim host. The attacker can create these flows sequentially to avoid a flood on himself.

Every security solution that protects mobility protocols against such attacks has to be gauged upon the underlying vulnerability of the network itself. This means that attacks via the secured mobility protocol should be as hard or harder to execute than using methods that exploit existing signaling protocols to accomplish the same goals. These latter methods can be summarized as follows.

Session interruption: To interrupt a session, the attacker can send to either MN or CN a control message such as ICMP Port/Destination Unreachable, TCP FIN, TCP RST or RTCP BYE while spoofing the peer's IP address. For this purpose, the attacker has to be able to:

(a) Eavesdrop on either direction of the path between both hosts to obtain information on the specific session.
(b) Send packets to one of the hosts spoofing the peer's IP address.

While, in principle, ingress filtering by intermediate routers protects against IP-address spoofing, this feature is not generally enforced.

Session hijacking is equivalent to continuing the session on behalf of one of the end points. We assume that the attacker wishes to continue the session with CN on behalf of MN. This is equivalent to sending a forged BU to CN on behalf of MN. For this purpose, the adversary must:

(c) Eavesdrop on all packets sent by CN to MN, i.e. reside on the path from CN to MN
(d) Send packets to CN on behalf of MN while spoofing MN's address.
(e) Suppress all further packets sent by MN to CN.

Conditions (c) and (d) are equivalent to (a) and (b) but they apply to only one flow direction, in contrast to the DoS attack, which can be performed on either of the two hosts.

Condition (e) is much harder to meet than the prior conditions since it requires from the attacker to have active control over the traffic stream. Alternatively, condition (e) may be met by the attacker performing a DoS attack on MN by sending a TCP FIN or RTCP BYE, which would end MN's traffic flow. In this case, the attacker must suppress MN's response, which carries a TCP FIN or RTCP BYE and would break the connection he wants to hijack. Therefore, this method also requires

active control over the traffic stream (although possibly a weaker, transient form of active control).

For TCP, one special method of session hijacking has been discussed, which uses sequence number de-synchronization [13]. The on-path attacker eavesdrops on the sequence numbers of packets sent by MN to CN. Then he inserts his own packets with sequence numbers expected next by CN. CN accepts these packets and reject MN's future packets since they have lower sequence numbers and appear as outdated duplicates.

While this hijacking method is well known, it can be easily averted. Since CN acknowledges the sequence numbers of the attacker's packets, MN receives acknowledgements to packets it has not yet sent. At this point, MN should send a RST to terminate the session. To overcome the RST message, the attacker needs to have active control over the traffic stream again.

Flood attack: To create a flood attack onto a victim host, the attacker can create multiple high-volume sessions with a set of servers while spoofing the victim's IP address. We make the additional assumption that session establishment with any of these servers involves an initial handshake. For that purpose, the attacker must:

(f) Eavesdrop on the paths from the servers to the victim in order to exercise the handshake with the servers on behalf of the victim.

(g) Spoof the victim's IP address to give the servers the impression they talked to the victim itself.

A summary of these threats is discussed in [12] and [14]. Conditions (a) to (g) are the basic capabilities required for an adversary to perform the corresponding attacks. We will use them as a measure of hardness to set the minimum level to the design of a secure solution to host-based mobility.

Conditions (a) to (e) contemplate situations where the adversary resides on the traffic path between MN and CN. It is also possible that off-path adversaries commit session-interruption or session-hijacking attacks. When off path, the adversary has to guess all information pertaining to the ongoing traffic session, such as IP addresses, port numbers and eventually sequence numbers. Further, forged packets carrying spoofed IP addresses become subject to ingress filtering by intermediate routers.

Although off-path attacks seem much harder to conduct than on-path attacks, they may be rather straightforward when security measures are not appropriately applied. In case administrative permissions on a host are set incorrectly, off-path attackers may well be able to set up a control session to either MN or CN (e.g. Telnet), which gives them access to system information and ongoing traffic sessions. Session information pertaining to IP addresses and port numbers may also be publicly known for some well-known applications. Even the introduction of randomness may provide limited protection in case the pseudo-random algorithm is publicly known and the seed is based on (fully or partially) publicly available parameters, such as time. This, for instance, applies to some open-source operating systems.

Off-path attacks have frequently been discussed in the context of TCP hijacking. We spend only limited attention to such attacks since they can be averted by properly restricting administrative access to the end hosts and by incorporating at least one random number into each traffic session, which cannot be easily guessed. While the former is a mere configuration matter, the latter needs a little attention and is addressed further below.

IV. Related Work

Since many protocols to host-based mobility or multi-homing have been proposed, a large number of security solutions is available. The individual security measures applied fall into the following categories:

- Strong authentication through pre-shared keys or PKI using trust relationships.
- Session key establishment through Diffie-Hellman exchange.
- Weak authentication using random numbers (token, nonce, cookie).
- Routability test using challenge/response based on random numbers.

Methods of strong authentication are enforced by HIP through the use of PKI. Since strong authentication is considered cryptographically secure, it protects against redirection attacks using forged BUs (i.e. DoS attacks and session hijacking). The availability of strong authentication, however, requires mutual trust relationships between MN and CN as prerequisite. HIP requires trust relationships to be enforced among all hosts supporting the protocol. It has been recognized, however, that the associated effort is tremendous and tends to outweigh the principal benefits of host-based mobility [16]. EMIPv6 and SHIMv6 use cryptographically generated addresses (CGA)[17] and hash-based addresses (HBA)[18]. Both approaches are based on public keys but circumvent the need for PKI. Although these approaches are cryptographically secure, they only apply to IPv6, and they require stateless auto-configuration.

As the next weaker alternative to strong authentication, the protocols TCP Migrate [19], MAST [20] and a session-layer solution by [21] propose key establishment through elliptic-curve Diffie-Hellman (DH) exchange. The DH key exchange is vulnerable to Man-in-the-Middle (MitM) attacks, where the MitM has active control over the traffic flow in both directions for the entire duration of the session. This requirement is stronger than conditions (a) to (e). The DH key exchange can therefore be considered sufficiently secure to protect mobility protocols against session interruption and session hijacking.

The DH key exchange demands substantial processing efforts on each host at session beginning. This can become a burden for many mobile devices that have limited processing capabilities. The processing cost incurs even if neither of the hosts undergoes a mobility event because the session is rather short or the hosts move at low speed. Even if one of the hosts knows that it is always stationary, it must carry the same processing burden as its mobile peer. Also, the processing-intense DH key exchange may create vulnerability itself, in case an adversary creates a flood of mobility sessions from different locations.

