A Proof of Security of a Mesh Security Architecture

*Doug Kuhlman, *Ryan Moriarty, Tony Braskich, Steve Emeott, Mahesh Tripunitara
Motorola Labs
1301 E. Algonquin Rd.
Schaumburg, IL 60196
{doug.kuhlman, ryan.moriarty, tony.braskich, steve.emeott, tripunit}@motorola.com

Abstract

The IEEE 802.11s standard is tasked to provide ways of establishing and securing a wireless mesh network. One proposal establishes a Mesh Security Architecture (MSA), with an interesting key hierarchy and full protocol definitions. This paper proves the correctness and security of the MSA proposal and its corresponding protocols. We also propose and prove the security of an additional protocol (an abbreviated handshake) which offers a substantial efficiency improvement in certain instances. To prove the entire architecture secure, we utilize Protocol Composition Logic (PCL) to prove each protocol secure. From that basis, we can show the protocols compose securely to prove the entire architecture. We also contribute some novel concepts to PCL, to allow us to prove the security of the overall architecture.

1 Introduction

Security is an important concern for many networks, particularly wireless ones, where attacks are easier to mount, because the network itself is so easy to detect and to use. Wireless protocols, too, have been successfully attacked. The most famous of these attacks are against the Wired Equivalent Privacy (WEP) protocol [3, 24]. The IEEE task group *i* [1] was created to provide a more secure protocol, ratified in June 2004. While the protocol was initially created simply using good design criteria, it was later proven to be secure under certain assumptions [20].

The IEEE 802.11s task group was formed to define extensions to IEEE 802.11 supporting mesh networking [19]. A goal of the task group is to secure the mesh utilizing existing IEEE 802.11 security mechanisms and extensions. Instead of waiting until the protocols and key derivations are ratified and in use, we examine a particular proposal [4, 5, 6, 7]. This document describes not only one protocol chain, but a multitude of intermixing protocols, designed to establish, distribute, and use keys within a wireless mesh network. Since this submission has not yet been fully ratified and implemented, we are hoping that an early analysis will provide support for valuable portions, while also pointing out a few areas of suggested improvements, which we hope can be instantiated in the final release of the standard.

While many methods of proving the security of a protocol exist, we chose to utilize the Protocol Composition Logic (PCL) methodology introduced by Durgin, Mitchell, and Pavlovic [17] and later improved on by many others. While we acknowledge the value of BAN logic, computational methods, and a host of other protocol proof methodologies, we feel that the use of PCL provides the appropriate level of abstraction, while providing the absolutely critical composition properties which allow us to extend proofs of security from individual protocols to the entire Mesh Security Architecture (MSA) proposed to the IEEE 802.11s standard. Additionally, the IEEE 802.11s proposal draws some, in spirit, at least, from the IEEE 802.11i standard. Since PCL was the choice for proof system for IEEE 802.11i in [20], we found it natural to continue its use in the security proof of MSA. The MSA proposal encompasses more protocols than does the 802.11i standard, including some with interesting timing mechanisms. Nodes playing more than one role in a mesh also invalidates much of the existing proof structure from [20].

In this paper, we provide PCL equivalents of the protocols presented in the MSA submissions. In a break from previous papers, which have limited their examination to only the portions of the protocol critical for proving matching conversations, we examine the totality of each protocol. We have found that extending protocols from basic primitives to fuller information exchange to be a nontrivial problem. Instead of leaving that as an exercise for the reader or the implementer, we have broken down the protocols into smaller por-

*principal authors

tions, proving the authentic delivery of additional information within each protocol. This was hinted at in [2], and we follow their example in not formally defining authentic delivery of additional information. The natural intuition that the information sent from one party is received at the other party exactly as sent will suffice for our purposes.

Our analysis of the protocols and key hierarchy of this particular IEEE 802.11s submission indicate that it was well-designed. We have only a few recommendations to make.

- The initial authentication message from the mesh authenticator to the requesting mesh point should include a nonce from the mesh authenticator and not simply use the key generation nonce provided by the mesh key distributor. This provides freshness guarantees for all parties appropriately.

- In the Group Key Handshake, we add the MAC addresses of the sender and the receiver inside the MIC. This is necessary to prove authenticity of the update message and to prevent a specific type of reflection/replay attack.

We elaborate on all of these points later in the paper.

The final result is that we provide a proof of all of the protocols used in MSA, with some of the listed modifications. We prove that all messages are directly tied to the sending and receiving nodes and must be delivered exactly as sent, if the protocol completes. Furthermore, we show that all of the protocols compose together arbitrarily. That is, the various protocols can be run concurrently, in sequence, or other ways, as required by higher-level logic. This provides a complete proof of the security of the MSA key architecture. We use standard assumptions from other PCL papers, with only a few additions.

The rest of the paper is organized as follows. Section 2 provides a brief background on PCL and the IEEE 802.11s standard. Section 3 contributes some minor but critical additions to the PCL taxonomy. Section 4 describes the MSA key hierarchy, upon which much of the security relies. Section 5 examines the proposed protocols of MSA. Section 6 suggests a new abbreviated handshake protocol, for quicker mobility within MSA. Section 7 provides a proof of the security and robustness of the key hierarchy already introduced, by examining the protocols of the previous two sections. Section 8 proves the composition of all the protocols and gives the major results of the paper. Section 9 concludes the paper.

2 Preliminaries

We motivate this paper and provide some additional background on the notation, terms, and conventions of this paper.

2.1 Overview of Proof Method

We use Protocol Composition Logic (PCL) to prove correctness and security of the Mesh Security Architecture. We briefly overview PCL in this section. PCL has been used for a security analysis of 802.11i [20] and IPv6 [14].

2.1.1 Cords

Protocols in PCL are modeled using *cords* and cord calculus. This provides a compact way of describing protocols, while simultaneously giving a formal logic definition. A typical two-person protocol will be modeled by two *threads*, one thread for the initiator and one thread for the responder, making a single cord. Possible actions inside a thread include nonce generation, signature creation, encryption, hash calculation, network communication, and pattern matching (which includes decryption and signature verification). Each thread consists of a number of basic sequences, each of which has pre and post conditions.

2.1.2 Proof Methodology

The proof methodology of PCL is described in [11, 12, 20, 10, 13, 17, 16, 9, 23]. We use the standard syntax of $\Theta[P]_X\Phi$. This means that with preconditions Θ before the run of actions P by thread X, the result (postcondition) Φ is proven to hold. The conditions of Θ and Φ usually indicate what actions a thread has already done, what information is available to certain parties, temporal order of corresponding actions that must have happened, or the like. These are useful for stating the states before and after a protocol run, from each participant's perspective. We use the notation that Θ is always a precondition, Φ always a postcondition, and Γ an invariant. Some postconditions for one step will become preconditions for another step. We consistently use a subscript of SI for security invariants. Other subscripts are ordered (protocol, description, principal) where some fields may be blank. We also use shorthand notation for various preconditions, postconditions, and/or invariants. These are denoted by { and } in the subscript, to indicate the conjunction of two (or more) conditions. For example, $\Gamma_{SI,\{ABBH,SIMO\},\{1,2\}}$ would be used to denote $\Gamma_{SI,ABBH,1}\wedge\ \Gamma_{SI,ABBH,2}\wedge\ \Gamma_{SI,SIMO,1}\wedge\ \Gamma_{SI,SIMO,2}$.