All other proposals, such as SCTP, MIPv6, TCP-R, MH TCP, Lin6 [22] and MPTCP [23] secure messages through random numbers that represent host-, session- or handshake identifiers and are often referred to as cookies, tokens or nonces. The protection such random parameters provide can be easily broken through eavesdropping. This allows an adversary to forge BUs for the purpose of interruption or hijacking of traffic connections. Note that breaking these mechanisms does not require active control of the traffic stream as necessary under condition (e). Therefore, these protocols add vulnerability to the network and do not meet our security requirements. Some protocols, like MIPv6, provide obfuscated solutions where multiple random numbers (tokens, nonces, etc.) are exchanged and mutually entangled. This, however, does not mitigate the principal vulnerability of this approach.

While strong authentication and DH key exchange provide enough security to protect against session interruption and session hijacking, they cannot prevent flood attacks. Since in such attacks, the adversary is the session owner he is entitled to send BUs. Many mobility protocols (e.g. MIPv6, EMIP6, HIP and TCP Migrate) insert a routability test into the BU handshake, which allows CN to verify that MN is actually reachable at the new IP address. For this purpose, CN sends a challenge, e.g. a random number, to MN which MN simply returns in a response message. In case MN is synonymous with the attacker and attempts to redirect CN's traffic stream to a victim host, the victim would receive CN's secret and simply not respond.

In order to break this protection provided by the routability test, the attacker, i.e. MN, has to eavesdrop on the link between CN and the victim. This allows the attacker to receive the challenge and respond on behalf of the victim while spoofing the victim's IP address. This effort is the same as for conditions (f) and (g) required to conduct the flood attack without mobility signaling. The routability test therefore provides equal protection when compared to the security level of present networks.

Some mobility protocols introduce additional processing to be conducted by the responder to the challenge (e.g. MIPv6). This does not improve the principal level of security against this type of flood attack, since the attacker as the legitimate session owner holds all capabilities to provide the desired response.

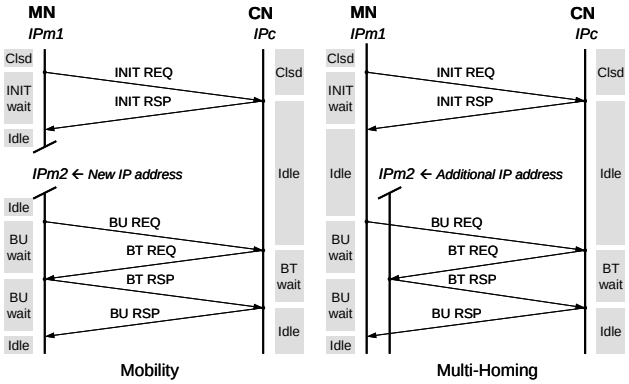

Figure 2. Signaling messages to support secure mobility and multi-homing protocols. Grey bars indicate the states according to Fig. 3.

V. PROPOSED SECURITY SOLUTION

We assume that strong authentication is not available on the protocol layer where the mobility-related functionality is exercised. Further, the users of the traffic session may apply some form of implicit or explicit authentication when the session starts. An adversary may therefore be tempted to hijack the session *after* it has started. To thwart such hijacking attacks, the mobility protocol has to protect the end-point's session ownership throughout the mobility events to follow. The strength of this guarantee has to be measured upon alternative means of session hijacking given by conditions (a) to (e).

In case the adversary plans to commit a flood attack and becomes the session owner, conditions (f) and (g) serve as the reference for protection.

A. Mobility Association

We make the logical distinction between the mobility session, which supports mobility-related signaling and security between two hosts, and the traffic session that runs in parallel. The actual linkage between both entities is discussed later.

Each mobility session defines at least *one* mobility association (MA), which is owned by one of the two hosts and furnishes this host with the rights to migrate the end points of its traffic connections that are linked to the mobility session. In case both hosts are mobile, each of them has to support its own MA within the same mobility session.

Permitting mobility sessions with only one MA improves security since session hijacking cannot be performed on behalf of the stationary node. This advantage is substantial since a large fraction of mobile Internet traffic involves stationary servers. Session hijacking to extort information from the mobile client is therefore averted.

The mobility session is established via an initial handshake we refer to as INIT (Fig. 2). Such a handshake is supported by all host-based mobility and multi-homing protocols. The INIT handshake defines the mobility session, the support of MAs and the exchange of the initial security parameters.

After INIT, all further handshakes occur on behalf of *one* MA, i.e. they only affect mobility functions pertaining to the host who announces the mobility event and who owns that MA. Both hosts can therefore exercise temporally overlapping mobility events, which is an important feature given the absence of supportive infrastructure on the network, such as anchors. Some minor effort has to be taken to ensure the synchronization of security-related information on each side and to disentangle retransmissions from message replays.

While the definition of the MA may be reminiscent to that of the Security Association (SA) used for protocols like IPsec and TLS, the properties differ substantially. An SA maintains strong security material pertaining to one simplex direction. An MA solely provides the means and the rights (protected by relatively weak security) of one host to migrate its session endpoint. In contrast to SAs, the MAs of both hosts can share security material.

B. Session Interruption and Hijacking

For protection of mobility-related signaling, we assume that each mobility-related exchange of information requires at least a two-way handshake, consisting of request and response messages, which we refer to as a Binding Update (BU) exchange (Fig.2).

The BU REQ is sent by one end host to announce addition, deletion or change of an IP address and it is acknowledged by the peer in the BU RSP. Such handshakes are generally in compliance with host-based mobility protocols.

To protect against DoS and session hijacking, the sender of the request has to assert that he is the owner of the mobility-session end point. This is equivalent to providing a proof that the sender has:

- generated and sent all prior mobility-related requests, and
- received all prior mobility-related replies,

which includes the initial INIT exchange messages. Note that the term "proof" is not meant to carry absolute mathematical weight. It merely refers to a level of protection that requires same or more effort to break than conducting equivalent attacks through existing signaling protocols. This can be accomplished in the following manner:

- The host authenticates its request by revealing a secret related to his previous request. This will create a "chain of proofs" of the same sender. The secret can be a cryptographic hash over a random number *R* generated in every request.
- The host signs the request with a symmetric authentication key, which is updated by the peer in every reply message. This proves that the host has received all prior replies. Such a method is frequently referred to as "temporal separation" [10].

The following shows the INIT handshake for a mobility session, where Host 1 initiates the mobility session:

INIT HANDSHAKE:

Host1		**Host2**
[Generate and cache MA identifier *MID*1]		
Generate and cache random number $R1_0$		
Generate and cache random key $A2_0$		
	REQ: [*MID*1], Hash($R1_0$), $A2_0$ ➔	
		[Cache *MID*1]
		Cache Hash($R1_0$)
		Cache $A2_0$
		[Generate and cache MA identifier *MID*2]
		Generate and cache random number $R2_0$
		Generate and cache random key $A1_0$
	← **RSP**: [*MID*2], Hash($R2_0$), $A1_0$	
[Cache *MID*2]		
Cache Hash($R2_0$)		
Cache $A1_0$		

The *MID* parameter is a random number, which represents the sender's MA identifier. It is used as a reference to the MA even when messages arrive from different IP addresses. One *MID* has to be introduced for each MA. If the host is stationary and does not support an MA, the parameter is omitted or set to zero, which automatically prohibits future mobility support for this host.

In case both hosts are stationary, the session initiator (Host 1) still has to send an INIT REQ since it does not know if its peer (Host 2) wishes to support mobility. Since Host 2 is also stationary, it sends an INIT RSP without (or with zero-valued) *MID2*. The response has to be sent to avoid retransmissions of INIT REQs by Host 1. After this handshake, both hosts resort to conventional transport without further mobility signaling.

$R1_0$ and $R2_0$ represent the secrets of Host1 and Host2, respectively. Since the hosts only send a cryptographic hash of the secret, an adversary eavesdropping on the message is not able to guess the *R*-value itself. The corresponding hash function can be SHA1, for instance.

$A1_0$ and $A2_0$ represent the initial values of the keys used to sign further message exchanges. *A*1 is used to protect handshakes initiated by Host1 and *A*2 vise versa. Note that the initial values of the keys are exchanged openly.

Since the INIT handshake only requires generation and caching of random numbers, it is very lightweight. This makes it compliant with frequent setup and tear-down of data sessions, as exercised in data communications nowadays, where session live time and mobility support are hard to gauge. The lightweight setup procedure also makes the host less vulnerable to a flood attack of setup requests.

For the following handshakes and without losing generality, we assume that Host1 is MN and Host2 is CN. MN's *i*th BU handshake looks as follows:

BU HANDSHAKE (index i):

MN **CN**

Generate and cache random number $R1_i$

BU REQ: $MID1$, $R1_{i-1}$, Hash($R1_i$), HMAC($A1_{i-1}$ | MSG) ➔

Verify cached Hash($R1_{i-1}$)

Verify HMAC($A1_{i-1}$ | MSG)

Cache Hash($R1_i$)

Generate random key $dA1_i$

Cache $A1_i = A1_{i-1} \oplus dA1_i$

➔ **BU RSP**: $MID1$, $dA1_i$, HMAC($A1_{i-1}$ | MSG)

Verify HMAC($A1_{i-1}$ | MSG)

Cache $A1_i = A1_{i-1} \oplus dA1_i$

The index i-1 refers to MN's previous BU or to INIT for i=1. The BU handshake also contains information relevant to the mobility event itself such as MN's new IP address or other parameters, for instance. The details depend on the specific mobility protocol and are not shown here.

The term HMAC(A | MSG) represents a hash-based- or any other kind of message authentication code that is based on key A and applied to the relevant part of the message body, referred to as MSG. The HMAC can be based on a SHA family member, as proposed by [24]. The ⊕ symbol refers to a bit-wise XOR operation. Any other operation, e.g., addition in a group, can be chosen that fully randomizes the result if any one of the inputs is random. We chose the XOR operation for its efficiency.

Note that the BU handshake only holds security parameters related to the handshake's initiator, i.e. $R1$ and $A1$ in the present example. A handshake initiated by CN (=Host2) would only refer to the corresponding parameters $R2$ and $A2$. This separation permits secure signaling when message handshakes initiated on both sides temporally overlap.

In contrast with the security parameters, the MID contained in the handshake refers to the MA rather than the handshake's initiator. For the above BU, which provides new information on behalf of MN (=Host1), both entities, i.e. handshake initiator and MA owner are the same. We show another example below, where this is not the case.

To break the protection of the presented security mechanism, an attacker must:

- Eavesdrop on the path between both hosts,
- Wait until the next BU arrives from the host, whose end point he wishes to hijack, and stop the propagation of this message,
- Reinsert a copy of this message with a new, self-chosen value for Hash(R), while spoofing the host's IP address.
- Listen to all packets sent by the peer since INIT to obtain knowledge of the present value of A and sign the forged request.

This effort matches conditions (a) to (e) for session-interruption and session-hijacking. Note that the chain of BU handshakes provides an even higher protection since the attacker has to be present for the entire time frame since INIT, and, further, must wait for a legitimate BU request to be issued by the right host. This requirement does not apply to conditions (a) – (e), where the attacker can step in at any time and take over the session.

In the above BU handshake, CN's reply is signed by the updated key A. This signature does not provide any additional security in case an attacker has broken key A and rewritten Hash(R). However, it allows MN to mitigate DoS attacks onto the mobility session itself. That is, without the HMAC, a transient attacker could easily break the mobility session by sending an arbitrary key update.

C. *Protection against Flood Attacks*

When an adversary uses mobility signaling to conduct flood attacks, securing BU messages is of no help since the adversary himself establishes the mobility session. Therefore, our security mechanism uses a simple routability test we refer to as Binding Test (BT). The BT handshake consists of a BT REQ message sent by CN to MN's new address and a BT RSP message in reverse direction.

Note that CN has to send the BT REQ to MN's new address even if the BU request was sent from the old address (Fig. 2). In case CN itself sustains multiple IP addresses, it is sufficient to use only one of them for the routability test since this ensures that MN can be reached at the new location. (There may be reasons of technical nature to conduct routability tests from all of CN's IP addresses to MN's new address. Such a procedure may, for instance, determine availability of certain paths).

The routability test involves a challenge sent by CN to MN, which has to be answered in a response by MN. CN has to ensure completion of the BT handshake before it sends larger amounts of traffic to MN's new IP address.

The BT handshake can be embedded into the above BU handshake (Fig.2):

COMBINED BU - BT HANDSHAKE

MN **CN**

…

BU REQ: $MID1$, $R1_{i-1}$, Hash($R1_i$), HMAC($A1_{i-1}$ | MSG) ➔

…

Create challenge, i.e. random number X

➔ **BT REQ**: $MID1$, $R2_{j-1}$, Hash($R2_j$), HMAC($A2_{j-1}$ |MSG), X

BT RSP: $MID1$, $dA2_j$, HMAC($A2_{j-1}$ | X) ➔

Verify response: HMAC($A2_{j-1}$ | X)

…

➔ **BU RSP**: $MID1$, $dA1_i$, HMAC($A1_{i-1}$ | MSG)

The challenge contains the random number X, which CN creates solely for this BT handshake. MN returns the signature over X to prove that its new address is routable.

Note that CN uses its own secret $R2$ and key $A2$ for the BT exchange since it is the initiator of this handshake. The BU handshake is initiated by MN and therefore uses MN's secret $R1$ and $A1$. While these handshakes use different keying material, they both hold the same MA identifier $MID1$ since they both pertain to MN's mobility

event.