The proof system is built on three fundamental building blocks. The first is a series of first-order logical axioms, described in [13]. The second is a series of cryptographic/security axioms. These are described in various papers, including [13, 20, 17]. These assume reasonably idealized cryptographic functionality, but most cryptographic primitives achieve them in practice, if not necessarily in theory. The third building block is the fundamental principle of *honesty.* Honesty imposes certain restrictions on roles. Honesty is a special type of invariance that allows one instance of a thread to reason about the actions of a corresponding thread, participating in the same protocol. Honest parties follow defined protocols in predictable ways. If the other party in a protocol is not honest, then nothing can be proven/reasoned, because the other party could have already sent out private key information. The actions of the attacker are not assumed to be honest, of course. We do, however, assume that the attacker is not a "legitimate" node already in the network.

The axioms used in this paper are described in the literature [9, 13, 20]. These describe the basic first-order actions, honesty principle and most of the security axioms needed. One missing security axiom, which we need to reason about the correctness of information at certain parties, is that a node which creates a signature of information must have that information. This is axiom **SIG1**: $\text{Computes}(X, SIG_{priv_X}(m)) \supset \text{Has}(X, m) \wedge \text{Has}(X, priv_X)$.

2.1.3 Composing Proofs

One of the core features of PCL is its ability to reason about how certain protocols interact. Since this paper will be covering an entire architecture, it is imperative that the large number of individual protocols be proven secure not only independently but working together in conjunction in the system. To this end, we extensively use the proofs of protocol composition developed in [13]. We will expand on this concept in Section 8.

2.2 Overview of 802.11s

The overall IEEE 802.11 standard is charged with providing a Wireless Local Area Network (WLAN) with an infrastructure typically connected using wired Local Area Networks (LANs). This fixed network is at odds with recent trends for applications that require a mobile infrastructure along with mobile end nodes. Additionally, there is a need for a network that provides increased bandwidth while using infrastructure nodes that individually offer a smaller communication range than is typical today.

To meet these needs, the 802.11 task group *s* ("Mesh Networking") is working to develop a mesh networking protocol, providing auto-configuring, multi-hop paths between wireless stations to support the exchange of data packets. Mesh networking may be used in a variety of usage scenarios to extend wireless coverage with minimal additional configuration and to locations previously impractical to reach. A goal of the task group is to utilize existing IEEE 802.11 security mechanisms, with extensions, to secure a Mesh in which all of the stations are controlled by a single logical administrative entity for security [19]. The 802.11 Mesh Networking task group continues to refine its draft specification through the resolution of comments received during a review of the specification in late 2006 [4, 7, 5, 6].

A mesh network of nodes, as defined by the 802.11s submissions [6, 7], has a few major types of nodes. A mesh can be identified with its *Mesh Key Distributor* (MKD). The MKD is responsible for much of the key management within its domain. No mesh can form without one node being designated as the MKD. For meshes which require authentication at an AAA server (e.g., a RADIUS server), the MKD is assumed to have a secure physical link with the AAA server. Because of the special nature of the MKD, we assign the MKD the "variable" T in this paper. All references in this paper to T are exclusive to the MKD. The MKD is also a regular member of the mesh and, as such, all normal nodes (typically denoted X or Y) include the MKD as well. A general node in the mesh will be either a *Mesh Point* (MP) or a *Mesh Authenticator* (MA). A MP is a full member of the mesh and can communicate with other nodes in the mesh. A MA is a MP which has established a session key with the MKD. A *Candidate MP* is an entity which wishes to join the mesh but is not yet a MP.

The protocols in MSA provide additional complications beyond the main protocols used in 802.11i in a few ways. The primary difference is the peer-to-peer nature of a mesh network. Nodes in a MSA mesh must be allowed to act in different roles at different times. Thus, the invariants used to prove the security of the 802.11i 4-way handshake, which rely on a node not sending certain messages, no longer apply. The new invariants introduced are slightly more complicated but serve to accomplish a similar goal (preventing reflection attacks). The peer-to-peer nature also poses some difficulties with timing. The 802.11i proofs used matching conversations to prove authenticity. However, in MSA, we must provide for the case that both parties simultaneously start instances of a protocol and messages are not necessarily well-ordered. Thus, the proofs from [20] do not carry over directly.

We also abuse key notation slightly. Most keys we will discuss are actually made of a plurality of keys, used for unique purposes. For example, the key material we denote as *ptk* actually has three parts, a transport key (*tk*) used for bulk encryption, a key encryption key (*kek*) used to encrypt the *gtk*, and a key confirmation key (*kck*) used for Message Integrity Code (MIC) generation. We simplify the exposition by utilizing *ptk* for all these parts.

Similarly, the MSA proposal describes using particular bits to create unique message IDs. We do not recreate that work here. For simplicity (and since we don't have bandwidth concerns), we simply use strings to denote the unique message IDs.

3 Additions to PCL and Proof Methodology

In order for our proofs and protocol descriptions to meet our required level of adequacy we need to contribute some ideas to PCL and introduce some new proof goals. In this section we elaborate on these ideas.

3.1 Flexible Temporal Ordering

The temporal ordering of actions in the original PCL definition is too strict for our applications. Protocols we wished to analyze using PCL include a simultaneous protocol where the order in which some of the messages are sent and received does not have to be strict. For example in the simultaneous open case of the abbreviated handshake, the final two messages of the protocol may be sent and received in either order. Currently in PCL one must decide on a strict ordering, thus we were not able to describe this protocol in PCL.

PCL needed to be updated to allow this particular application. The necessary change to PCL can be realized as a simple add-on to the language. In other words, the proposed modification does not fundamentally change PCL, it only adds capability to the language. Thus all previous proofs under PCL will still hold true with our addition to PCL.

Our change to the language is adding an "action group" and redefining a strand. We define an action group as: (action group) $g ::= (a; \ldots ; a)$, where a is an action as defined in [13]. We also redefine a strand as: (strand) $s ::= [g(;\text{or }:) \ldots (;\text{or }:)g]$. Thus a strand is now composed of an arbitrary number of action groups separated by colons or semicolons. The idea behind the action group is that the actions in an action group must be done in the order they appear. However, the action groups within a strand separated by a colon (:) can be done in any order and action groups separated by a semicolon (;) must be done in the order they appear. Note that any strand defined previous to this addition to the language can still be defined exactly the same way by defining each action group to be one action and by setting all the separators inside a strand to semicolon.