The security parameters in the BT REQ are necessary to authenticate CN to MN. Otherwise, an on-path adversary could hijack the session by sending a forged BT REQ to MN spoofing CN's address. If the forged BT REQ arrives before the legitimate BT REQ the mobile node replies to the wrong message. While CN ignores MN's new address since its own routability test has failed, the adversary can continue communicating with MN. (Note that in another publication [25], we proposed a simpler mobility protocol, which did not require the protection against this specific attack. Here we provide stronger security at the cost of a slight increase in complexity.)

The HMAC in the BT RSP message protects against DoS attacks on the mobility session and hijacking attacks in the presence of NATs as discussed below.

Ideally, MN's new IP address should not be used until the BT has completed successfully. This means that CN must have received the BT RSP and MN the BU RSP. While this restriction is acceptable for multi-homed hosts, it creates significant delay in break-before-make mobility scenarios, where the old link is discontinued before the new link can be established. To reduce this delay, it has been proposed to allow limited traffic during the BT handshake in such scenarios [26]. From the security perspective, the associated performance improvement has to be weighted against the potential damage this may cause if used for a flood attack. The size and severity of such a flood may be considered small as long as the time interval is limited to typical BT completion times (a few seconds at most).

D. Retransmissions and Replay Attacks

When one of the above BU messages fails, the BU handshake does not complete. In this case, MN should engage into retransmission. The retransmitted message should be an exact copy of the prior message. In case CN did receive the prior request but the reply to MN got lost, the validation of $R1_{i-1}$ will fail since CN has already been updated with the new Hash($R1_i$). Therefore, CN needs to keep a copy of the old Hash($R1_{i-1}$) so that it can also validate request retransmissions.

CN should respond to retransmitted requests with a retransmission of the prior response. For this purpose, CN has to cache *dA* for some time frame after the original response was sent.

To make the retransmission policy robust against replay attacks, retransmissions must not invoke any change on the receiving host's security- or mobility-related state information. Instead, the reception of a retransmitted request should only cause the retransmission of the corresponding reply. This guarantees that replay attacks of manipulated messages have no impact.

In case messages get delayed but not lost, MN may invoke retransmission of requests and consequently receive muliple replies. Since the key *A* was already updated in the first reply, the HMAC validation of the retransmitted reply will fail. This has no further consequences. Therefore, MN does not have to keep a copy of the old value for *A*.

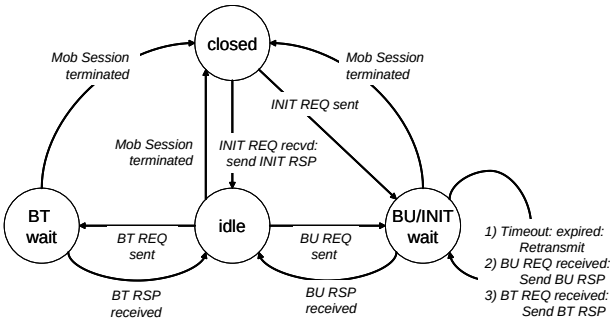

Figure 3: State machine of security solution

In contrast with BU REQs, BT REQs should not be retransmitted. In case BT REQ or BT RSP get lost, MN should retransmit the initial BU REQ, instead.

E. State Machine

Fig. 3 shows the state machine for security-related messages. The corresponding states are also included in the message flows on Fig.2.

While the presented security solution focuses on a simple set of mobility messages, i.e. IP address announcements and routability tests, host-based mobility protocols usually provide a more refined set of message exchanges. These message exchanges can be mapped onto the same security mechanism. Generally, handshakes initiated by the MA owner map onto BUs while those initiated by the MA owner's peer map onto BTs. Examples are given in the next section.

F. Compliance with Network Address Translators

While the HMAC's principal purpose is to prove the sender's knowledge of key *A*, it cannot cover the message's IP- and transport headers. Otherwise, mobility messages would not pass through Network Address Translators (NAT), where these headers are rewritten. We can show that this limitation does not reduce the protection of the proposed security protocol.

In case MN announces a new IP address pertaining to a private network, CN has to derive the corresponding public IP address from the IP-header of MN's BU REQ (mostly, MN only knows its private but not its public address). Since the packet header is not covered by the HMAC, a transient on-path attacker with active control over the traffic is capable to overwrite the header entry with his own IP address. Consequently, CN receiving the BU REQ assumes that MN resides at the attacker's address.

This hijacking attempt, however, will fail since the transient attacker cannot properly reply to the consecutive BT REQ sent by CN, since this requires knowledge of key *A* to accurately compute the HMAC.

This example demonstrates that our security solution is compliant with operation across NATs. It further shows

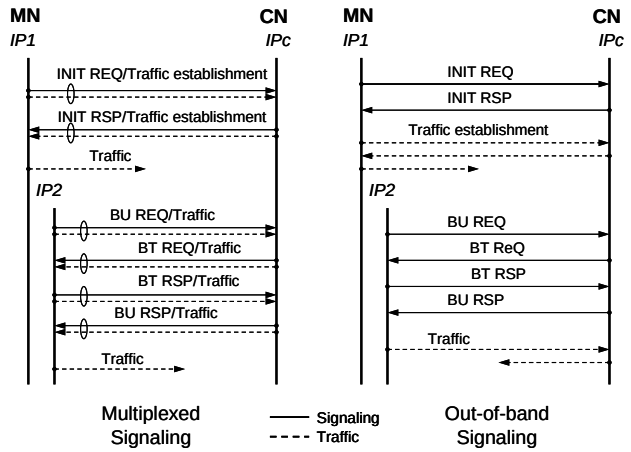

Figure 4. Linking traffic connections to mobility session

that the BT handshake is necessary to protect against both flood attacks as well as session hijacking.

G. Linking Transport- and Mobility Sessions

While the mobility session can be sufficiently protected against various forms of attacks, it must be properly linked to the traffic connections sustained between both hosts. Otherwise, an attacker can take control over traffic connections between MN and CN by overlaying a mobility session with CN while spoofing MN's IP address. Since CN believes that it shares traffic connection and mobility session with MN, the attacker can use its ownership of the mobility session to hijack the traffic connection.

Such an attack does not require active control over the traffic flow according to condition (e). It is therefore easier to conduct than breaking the protection of the mobility session itself.

There are two distinct methods of linking traffic connections to mobility sessions: In one, mobility-related signaling is multiplexed onto the traffic connection. In the other, mobility control is conducted through out-of-band signaling. Both solutions have been exercised in the literature and in standard proposals. A set of principal rules can be established on how to securely link both properties.

The following rules apply for out-of-band signaling (Fig. 4):

- The mobility session must undergo a signaling exchange directly before establishment of the traffic connection that is to be linked to it. This signaling can be an INIT- or BU handshake. In this message, it must explicitly announce the intention to link the traffic connection.
- Signaling and traffic connection must initially use the same IP addresses. The signaling can refer to the traffic connection by a set of traffic connection identifiers such as port numbers and protocol type.
- The traffic connection must be established within a short time frame after the signaling handshake that is announcing it. If it is not, CN can simply reject the mobility support for this traffic connection.
- Mobility sessions should not run idle without traffic connections linked to them. They must be torn down soon after the corresponding traffic connections have been terminated. This prohibits an adversary to preemptively create a mobility session with CN spoofing MN's address to hijack future traffic connections.

The principal security of this rule set relies on the temporal sequence of events. It assumes that an adversary does not know if and when the host plans to start a traffic connection with its peer. The protection of this linkage can be improved if connection-specific details are added to the linking message, such as the randomly chosen port number of the traffic source.

Additional attention to this rule set is necessary when the message initiator resides behind a NAT. In this case, multiple hosts behind the same NAT can map to the same public IP address, which makes the mapping between the source address of mobility session and transport connection ambiguous. Further, information provided by the session initiator on the source port number of the traffic connection becomes meaningless since it is altered by the NAT. The only way to overcome this ambiguity is through the explicit mapping of the transport connection to the mobility session. This requires that at least some signaling information (e.g. *MID*) is multiplexed onto the transport connection.

The linkage is easier for in-band (or multiplexed) signaling. Although it is convenient to create one mobility session per traffic session in this case, such a requirement is not imperative (see solution for MIPv6 in the next section, for instance). We therefore assume with more generality that a mobility session can support multiple transport sessions:

- The mobility session must enclose a signaling handshake on the *first packet* sent in each direction. The signaling can be an INIT- or BU handshake. It implicitly announces the intention to link the corresponding transport connection.
- As for out-of-band signaling, the linkage of mobility sessions should not run idle without traffic connections that are linked to them.

The interpretation of the term "first packet" depends on the protocol layer, where the host-based mobility feature is implemented. For TCP connections and protocols operating on network- or transport layer (such as MIPv6 or MPTCP, respectively), the first packets are those carrying the SYN flag. For session-layer solutions that multiplex signaling and data traffic in the packet's payload [21], the first packet exchange occurs when the SYN/ACK procedure has been completed, i.e. the socket is connected.

When hosts reside behind a NAT and link multiple traffic connections to one mobility session, the same ambiguity exists as discussed for out-of-band signaling. Hence, the same principal solutions can be applied.

When each transport connection supports its own

mobility session, INIT is multiplexed onto the first packets exchanged between both hosts. Since all further BU and BT messages are multiplexed onto the traffic session to which they apply, all ambiguities due to NATs are avoided.

H. Random Number Generation

All random numbers, i.e. secret R, key A and challenge X, can be created via pseudo random-number generators (PRG) with strong randomness properties. The PRG must be initialized with a random seed, not guessable by an adversary. This is necessary to avoid off-path attacks.

Many pseudo-random generators derive the seed from the local CPU time. This alone may be insufficient if the seed is generated within a short or predictable timeframe before the random parameters are transmitted in a signaling message. It is possible, however, to incorporate local parameters into seed generation [27].

The length of the random numbers has to be chosen large enough to avert brute force attacks. A value of L = 128 bits is more than sufficient [27]. While most of the secure algorithms for cryptographic Hash generation produce numbers that are substantially longer (160 bits for SHA1), they can be truncated to the length of L before they are included in the message. The same accounts for HMAC values. (Here we assume that the "strength", or hard bits of SHA and HMAC are distributed "evenly" in the output of SHA/HMAC. We stress that this is a reasonable assumption, and is weaker than the very commonly used Random Oracle assumption.) We note that, if message length is a constraint, we can use shorter randomness, e.g. of size 50-60 bits. This is because brute-forcing 60 bit keys is still a massive effort, much harder than other attack avenues. We add that the meet-in-the-middle attacks on hash functions, which compute collisions and run in square-root-time in the size of the message space, are not useful for attacking our protocols.

I. Session Hijacking of Encrypted Traffic

While our security mechanism sufficiently protects the migration of conventional transport connections, where session ownership is the core property of concern, the question arises to what extend it can cover mobility of IPsec- or TLS-secured traffic [27] [29]. MobIKE [30], for instance, supports a host-based mobility solution for IPsec-protected traffic, where the mobility-related signaling messages are inserted into the IKEv2 exchange and therefore enjoy higher level of protection.

The question carries a technical- as well as security-related component, since it involves the relative layering of mobility support and transport encryption in the protocol stack. When mobility support is provided *above* the transport protection, a separate SA has to be established for each transport path. This applies, for instance, when TCP-Migrate is paired with bump-in-the-stack IPsec implementations. Since every SA-protected transport path is stationary, no security concerns should arise.

When mobility support occurs *below* transport encryption, the transport path can be changed underneath the protected transport session without using the strong authenticators and keying material established for encryption purposes. This applies, for instance, to MIPv6-supported TLS transport sessions.

We claim that in this latter scenario, mobility support also does not create additional vulnerability since the transport protection already provides implicit protection against session hijacking. In case an adversary is a MitM with active control over the traffic flow, he cannot overcome the encryption applied by the end nodes. Therefore, he cannot continue the session on behalf of one of the end points even if he steers the session to his own location. Such a session hijacking attempt would solely result in a DoS attack, which could be accomplished by easier means (e.g. TCP FIN or RST).

The adversary may see a benefit in redirecting the secured transport data to his own location so as to store the data and break the encryption (via brute force) over time. In this case, he may be better advised to keep the secured session intact and create a copy of each packet for his own purposes. This is much easier to accomplish since it does not require active control over the traffic flow.

As discussed earlier, traffic encryption does not help to prevent flood attacks either since the adversary is the legitimate end point of the encrypted traffic flow.

J. Attack on Mobility Session

While the presented security solution provides a mechanism to protect host-based mobility, the mobility session itself can become the victim of DoS attacks. An adversary can commit such a DoS attack by sending a forged BU Reply in response to a BU Request before the peer can respond. The sender of the BU Request will accept the attacker's over the peer's reply, which arrives at a later point in time. For this purpose, the attacker must know the key A, i.e. he must have observed all of the peer's previous messages.

Note that such an attack is harder to conduct than a DoS attack on the traffic connection itself, which can be done via steps (a) and (b). Further, a DoS attack on the mobility session only disables mobility but it does not disrupt the traffic connection. Therefore, an attacker having means (a) and (b) may be better advised to directly interrupt the traffic connections rather than taking down the mobility session and hoping that MN will move.