We now update Axiom **AA4** to reflect this addition to the language. Recall [13] that **AA4** is: $\top[a; \ldots ; b]_X a < b$. We now include action groups and the new strand in this axiom. Thus we redefine **AA4** as: $\top[a : b; \ldots ; c : d]_X a \wedge b < c \wedge d$, where a, b, c and d are action groups. Note that nothing about the temporal order of a compared to b or c compared to d is indicated. We also include a new axiom **AA5** as $\top[(a; \ldots ; b)]_X a < b$, where a and b are actions, to deal with the temporal ordering of action groups. Also note that if each action group is exactly one action and only semicolons are used in the new strands our **AA4** becomes exactly the **AA4** previously defined and **AA5** is in this case irrelevant.

Protocols whose definition includes a colon add an additional complication in the determination of basic sequences. Since a basic sequence is defined as any actions before a receive, there may be different sets of actions that occur before a receive depending on the sequence of events in reality. Thus we must ensure that invariants and preconditions hold over all possible basic sequence orderings and compositions.

3.2 Generalized Matching Conversations and Generalized Mutual Authentication

The proofs of mutual authentication used in many previous PCL papers have been based on the standard notion of authentication called matching conversations [2]. This is natural as these protocols are "turn-by-turn protocols" in which one a participant receives a message and then responds to the message, then receives a message and responds to it and so on. However, some of the peer-to-peer protocols in this paper can never obtain matching conversations as the order in which messages are sent and received is necessarily flexible, as a functional requirement. Thus we generalize the key properties of matching conversations and informally define two new notions which we call maximal conversations and generalized matching conversations. We feel these definitions will have significant impacts beyond the scope of this paper.

We loosely define conversation and then matching conversation from [2]. A conversation is the set of ordered triples of time, received message, and sent messages for a party. It describes the visible network actions of a single party. A matching conversation is where one party's conversation exactly matches another's party conversation, in reverse (where one party's sent messages match the other's received messages and vice versa). Additionally, the temporal order of the sends and receives must follow what one would consider natural for a turn-based protocol.

We loosely define the *maximal conversation* for a participant A. We first determine the maximal possible temporal ordering. To do this we consider all legal orderings in an ideal world (one with no adversarial interference) from the view of a participant A in a protocol. Given this maximal temporal ordering, we note the existence of messages for which A can never confirm reception, because A could not confirm the reception of the message in an ideal world with no adversarial interference. This is analogous to the last send in a turn-by-turn protocol, for which the sender cannot possibly verify reception. We take the maximal temporal ordering and remove any send or receive that A could not verify in the ideal world – the remaining actions represent the maximal conversation for participant A.

With this definition in hand, we now define *generalized matching conversations* for a participant A. We say A has generalized matching conversations, if in every run of the system, every action in the maximal conversation for participant A has a corresponding action at participant A (e.g., A does all its actions) and at the appropriate other participant in the system. For two-participant protocols (like all those in this paper), this means that the maximal conversation for participant A has messages exactly matching the other participant's maximal conversation, with the strictest time ordering possible.

We can now informally define *generalized mutual authentication*. In the world where an adversary has access to every message and can act on them within the restraints of the proof system (symbolic or computational), generalized mutual authentication means that generalized matching conversations for every participant implies acceptance and acceptance implies generalized matching conversations for every participant. For the purpose of this paper we wish to keep the definition of generalized mutual authentication general. We explore all these definitions is detail in separate work.

When our definition is applied to a "turn-by-turn" protocol it becomes exactly the definition from [2]. In every other instance our definition requires that the ordering of actions be maximal with respect to what is possible in the ideal world. Since this definition imposes maximal temporal ordering on a protocol, this definition is at least sufficient. Most protocols in the MSA are turn-by-turn protocol and thus the [2] definition suffices. The three exceptions are the simultaneous open case of the abbreviated handshake (which is a peer-to-peer protocol and has some timing flexibility), Peer Link Establishment (which is not a cryptographic protocol in itself and requires no temporal ordering) and the Push protocol which is actually the composition of two protocols.

We note that the generalized matching conversations property encompasses the matching record of runs property [15] too. Additionally, this property guarantees all desired properties from [21] and implies all the possible authentication definitions in [18].

3.3 Security Invariants

In PCL security goals are generally shown to hold upon successful completion of a protocol. However some security goals must hold throughout the entire run of a protocol, even if the protocol aborts prematurely. The Insecure Key Transfer Protocol in figure 1 illustrates this point. If we assume protocol completion from the point of view of RESP we can prove that the secret key was distributed correctly, since the validity of INIT's public key is established once RESP receives the third message. However RESP uses the public key in the second message before the validity of the public key can be established. Thus if the protocol aborts after the RESP sends the second message, it may be the case that the public key sent in message one was actually an adversary's public key. It is therefore possible for the adversary to intercept the secret key. While this protocol is slightly contrived, in larger protocols with different security goals it may be the case that a subtle insecurity like this will go unnoticed. Thus for certain security goals we advocate showing they hold after every possible point at which the protocol may abort. We call these security goals security invariants.

Inputs and Parties:
Two parties: INIT and RESP. Shared input: confirmation key (ck). INIT private input: INIT public key (PK_{INIT}). RESP private input: secret key (sk).
Goal: Has$(Z, sk) \supset Z = \text{INIT} \vee Z = \text{RESP}$

Insecure Key Transfer Protocol:

1. INIT sends PK_{INIT} to RESP.
2. RESP receives PK_{INIT}; encrypts sk under PK_{INIT}, computes the keyed hash of the encryption with key ck; and sends $(\{sk\}_{PK_{INIT}}, HASH_{ck}(\{sk\}_{PK_{INIT}}))$ to INIT.
3. INIT receives $(\{sk\}_{PK_{INIT}}, HASH_{ck}(\{sk\}_{PK_{INIT}}))$, verifies the keyed hash; decrypts sk; computes the keyed hash of sk and PK_{INIT} with the ck and sends $HASH_{ck}(sk, PK_{INIT})$ to RESP.
4. RESP verifies the signature.

Figure 1: Insecure Key Transfer

Security invariants differ from both typical invariants and security goals, though they share some similarities with both. Typical invariants are statements that are shown to hold throughout the system being proved and then used to prove the security goals of the protocol. Security invariants are statements that hold security properties within themselves. They are different than security goals as we want them to hold throughout the entire run of the protocol and not simply upon a successful completion. We follow the logic of [13], requiring security invariants to provably hold after every basic sequence in a protocol. This is similar to work done in [23, 22], although we feel we make the concept more explicit.

A key distribution protocol has a very natural (and very necessary) security invariant. One would want to show after every basic sequence that no party other than the two distributing the key could have access to the key, $(\text{Has}(Z, sk) \supset Z = \text{INIT} \vee Z = \text{RESP})$. Note that in the Insecure Key Transfer Protocol, it would not be possible to prove this security invariant from the point of the view of the Responder after RESP sends the second message, but it would be possible to show immediately after the RESP receives the fourth message.

3.4 Return to Global

In examining the proof of the Group Key Handshake from [20] we found an ambiguity in the update of the counter. Since most of PCL uses static variables in its thread definition, it would seem the global variable is only updated at the conclusion of the protocol. If this is the case, an invariant is violated and thus the proof of the Group Key Handshake property is incorrect. Fundamentally, the invariant claims that "if a message was sent after another message then the earlier message has a lower counter value." This is not always the case as two messages can be sent before the sequence counter is updated on a global level. On the other hand, updating variables globally as soon as they are modified/used/created leads to much larger complications within the rest of the PCL framework. Since counters are only referenced in [20] to our knowledge, we are led to conclude the counter implications are a slight oversight.