VI. Application to Existing Protocols

In this section, we show how our security solution can be applied to the existing mobility protocols MPTCP and MIPv6. MPTCP is currently under development by the IETF Multipath TCP workgroup. It allows parallel utilization of multiple transport paths to maximize the aggregate end-to-end throughput. MIPv6 is a well known

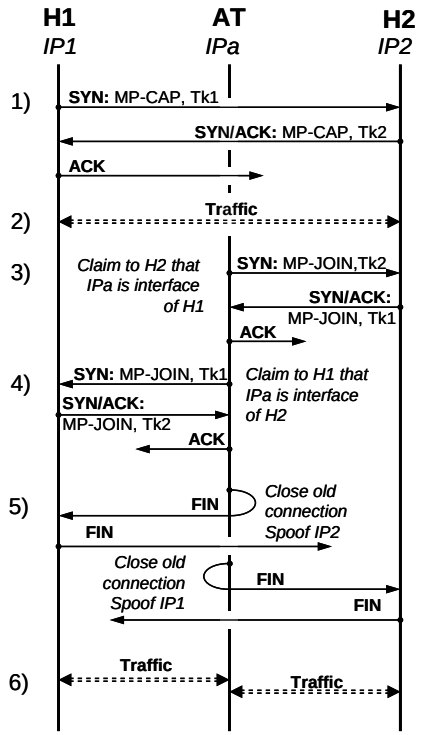

Figure 5. Session Hijacking using MPTCP

mobility protocol which incorporates host-based mobility features for the purpose of route optimization.

A. Securing MPTCP

MPTCP operates on transport layer and is restricted to TCP transport protocol. From the application-layer's perspective, each MPTCP session appears as one stream-oriented transport socket. Within the transport layer, MPTCP can support multiple TCP connections along different paths between host and peer. Each TCP connection is referred to as a subflow and supported by its own flow engine. On top of all subflows resides one aggregate flow engine that guarantees in-order delivery of data pertaining to the subflows. Obviously, both end hosts have to support the MPTCP protocol.

While data scheduling along multiple subflows and congestion control are unique to MPTCP, setup and tear down of subflows follow the same principles as for all host-based multi-homing protocols. From the security perspective, MPTCP can therefore be treated along the same lines.

MPTCP currently protects mobility messages using random numbers referred to as tokens. The tokens are chosen at the beginning of the MPTCP session and also serve as end point identifier across all subflows.

On hand of MPTCP, we want to demonstrate on how a passive on-path adversary can hijack a session using mobility messaging that is only token-secured. In this scenario, the adversary smoothly inserts himself into the session and ends up sustaining two independent sessions, one with each host. The steps are the following (Fig. 5):

1. *The two hosts, H1 and H2, start a TCP session using SYN-ACK handshake, mutually announce their support for MPTCP (MP_CAPABLE) and exchange respective tokens Tk1 and Tk2. The on-path adversary AT follows the MPTCP session setup and learns the tokens as well as the mapping between data- and subflow sequence numbers of both hosts.*
2. *After session setup, the hosts exchange traffic.*
3. *The adversary starts a TCP session with H2 pretending to be H1 and he introduces his own IP address (IPa) as the endpoint for a new subflow of the MPTCP session. For this purpose, he inserts the MP_JOIN message as well as H2's token Tk2 into TCP SYN. H2 accepts since it recognizes Tk2.*
4. *The adversary undergoes the same procedure with H1 using Tk1. This step can be done in parallel to step 3.*
5. *The adversary terminates the existing subflow between H1 and H2 by sending a TCP FIN message to each host spoofing the corresponding peer's IP address.*
6. *Since the direct TCP session between H1 and H2 has been discontinued, H1 and H2 continue communicating on the respective TCP connections with the adversary.*

MPTCP does not support an explicit routability test. According to the workgroup [31], a flood attack is unlikely since the victim does not respond with ACKs to the packets of the flood, which sooner or later terminates the session. It is even more likely, that the victim sends a TCP RST when receiving packets from an unknown host.

Our security solution can be applied to MPTCP to mitigate its vulnerability to on-path attacks. For this

TABLE I.
MESSAGE MAPPING FOR MPTCP

MPTCP	Security Solution	Direction of flow
SESSION INITIALIZATION		
MP-CAPABLE SYN	INIT REQ	Host1 → Host2
MP-CAPABLE SYN-ACK	INIT RSP	Host1 ← Host 2
FORWARD JOIN		
MP_JOIN SYN	BU REQ	New subflow: MN→CN
MP_JOIN SYN-ACK	BU RSP	New subflow: MN←CN
REVERSE JOIN		
ADD_ADDRESS	BU REQ	Old subflow: MN→CN
MP_JOIN SYN	BT REQ	New subflow: MN←CN
MP_JOIN SYN-ACK	BT RSP	New subflow: MN→CN
(no message) ACK	BU RSP	New subflow: MN←CN

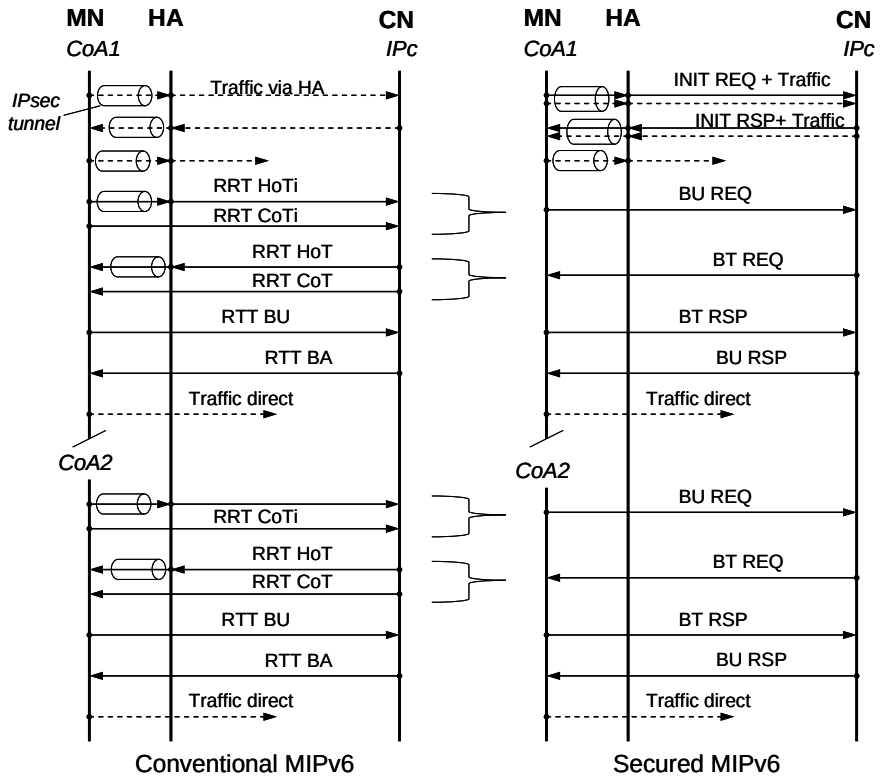

Figure 6. Current RRT procedure vs. proposed security solution for Molile IPv6.

purpose, the INIT exchange has to be mapped onto the MP_CAPABLE exchange as shown in Table I.

The mapping of BU/BT-handshakes to MPTCP's mobility signaling depends on the specific manner of subflow establishment. In one scenario, the MN starts a TCP connection from its new IP address to CN and encloses the MP_JOIN message into the SYN packet. In response, CN encloses the MP_JOIN message into the SYN/ACK packet. For this scenario, the BU REQ maps onto MN's MP_JOIN and the BU RSP to CN's response (Table. I). Note that this mapping does not contain the routability test.

MPTCP also permits subflow establishment in reverse direction. For this purpose, the MN sends its new IP address in an ADD_ADDRESS message on an existing subflow to CN. Upon reception, CN initiates a TCP connection to MN's new IP address inserting MP_JOIN onto SYN. MN responds with MP_JOIN inserted onto SYN/ACK. This procedure permits subflow establishment when CN resides behind a firewall or NAT. The corresponding mapping to our security solution is also shown in Table I.

Based on these mapping scenarios, MPTCP has to insert the corresponding security parameters (R, Hash(R), A, X and HMAC) into the mobility messages. The presently used tokens can serve as *MID*s even though its usage needs to be changed slightly. Since signaling is multiplexed with data traffic, we do not see a problem in linking mobility- and traffic sessions.

Unfortunately, such an upgrade is not straightforward since MPTCP uses TCP Options headers for signaling whose length is limited to 40B. Assuming that each security parameter (R, A, Hash(R), HMAC and X) cover 8B (= 64 bits), the BU REQ and BT REQ would use respective 24B and 32B. This is too large given that MPTCP- and TCP-specific control data have to be carried in this space as well. To overcome this constraint, MPTCP can resort to other means of signaling to hold the security-related parameters (in-payload or out-of-band).

B. Securing Mobile IPv6

While MIPv6 requires a global anchor referred to as home agent (HA), it also supports route-optimization (R/O), which allows traffic to be exchanged directly between hosts. In this case, MIPv6 still involves the HA for the mobility-related signaling handshake. This handshake applies methods of weak authentication, i.e. exchange of nonces and tokens as well as routability tests. While the handshake is rather complex, its security can be broken through eavesdropping and IP spoofing without the need for active control over the traffic flow. The attacker can further be transient since no security-related state is held on the end hosts to authenticate future mobility events.

It would be desirable to raise the security of MIPv6 to the level presented by our work. Two solutions, [5] and [32], have been proposed, which both use CGA. While these solutions meet our security requirements, they inflict a processing load at the end hosts, which is even larger than for the DH key exchange. CGA is further not compliant with managed configuration. Our security

TABLE II.
MESSAGE MAPPING FOR MOBILE IPV6

MIPv6	Security Solution	Direction of flow
SESSION INITIALIZATION		
(no message) (first traffic packet)	INIT REQ	Host 1 → Host2 Direct path or via HA
(no message) (first traffic packet)	INIT RSP	Host1 ← Host 2 Direct path or via HA
ROUTE OPTIMIZATION		
HoTi	BU REQ	MN → CN Direct path or via HA
CoTi		
HoT	BT REQ	MN ← CN Direct path
CoT		
RTT BU	BT RSP	MN → CN Direct path
RTT BA	BU RSP	MN ← CN Direct path or via HA

solution, in contrast, provides sufficient protection without these limitations.

In order to apply our solution, the concept of a mobility session between both hosts has to be introduced. One may be tempted to associate the route optimization state with the mobility session. This, however, leads to security problems since MIPv6 requires setup of transport connections *before* rather than *after* engaging into route optimization.

When the traffic session is started while MN resides in its home network, the HA is bypassed. In this case, the INIT handshake follows along the same lines as for host-based mobility protocols discussed above. When the traffic session is started while MN roams, the INIT exchange must run through the HA. After the handshake has succeeded, MN can engage into R/O. In case it fails, MN has to assume that CN does not support the mobility protocol and uses the HA as a permanent anchor.

Since MIPv6 sends signaling messages on IP Extension Option headers, they can be multiplexed onto traffic data. The INIT exchange can therefore be conducted as part of the initial traffic packet exchange for UDP transport or the SYN/ACK connection setup procedure for TCP transport. This allows session establishment without invoking additional delay (Fig.6). Multiplexing signaling and traffic has advantages as discussed before.

When MIPv6 is supported by both hosts, R/O is enforced through the MIPv6 Return-Routability Test (RRT), which has the same security function as the combined BU/BT handshake. While both RRT and BU/BT are 4-way handshakes, the RRT demands some messages to be sent along two paths, i.e. the direct path and the path via HA, each holding different security parameters. The RTT therefore requires six messages rather than four.

Since the two-path signaling procedure serves security purposes, it can be simplified when applying our stronger security solution. The resulting mapping of messages is shown in Table II and in Fig. 6:

- The Home-Test-Init- and Care-of-Test-Init messages are replaced by the BU REQ, which can be sent along either of the two paths.
- The Home-Test and Care-of-Test replies are replaced by the BT REQ, which must be sent along the direct path.
- The "RTT Binding Update (BU)" message of MIPv6 is replaced by the BT RSP and sent along the direct path.
- The "RTT Binding Acknowledgement (BA)" message of MIPv6 is replaced by the BU RSP, which can be sent along either path.

While the mobility-related content of the messages stays the same, the security-related parameters, such as nonces, tokens and RTT HMACs, have to be replaced by the corresponding parameters (R, A, X etc.) of the presented security solution

Apart from improving protection, the security-related design modification changes the overall appearance of MIPv6: MN now appears to CN as a multi-homed host that simultaneously supports two addresses, the Home Address (HoA) and the Care-of Address (CoA). While the mobility session is established via MN's primary IP address (HoA), the BU/BT handshakes introduces MN's secondary IP address (CoA) to CN.

The MIPv6 modification also adds functionality: Since the BU handshake can run along either path, i.e. the direct path or the path via HA, the HA can serve as a relay in case the CN resides behind a firewall. The BT REQ sent by CN consequently hole-punches the firewall so that the BT RSP and MN's traffic can pass. This scenario therefore offers the same functionality as proposed by MPTCP in the reverse path-establishment feature.

While conventional MIPv6 solely permits route optimization, the modified design can also operate as a true host-based mobility protocol, i.e. without HA-support for roaming mobiles. The HA can be flexibly inserted to support legacy hosts. MNs residing at their home network exercise INIT at session start and they know right away if they need to involve the HA in case they wish to roam. Roaming MNs, on the other hand, can establish sessions via HA to have a fallback anchor in case CN does not support the mobility protocol. The modified MIPv6 therefore supports a smooth migration path from network- to host-based mobility.