The most "unobtrusive" solution to this problem is to simply add a global update action that instantly updates the value in the main thread that instantiated the current thread. Thus any role that is started after the end of this basic sequence will contain the updated global variable. It is important to note that in PCL basic sequences are autonomous steps and thus we are guaranteed that the value that is updated at the global level will not be used until after the basic sequence that updated it completes. No values will change in the local state, thus the local state will still have all bound values as required by PCL. We need a way to refer to these global variables, thus for any variable x, *globalx* represents the same variable on a global level. The *return* action within a thread describes this action of updating a global variable with a local value.

It is not completely clear that the strong postcondition we prove utilizing this additional functionality is strictly necessary, but it is straightforward to accomplish and builds upon the work in [20], so we felt it appropriate to include it here. We found this addition to be necessary to prove the invariants and postcondition on ordering of messages with counters.

3.5 Mid-Protocol Composition

For many of the protocols in MSA, the protocol may instantiate another protocol partway through its run. For example, in a key exchange protocol, if both parties who are trying to establish a session key have a shared key cached locally, then they do not need to run a protocol to retrieve it. However if that key is not cached locally, one of the parties must pause their current protocol and run a key pull protocol. To further complicate matters, this could potentially happen in the middle of a basic sequence.

This is new ground for PCL and we have devised a non-trivial proof (see section 8) that enables us to frame this complex action in PCL and develop sound proofs. While we will elaborate on this subject later in the paper, we give a very brief description here. Essentially, we define the inception of functions that may need to run a separate protocol to be breaks in basic sequences. Then before and after these function calls we define basic sequence pre and post conditions that must be satisfied for a successful completion of the protocols. The idea of basic sequence pre and post conditions were give in [20] to enable staged composition and remain standard in the language [13], although they have not been previously used in this way to enable mid-protocol composition.

3.6 Miscellaneous Information and Functions Used in MSA

Part of the MSA deals with exchanging authentic information. This information is used to determine basic network functions (e.g., bandwidth selections), security information (e.g., cipher suite selection), and to help establish the node. From this collection of information, certain pieces are simultaneously chosen from two sets of information, by various functions. We discuss our model of this information and these functions.

$INFO_X$. A principal will need to exchange information with another principal during the Peer Link Establishment, Abbreviated Handshake, Mesh Key Holder Security Handshake (MKHSH), and key holder transport protocols. This information is represented as $INFO_X$. Its contents differ for each protocol. For the Abbreviated Handshake and Peer Link Establishment protocols, its contents include identifiers of cached keys, supported cipher suites and authentication methods, an identifier of the security domain, whether the principal is a MA and whether it can communicate with the MKD. During Peer Link Establishment, additional information about the MKD's identity and protocols it supports are included. During the MKHSH, $INFO_X$ includes identifiers of the mesh network and the security domain, and a list of protocols the MKD supports.

Select(). Two principals X and Y must make simultaneous selections of link and protocol options from exchanged information $INFO_X$ and $INFO_Y$. This selection is represented as one function, select(). During Peer Link Establishment and Abbreviated Handshake protocols, select() determines the key to be used based on information each principal sends about the keys it has cached and whether it is a MA and capable of retrieving the key from the MKD. Thus, the function ensures that a key is either locally cached or may be retrieved from the MKD if the protocol is to continue. Similarly, in the Peer Link Establishment protocol, select() determines the principal that will initiate the 4-way handshake. A pairwise cipher suite for use after the completion of the Peer Link Establishment and Abbreviated Handshake protocols also must be selected. The select() function added to PCL determines the cipher suite to be used based on $INFO_X$ and $INFO_Y$, selecting the most-preferred common cipher suite of the principal with the numerically-larger address.

Retrieve(). The retrieve function actually gets the key to the strand, after the selection of which key will be used. Retrieve takes a key name (*pmkN*, corresponding to a specific *pmk*) as its input. Then the retrieve function will look to see if the key is locally cached on disk. If it is not and the principal executing the retrieve function is a MA (has a connection set up with the MKD), the retrieve function will initiate a key pull. If the key is not on disk and the principal is not a MA, retrieve will fail and the protocol that called it will abort. Because the retrieve function may or may not perform a key pull, we create a break in the basic sequence directly before and directly after the retrieve function. Retrieve is an addition to the PCL action set.

The retrieve function has inherent preconditions and postconditions as it is a basic sequence. The preconditions and postconditions are intuitive given the explanation above. As a precondition the retrieve function must have the key looked for cached locally ($\text{Has}(X, pmk)$) or it must have a key that implies a connection with the MKD ($\text{Has}(X, mptk_{X,T})$). Thus the precondition is $\text{Has}(X, mptk_{X,T}) \vee \text{Has}(X, pmk)$ where *pmk* matches the input *pmkN*. The postcondition is simply $\text{Has}(X, pmk)$. The retrieve function has security requirements only if the principal must perform a key pull. We deal with the security requirements in the Pull protocol in Subsection 5.4.

Additional Functions. Additionally throughout the paper we use the Increment(x) function, which returns a number higher than x, IsLess(x, y), to check if $x < y$, IsLessEqual(x, y) to check if $x \leq y$, and return(x) which we described in detail in Section 3.4.

4 Key Hierarchy of MSA

The key hierarchy of MSA is critical for understanding how the various protocols interact and for determining what keys guard which other keys[6]. We describe the hierarchy here and prove its correctness in section 7. This delay will allow us to provide definitions of the protocols over which it is provably secure.

4.1 Description of the Key Hierarchy of MSA

Recall that the MSA mesh network is comprised of a Mesh Key Distributor (MKD), Mesh Authenticators (MA), Mesh Points (MP) and nodes trying to join the mesh we call Candidate MPs. The protocols describing entity interactions are given in the Mesh Security Architecture (MSA). The Mesh Security Architecture has an inherent key hierarchy. We give this key hierarchy in Figure 2. Each node in the graph represents a key and labels on each edge represents the protocols needed to obtain the key at the destination node from the key at the starting node. We note that the subscripts of the keys are ordered, for example the $pmk_{X,Y}$ is protected by the $pmkmkd_X$ and the $pmk_{Y,X}$ is protected by $mptk_{X,T}$. The entities in the system that may have access to the keys are listed next to each key, with the exception of the *gtk*.

Let X be a Candidate MP and let T be the MKD. X either has a shared xxKey with T or it shares public key credentials with T. If it shares public credentials with the MKD, then the MKD and X run the Transport Layer Security (TLS) protocol in order to derive the $xxKey_X$. Once T and X share an $xxKey_X$ they need

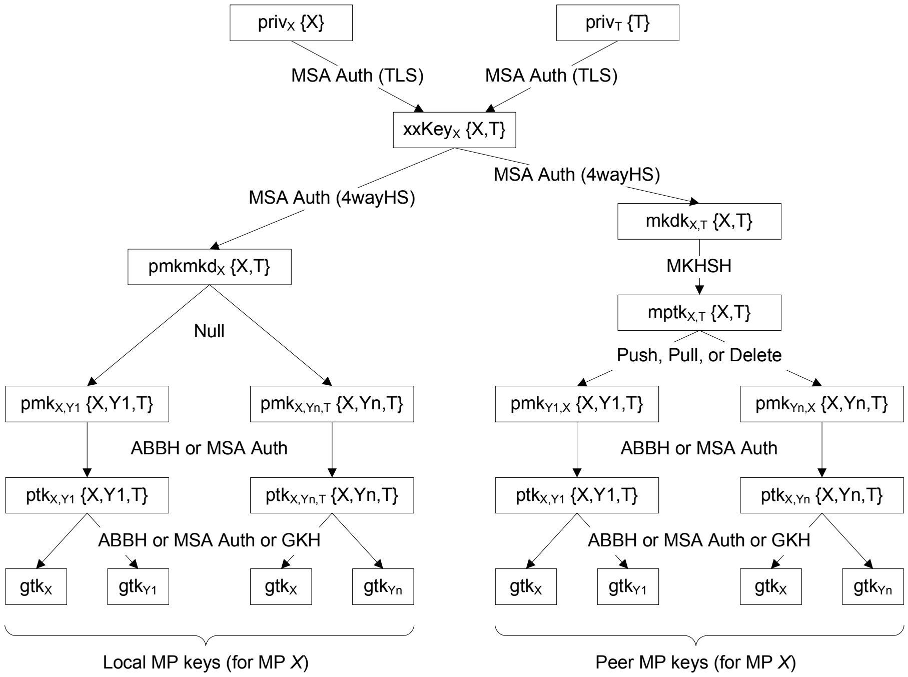

Figure 2: MSA Key Hierarchy for Node X

to share a nonce to derive the $pmkmkd_X$. This nonce is currently delivered to X when X runs a Four-Way Handshake with a MA. This Four-Way Handshake is part of the MSA Authentication. Once this nonce is delivered, X can derive the $pmkmkd_X$, $pmk_{X,Z}$ for any Z and the $mkdk_{X,T}$. When the Four-Way Handshake completes with node Y, X will have derived $ptk_{X,Y}$, received the gtk_Y from Y, and delivered gtk_X to Y.

At this point X becomes a Mesh Point (MP) but is not yet a MA. To become a MA, X needs to run the MKHSH with T, to derive the $mptk_{X,T}$, which is a session key between X and T. This will enable X to run the Push/Pull/Del protocols with T in order to retrieve $pmk_{Z,X}$ for any Z.

The 802.11s task group has received submissions expressing an interest in developing an Abbreviated Handshake (ABBH) [8, 25]. The ABBH is used by a MP or a MA (X) to derive a $ptk_{X,Z}$, and exchange gtk_X and gtk_Z with another MA or MP (Z). In [5], the method of exchanging these credentials is to have the MP or MA run the full MSA Authentication with the other MA or MP. In this paper we develop a candidate ABBH and prove its security and composability with the rest of the MSA architecture.

The only protocol not mentioned thus far is the Group Key Handshake protocol. This is used by a MA or MP to update its group key. The protocol only works with nodes with which it maintains a security association (shared ptk).

4.2 Proof of Key Hierarchy of MSA

In this paper we give a full proof of the key hierarchy. That is, we show that as long as the principals are honest and act according to the protocols, the ownership of the keys can be guaranteed according to the key hierarchy. We emphasize that our proof methods allow us to show this proof holds for ANY possible run of the MSA with honest principals. We believe this is the first proof in PCL of a full mesh key hierarchy.

We do this using the work of [23] and combining the ideas of that paper with our ideas of security invariants. We give the security invariants relevant to the key hierarchy in Figure 3. In Section 7 we give a proof that these security invariants do in fact hold throughout ANY run of the MSA, as long as the indicated principals are in fact honest.

5 Existing Protocols of MSA Proposal

In this section we give the description, invariants and goals of the existing protocols in the MSA proposal [7]. These descriptions are abstractions that contain the parts of the protocols relevant to the security proofs.

$\boldsymbol{\Gamma}_{\mathbf{TLS,SI,1}} :=$
$\text{KOHonest}(\mathit{priv}_X, \{\}) \supset$
$\text{Has}(Z, \mathit{priv}_X) \supset \hat{Z} = \hat{X}$

$\boldsymbol{\Gamma}_{\mathbf{TLS,SI,2}} :=$
$\text{KOHonest}(\mathit{xxKey}_X, \{\mathit{priv}_X, \mathit{priv}_T, \mathit{xxKey}_X\}) \supset$
$\text{Has}(Z, \mathit{xxKey}_X) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}$

$\boldsymbol{\Gamma}_{\mathbf{4WAY,SI,1}} :=$
$\text{KOHonest}(\mathit{pmkmkd}_X, \{\mathit{xxKey}_X\}) \supset$
$\text{Has}(Z, \mathit{pmkmkd}_X) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}$

$\boldsymbol{\Gamma}_{\mathbf{4WAY,SI,2}} :=$
$\text{KOHonest}(\mathit{mkdk}_{X,T}, \{\mathit{xxKey}_X\}) \supset$
$\text{Has}(Z, \mathit{mkdk}_{X,T}) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}$

$\boldsymbol{\Gamma}_{\mathbf{PPD,SI,1}}, \boldsymbol{\Gamma}_{\mathbf{MKHSH,SI,1}} :=$
$\text{KOHonest}(\mathit{mptk}_{X,T}, \{\mathit{mkdk}_{X,T}\}) \supset$
$\text{Has}(Z, \mathit{mptk}_{X,T}) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}$

$\boldsymbol{\Gamma}_{\mathbf{PPD,SI,2}} :=$
$\text{KOHonest}(\mathit{pmk}_{X,Y}, \{\mathit{pmkmkd}_X, \mathit{mptk}_{Y,T}\}) \supset$
$\text{Has}(Z, \mathit{pmk}_{X,Y}) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y} \vee \hat{Z} = \hat{T}$

$\boldsymbol{\Gamma}_{\mathbf{ABBH,SI,1}}, \boldsymbol{\Gamma}_{\mathbf{4WAY,SI,3}}, \boldsymbol{\Gamma}_{\mathbf{GKH,SI,1}} :=$
$\text{KOHonest}(\mathit{ptk}_{X,Y}, \{\mathit{pmk}_{X,Y}, \mathit{pmk}_{Y,X}\}) \supset$
$\text{Has}(Z, \mathit{ptk}_{X,Y}) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y} \vee \hat{Z} = \hat{T}$

$\boldsymbol{\Gamma}_{\mathbf{ABBH,SI,2}}, \boldsymbol{\Gamma}_{\mathbf{4WAY,SI,4}}, \boldsymbol{\Gamma}_{\mathbf{GKH,SI,2}} :=$
$\text{KOHonest}(\mathit{gtk}_X, \{\mathit{ptk}_{X,Y_1}, \ldots \mathit{ptk}_{X,Y_n}\}) \supset$
$\text{Has}(Z, \mathit{gtk}_X) \supset \text{Has}(Z, \mathit{ptk}_{X,Y_i})$

Figure 3: MSA Key Hierarchy

In some cases it was necessary to change these protocols slightly from their current description in MSA in order to complete the proofs of security; we always indicate when a change was made. In all cases, without these changes, the security goals were not true. It was not a deficiency in the ability of PCL to prove seemingly correct assertions but actual deficiencies in the protocols.