VII. CONCLUSION

We presented a security solution for host-based mobility and multi-homing protocols, which combines methods of temporal separation, routability tests and secret/answer chains. We have shown that such methods, if designed properly, can sufficiently protect mobility-related signaling messages while keeping memory and processing requirements at the end nodes very low.

The presented security solution can be applied to

existing or future mobility and multi-homing protocols as we demonstrated on hand of Mobile IPv6 and MPTCP. The proposed modification of Mobile IPv6 has exhibited a migration path from network- to host-based mobility. The combined effort of balanced signaling security and incremental deployment should lead to a more scalable and cost-efficient design of mobility networks.

REFERENCES

[1] 3GPP, "3GPP system to Wireless Local Area Network (WLAN) Interworking, System Description (Release 6)", TS 23.234, v.1.14.0, 2010.

[2] R. Moskowitz & P. Nikander, "Host Identity Protocol (HIP) Architecture", RFC 4423, IETF, May 2006.

[3] D. Johnson, C. Perkins, J. Arkko, "Mobile Support in IPv6", *RFC 3775*, IETF, June 2004.

[4] D. Funato, K. Yasuda, and H. Tokuda. "TCP-R: TCP mobility support for continuous operation", *Proc. IEEE ICNP'97*, pp. 229-236, 1997.

[5] Guo, C., Wu, H., Zhang, Q., Song, J., Zhou, J., Huitema, C. and Zhu, W., "End-system-based mobility support in IPv6", *IEEE J. Sel. Areas Commun*. v23 i11, pp. 2104-2117, 2005.

[6] Le, D., Fu, X., Hogrefe, D. (2006), "A review of mobility support paradigms for the Internet", *IEEE Comm. Surveys and Tutorials*, 1st Quarter, Vol. 8 No.1, pp.38-51, 2005.

[7] R. Stewart et al., "Stream Control Transmission Protocol", RFC 2960, IETF, 2000.

[8] A. Matsumoto, M. Kozuka, K. Fujikawa, Y. Okabe, "TCP Multi-Home Options", draft-arifumi-tcp-mh-00, IETF, 2003.

[9] E. Nordmark, M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", RFC 5533, IETF, 2009.

[10] A. Ford, C. Raiciu, M. Handley, J. Iyengar, "Architectural Guidelines for Multipath TCP Development", draft-ietf-mptcp-architecture-02, IETF, 2010.

[11] J. Arkko and P. Nikander, "Weak Authentication: How to Authenticate Unknown Principals without Trusted Parties", *Proc. of Security Protocols Workshop*, Cambridge, UK, pp. 5-19, 2002.

[12] P. Nikander, J. Arkko, T. Aura, G. Montenegro, E. Nordmark, "Mobile IP Version 6 Route Optimization Security Design Background", RFC 4225, IETF, 2005

[13] 3GPP, "Local IP Access and Selected IP Traffic Offload (Release 10)", TS 23.829, v1.0.1, 2010

[14] B. Harris, and R. Hunt, "TCP/IP security threats and attack methods", *Computer Communications*, vol 22/10, pp. 885 – 897, 1999

[15] E. Nordmark, T. Li, "Threats Relating to IPv6 Multihoming Solutions", RFC 4218, IETF, 2005.

[16] A. Slagell, R. Bonilla, W. Yurcik, "A survey of PKI components and scalability issues", *25th IPCCC*, pp.484 – 493, 2006.

[17] T. Aura, "Cryptographically Generated Addresses (CGA)", RFC 3972, IETF, 2005.

[18] M. Bagnulo, "Hash-Based Addresses (HBA)", RFC 5535, IETF, 2009.

[19] A. C. Snoeren, H. Balakrishnan, "TCP Connection Migration", draft-snoeren-tcp-migrate-00, IETF, 2000.

[20] D. Crocker, "Multiple Address Service for Transport (MAST): an Extended Proposal", draft-crocker-mastproposal-01, IETF, 2003

[21] V. Zandy, B. Miller, "Reliable network connections", *8th Int. Conf. Mob. Comp & Netwrk*., pp. 5 – 106, 2002

[22] F. Teraoka, M. Ishiyama, M. Kunishi, "LIN6: A Solution to Mobility and Multi-Homing in IPv6", draft-teraoka-ipng-lin6-02, IETF, 2003.

[23] A. Ford, C. Raiciu, M. Handley, "TCP Extensions for Multipath Operation with Multiple Addresses", draft-ietf-mptcp-multiaddressed-02, IETF, 2010

[24] H. Krawczyk, M. Bellare, R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, IETF, 1997

[25] G. Hampel, V. Kolesnikov, "Lightweight security solution for host-based mobility & multi-moming protocols", *Workshop on Seamless Mobility*, Globecom10, in print, 2010

[26] C. Vogt, "Credit-based authorization for HIP mobility with concurrent", draft-vogt-hip-credit-based-authorization-00, IETF, 2005.

[27] D. Eastlake, J. Schiller, S. Crocker, "Randomness Requirements for Security", RFC 4086, IETF, 2005.

[28] S. Kent, K. Seo, "Security architecture for the Internet Protocol", RFC 2401, IETF, 2005.

[29] T. Dierks, C. Allen, „The TLS protocol, version 1.0", RFC 2246, IETF, 1999.

[30] P. Eronen, "IKEv2 mobility and multihoming protocol (MOBIKE)", RFC 4555, IETF, 2006

[31] M. Bagnulo, "Threat analysis for multi-addressed/multi-path TCP", draft-ietf-mptcp-threat-03, IETF, 2010.

[32] J. Arkko, C. Vogt, W. Haddad, "Enhanced route optimization for Mobile IPv6", RFC 4866, IETF, 2007.

Georg K. Hampel is Researcher in the Networking & Networks Domain at Bell Labs/Alcatel-Lucent in New Jersey, USA. He received his M.S. in Physics in 1989 and his Ph.D. in Physics in 1994 from J.W. Goethe Universität in Frankfurt am Main (Germany). He has worked in a variety of disciplines such as network protocols, wireless communications, application development and physical sciences. His current research interests include Internet mobility, flat-hierarchy networks and application enablement.

Dr. Hampel has published over 20 papers in renowned journals and holds many patents in the area of networking and wireless communications. He has presented his works at conferences and universities. Dr. Hampel is Member of the IEEE.

Vladimir Kolesnikov is a Member of Technical Staff in Bell Labs' Enabling Computing Technologies domain in Murray Hill, NJ. He received his Ph.D. in Computer Science from the University of Toronto in 2006. His research interests include secure multiparty computation, key exchange, foundations of cryptography and network security. His work is connected to the practice of cryptography. He has worked on securing channels in Smart Grid and WiMAX, biometric authentication, and other subjects.

Dr. Kolesnikov published his work in top cryptographic and security conferences and journals. He has served on program committees of several international cryptography conferences. He is an editor of the WiMAX "Server Certificate Profile" and "Device Certificate Profile" standards documents.