5.1 MSA Authentication

The MSA Authentication protocol can be run at different times throughout the MSA [5]. The MSA Authentication protocol can be used by a Candidate MP to join the mesh or by a MA or MP to establish a connection with a MP or MA so long as one of the parties is a MA. For clarity of presentation we will assume that X is a Candidate MP and Y is a MA for the length of this subsection.

The MSA Authentication protocol can be thought of as a single large protocol. It contains a Peer Link Establishment subprotocol, an optional TLS subprotocol, an optional Key Pull subprotocol and a Four-Way Handshake subprotocol. We examine each of the subprotocols separately. We then connect these subprotocols using their pre and post conditions. Below we discuss each of these parts separately.

5.1.1 Peer Link Establishment

The Peer Link Establishment is the first subprotocol to be run during MSA Authentication. During Peer Link Establishment X and Y will exchange various networking information. As we mentioned in Section 3.5, $INFO_X$ and $INFO_Y$ will represent this information. There is no cryptography of any kind in the Peer Link Establishment protocol; it is simply an unauthenticated exchange of information.

The Peer Link Establishment Protocol is a peer-to-peer protocol and because it was designed with this in mind it has some interesting properties. First the Peer Link Establishment phase of MSA Authentication is symmetric for each party. Also, X and Y do not need to send or receive the first message in any order. This is because it is valid for either party to initiate the protocol. For example if X is already a MP or a MA and is establishing a connection to Y who is already a MP or a MA, Y may have tried to start this establishment with X before it receives any message from X. In fact the parties may decide to start this protocol *at the same time.* One can verify that while the roles of X and Y do not necessarily have to be symmetric, symmetry gives us some desired properties.

As stated earlier, we will assume (without loss of generality) that X is the Candidate MP and Y is a MA. We give the strands, preconditions and postconditions of the Peer Link Establishment protocol in Figure 4. We note that the strands are symmetric thus we only present one strand, however because X is a Candidate MP and Y is a MA their preconditions and postconditions are different.

The only requirements we have of the Peer Link Establishment protocol is that upon completion each party involved in the protocol has received and sent the messages that should be sent in a Peer Link Establishment protocol and any party involved in the protocol must share either public credentials or a $xxKey_X$ with T. We can not hope to show our generalized mutual authentication goal as this is not a cryptographic protocol in itself. While these postconditions are trivial, we will use them as preconditions to later protocols. Formally we must show the following theorem. The case we are examining is the case where $\neg\text{Has}(X, \mathit{pmkmkd}_X)$ (so that X is not part of the mesh).

Theorem 1. *Given precondition Θ_{PLE} upon executing the Peer Link Establishment protocol $\Phi_{PLE,1}$ and*

$\mathbf{PLE} = (X, \hat{Y}, INFO_X)$
$[(\text{send } \hat{Y}, \hat{X}, INFO_X, \text{“IATH1”};$
$\text{rcve } \hat{X}, \hat{Y}, INFO_X, \text{“IATH3”}) :$
$(\text{rcve } \hat{X}, \hat{Y}, INFO_Y, \text{“IATH1”};$
$\text{send } \hat{Y}, \hat{X}, INFO_Y, \text{“IATH3”})]_X$

$\mathbf{\Theta_{PLE,INIT}} := (\text{Has}(X, cred_X) \wedge \text{Has}(T, cred_T)) \vee$
$(\text{Has}(X, xxKey_X) \wedge \text{Has}(T, xxKey_X)) \vee$
$(\text{Has}(X, pmkmkd_X))$

$\mathbf{\Theta_{PLE,MA}} := \text{Has}(Y, mptk_{Y,T})$

$\mathbf{\Phi_{PLE,INIT,1}} :=$
$\text{Send}(X, \text{IATH1X}) \wedge \text{Rcve}(X, \text{IATH1Y}) \wedge$
$\text{Send}(X, \text{IATH3X}) \wedge \text{Rcve}(X, \text{IATH3Y})$

$\mathbf{\Phi_{PLE,INIT,2}} := \mathbf{\Theta_{PLE,INIT}}$

$\mathbf{\Phi_{PLE,MA,1}} :=$
$\text{Send}(Y, \text{IATH1Y}) \wedge \text{Rcve}(Y, \text{IATH1X}) \wedge$
$\text{Send}(Y, \text{IATH3Y}) \wedge \text{Rcve}(Y, \text{IATH3X})$

$\mathbf{\Phi_{PLE,MA,2}} := \mathbf{\Theta_{PLE,MA}}$

Figure 4: Strand Preconditions and Postconditions of Peer Link Establishment

$\Phi_{PLE,2}$ *are guaranteed to hold. Formally,*

$$\Theta_{PLE}[PLE]_X \Phi_{PLE,1} \wedge \Phi_{PLE,2}$$

We give no proof as it is trivial. Note the lack of implication that the same messages were received by the peer. The information exchanged during the PLE is verified during the Four-Way Handshake. Thus upon completion of the Four-Way Handshake we prove that in fact the messages sent and received by one party are the same as the messages sent and received by the other party in the Peer Link Establishment. It is natural that we can not show this security property until after the completion of the Four-Way Handshake as the messages are not verified by any cryptographic means until the Four-Way Handshake.

5.1.2 TLS

Assume X and Y have just run the Peer Link Establishment protocol. In order to participate in the Four-Way Handshake, X must first be sure to share a $xxKey_X$ with T, the MKD. If X only shares public key credentials with the MKD and not the $xxKey_X$, then following the completion of the Peer Link Establishment protocol, X and T must run TLS. We define the public key credentials of X as $cred_X := (priv_X, pub_X, pub_{CA}, SIG_{CA}(X, pub_X))$.

TLS was proven secure in [20], however the invariants and thus the proof relied on the assumption that one party always acted as the “client” and one party always acted as the “server.” This will not always be the case as any node could possibly be the MKD of another mesh network. Thus we reprove TLS and give a proof in which the role of each party is not restricted. We mention that this may be of independent interest as often a party will act as both a server and a client in TLS. We give the strands of TLS (in this context) in Figure 5 and the Preconditions, Invariants and Security Goals in Figure 6. We note that although we call the secret exchanged the $xxKey_X$, it is still an arbitrary secret.

$\mathbf{TLS{:}CLNT} = (X, \hat{T}, VerSU_x)$
$[\text{new } Nx; \text{ send } \hat{X}, \hat{T}, Nx, VerSU_x;$
$\text{rcve } \hat{T}, \hat{X}, Nt, VerSU_t, cert_1;$
$\text{mtch } cert_1 / SIG_{CA}(\hat{T}, pub_T);$
$\text{new } xxKey_X; \text{ mtch } SIG_{CA}(\hat{X}, pub_X)/cert_2;$
$\text{mtch } ENC_{pub_T}(xxKey_X)/enc_2;$
$\text{mtch } SIG_{priv_X}(handShake1)/sig_2;$
$\text{mtch Hash}_{xxKey_X}(handShake1, \text{“client”})/hash_2;$
$\text{send } \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2;$
$\text{rcve } \hat{T}, \hat{X}, hash_3;$
$\text{mtch Hash}_{xxKey_X}(handShake2, \text{“server”})/hash_3]_X$

$\mathbf{TLS{:}SRVR} = (T, VerSU_t)$
$[\text{rcve } \hat{X}, \hat{T}, Nx, VerSU_x;$
$\text{new } Nt; \text{ mtch } SIG_{CA}(\hat{T}, pub_T)/cert_1;$
$\text{send } \hat{T}, \hat{X}, Nt, VerSU_t, cert_1;$
$\text{rcve } \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2;$
$\text{mtch } SIG_{CA}(\hat{X}, pub_X)/cert_2;$
$\text{mtch } SIG_{priv_X}(handShake1)/sig_2;$
$\text{mtch } ENC_{pub_T}(xxKey_X)/enc_2;$
$\text{mtch Hash}_{xxKey_X}(handShake1, \text{“client”})/hash_2;$
$\text{mtch } hash_3/\text{Hash}_{xxKey_X}(handShake2, \text{“server”});$
$\text{send } \hat{T}, \hat{X}, hash_3]_Y$

Figure 5: Strands of TLS

Security Requirements of TLS. We follow [20] to choose our desired security properties for TLS.

1. We require that the principals agree on each other’s identity, the protocol completion status (to the point it can be verified), the values of the protocol version, cryptographic suite, and the secret (in this case the $xxKey_X$) that the client sends to the server. For the client this property is given as matching conversations $\Phi_{TLS,AUTH,CLNT}$ and key delivery $\Phi_{TLS,KD,CLNT}$ the analogous properties for the server are given in Section A in the Appendix.

2. We require that only the client and the server know the secret the client generates (in our case the $xxKey_X$), this is realized in the security invariant $\Gamma_{TLS,SI,2}$ in Figure 3.

$\mathbf{\Theta_{TLS}} := \text{Has}(X, cred_X) \wedge \text{Has}(T, cred_T)$

$\mathbf{\Gamma_{TLS,1}} := \text{Honest}(\hat{X}) \wedge \text{Send}(X, m) \wedge$
$\text{Contains}(m, SIG_{priv_X}(handShake1)) \supset$
$\text{Send}(X, \text{TLS1}) < \text{Rcve}(X, \text{TLS2}) <$
$\text{Send}(X, \text{TLS3})$

$\mathbf{\Gamma_{TLS,2}} := \text{Honest}(\hat{T}) \wedge \text{Send}(T, m) \wedge$
$\text{Contains}(m, \text{Hash}_{xxKey_X}(\hat{Y}, \hat{T}, Ny, VerSU_y,$
$\hat{T}, \hat{Y}, Nz, VerSU_t, cert_1,$
$\hat{Y}, \hat{T}, cert_2, sig_2, enc_2, hash_2, \hat{T}, \hat{Y}, \text{“server”})) \supset$
$(Z = T \wedge \text{Rcve}(T, \text{TLS1}) < \text{Send}(T, \text{TLS2}) <$
$\text{Rcve}(T, \text{TLS3}) < \text{Send}(T, \text{TLS4}))$

$\mathbf{\Phi_{TLS,AUTH,CLNT}} :=$
$\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset$
$\text{Send}(X, \text{TLS1}) < \text{Rcve}(T, \text{TLS1}) <$
$\text{Send}(T, \text{TLS2}) < \text{Rcve}(X, \text{TLS2}) <$
$\text{Send}(X, \text{TLS3}) < \text{Rcve}(T, \text{TLS3}) <$
$\text{Send}(T, \text{TLS4}) < \text{Rcve}(X, \text{TLS4})$

$\mathbf{\Phi_{TLS,KD,CLNT}} :=$
$\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset$
$\text{Has}(\hat{X}, xxKey_X) \wedge \text{Has}(\hat{T}, xxKey_X)$

Figure 6: Preconditions Invariants and Security Goals for the Client of TLS

To formally show that the security requirements listed above hold we must show the following theorem.

Theorem 2.

(i) $\Gamma_{TLS,\{1,2,\}} \wedge \Gamma_{TLS,SI,\{1,2\}} \vdash$
$\Theta_{TLS}[\textbf{TLS} : \textbf{CLNT}]_X \Phi_{TLS,\{AUTH,KD\},CLNT}$

(ii) $\Gamma_{TLS,\{1,2,\}} \wedge \Gamma_{TLS,SI,\{1,2\}} \vdash$
$\Theta_{TLS}[\textbf{TLS} : \textbf{SRVR}]_T \Phi_{TLS,\{AUTH,KD\},SRVR}$

(iii) TLS $\vdash \Gamma_{TLS,\{1,2\}} \wedge \Gamma_{TLS,SI,\{1,2\}}$

Enumeration (i) reads “Given preconditions Θ_{TLS} and invariants $\Gamma_{TLS,1}, \Gamma_{TLS,2}, \Gamma_{TLS,SI,1}$ and $\Gamma_{TLS,SI,2}$ upon executing the Client role the security goals $\Phi_{TLS,AUTH,CLNT}, \Phi_{TLS,KD,CLNT}$ are guaranteed to hold.” Enumeration (ii) is the analogous to (i), but for the server. Enumeration (iii) shows that $\Gamma_{TLS,\{1,2\}}$ and $\Gamma_{TLS,SI,\{1,2\}}$ are invariants of TLS. We give the formal proof of (i) and (ii) in the appendix. We give the proof of the security invariants in Section 7. We give a general overview of all the proofs below.

Client’s View. If we assume that the server is honest, the server is the only one who has the private key to decrypt the $xxKey_X$ that the client sent with the server’s public key and neither will send out the $xxKey_X$; this is captured in $\Gamma_{TLS,SI,1}$ and $\Gamma_{TLS,SI,2}$. Thus a hash under the $xxKey_X$ can only come from the client or the server. However due to the form of the hash from $\Gamma_{TLS,2}$ we know it came from the server. The hash from the server contains the entire conversation thus far and it matches the record of the conversation that the client has, thus the client and server must share all the same variables. Additionally from $\Gamma_{TLS,2}$ we know that the server will only send a hash of the entire conversation if it has sent and received all the previous messages of this TLS session in the correct order. We can further order the messages using the freshness of the nonces. Combining all these parts we get matching conversations. Key Distribution is trivial as we have previously shown they share the same variables.

Server’s View. If we assume that the client is honest only the client has $priv_X$. Again this is captured in $\Gamma_{TLS,SI,1}$. Thus the signature the server receives can only come from the client. The signature contains the entire conversation thus far including the encryption of the $xxKey_X$. Thus the client and the server must share all the same variables used to this point. We are given from $\Gamma_{TLS,1}$ that if the client sends out the signature of the entire conversation, the client must have sent and received the previous messages of this TLS session in the correct order. We can order the messages using the freshness of the nonces and the fact that if a message arrives that only the client could have sent than the client must have sent it before it arrived. Combining all these parts we get matching conversations. Again Key Distribution is trivial.

Invariants. We need to show that the invariants hold throughout all the basic sequences. This is normally a simple task; it is not in this case. $\Gamma_{TLS,1}$ requires that the client send and receive the first two messages of the protocol before sending the third and similarly $\Gamma_{TLS,2}$ requires that the server send and receive the first three messages before sending the final message. While this is true at this point in the protocol, it cannot be shown to be true while looking at a single basic sequence in isolation. Thus we implement the proof technique from Section 3. We note that after each basic sequence any send or receive action will be a post condition of that basic sequence. Also we note that any precondition to a basic sequence that is solely a send or receive action will be a postcondition to that basic sequence as well. This enables us to show that $\Gamma_{TLS,1}$ and $\Gamma_{TLS,2}$ indeed hold over the run of this protocol. The security invariants are proved later in Section 7.

5.1.3 Four-Way Handshake

The Four-Way Handshake will verify the information exchanged in the Peer Link Establishment protocol as well as establish a session key between X and Y. We present the strands of the MSA Four-Way Handshake in Figure 7 and give the preconditions, invariants and selected security goals of the Candidate MP in Figure 8. The security goals of the MA are analogous and can be found in the appendix in Section B.1.1. The security invariants have previously been given in Figure 3.

The Four-Way Handshake is based on the 802.11i Four-Way Handshake, but it is fundamentally different. In the *ptk* derivation function in MSA, the sender of the first message's nonce will always be after the recipient of the first message's nonce. In 802.11i the nonces in the derivation function are ordered by value. Ordering the nonces by the sender and receiver of a set of messages allows each party to know who is the initiator and who is the recipient of any MIC'd message, without carrying any state. This is because the nonce ordering in the key (and thus the key itself) along with the message id *distinctly identifies the sender of the message.* In MSA, based on the key and the message id, the sender of the message is ambiguous without keeping state. In [5], the ordering of the nonces is slightly different, based on role selection, but it is unambiguous based on other information.

We prove a slight modification of the four-way handshake proposed in [5]. We add a nonce from the MA in this protocol, so that the derived key is provably fresh from both perspectives. Thus, the *ptk* derivation is not based off of the MKD-nonce for the candidate MP (a static value we denote TN_x for T-Nonce for X). Instead, we generate a nonce at the MA (Y) and use that in the derivation of the *ptk* and propose using the TN_x only for deriving portions of the key hierarchy. Without this change, key freshness could not be proven.

Although the 802.11i Four-Way Handshake was proven secure in [20] we can not reuse that proof for two main reasons. First, the proof from [20] was reliant on one party only ever carrying out one role. Thus, to have any guarantee of security the parties could not exchange roles. We can not make this guarantee for mesh networks. Second, the handshake in 802.11i assumes that the parties already have established keys. Again this is not the case in a mesh of peers, thus we need a proof that deals with with a mid-protocol key derivation and a key pull.

4WAY:INIT $= (X, \hat{Y}, INFO_X, INFO_Y, gtk_X)$
$[$rcve $\hat{X}, \hat{Y}$, "IAUTH5"$, y, TN_x$;
mtch $SELECT(INFO_X, INFO_Y)/CS, pmkN$;
mtch $HASH(xxKeyey, TN_x, \hat{X}, \hat{Y})/pmkmkd_X$;
mtch $HASH(pmkmkd_X)/pmk_{X,Y}$;
mtch $RETRIEVE(pmkN)/pmk$;
new x; mtch $HASH_{pmk}(x, y)/ptk_{X,Y}$;
mtch $\mathrm{enc}_{ptk_{X,Y}}(gtk_X)/enc_1$;
mtch $\mathrm{HASH}_{ptk_{X,Y}}$
 ("IAUTH6"$, x, pmkN, INFO_X, enc_1)/mic_1$;
send $\hat{Y}, \hat{X}$, "IAUTH6"$, x, mic_1, INFO_X, enc_1$;
rcve $\hat{X}, \hat{Y}$, "IAUTH7"$, y, mic_2, INFO_Y, TN_x, enc_2$;
mtch $enc_2/\mathrm{enc}_{ptk_{X,Y}}(gtk_Y)$;
mtch $mic_2/\mathrm{HASH}_{ptk_{X,Y}}$
 ("IAUTH7"$, y, pmkN, INFO_Y, TN_x, enc_2)$;
mtch $\mathrm{HASH}_{ptk_{X,Y}}($"IAUTH8"$)/mic_3$;
send $\hat{Y}, \hat{X}$, "IAUTH8"$, mic_3]_X$

4WAY:MA $= (Y, INFO_Y, INFO_X, gtk_Y)$
$[$mtch $SELECT(INFO_X, INFO_Y)/CS, pmkN$;
mtch $RETRIEVE(pmkN)/pmk, TN_x$;
new y; send $\hat{X}, \hat{Y}$, "IAUTH5"$, y, TN_x$;
rcve $\hat{Y}, \hat{X}$, "IAUTH6"$, x, mic_1, INFO_X, enc_1$;
mtch $HASH_{pmk}(x, y)/ptk_{X,Y}$;
mtch $enc_1/\mathrm{enc}_{ptk_{X,Y}}(gtk_X)$;
mtch $mic_1/\mathrm{HASH}_{ptk_{X,Y}}$
 ("IAUTH6"$, x, pmkN, INFO_X, enc_1)$;
mtch $enc_2/\mathrm{enc}_{ptk_{X,Y}}(gtk_Y)$;
mtch $mic_2/\mathrm{HASH}_{ptk_{X,Y}}$
 ("IAUTH7"$, y, pmkN, INFO_Y, TN_x, enc_2)$;
send $\hat{X}, \hat{Y}$, "IAUTH7"$, y, mic_2, INFO_Y, TN_x, enc_2$;
rcve $\hat{Y}, \hat{X}$, "IAUTH8"$, mic_3$;
mtch $mic_3/\mathrm{HASH}_{ptk_{X,Y}}($"IAUTH8"$)]_Y$

Figure 7: Strands of the Four-Way Handshake

Recall that X is a candidate MP and that Y is a MA. X and Y have not communicated as part of this mesh thus they will not share a key in the mesh. Since X is not a MA and Y is a MA we know that select() will output $pmkN_{X,Y}$. It therefore is the case that when Y executes the retrieve function as part of the Four-Way Handshake Y will need to perform a Key Pull protocol with the MKD, as Y is the only MA. It is simple to verify that Y satisfies the preconditions of the retrieve function since $\mathrm{Has}(Y, mptk_{Y,T})$ is a precondition of the Four-Way Handshake. We also mention as a side note that Y will retrieve the TN_x from the MKD in addition to $pmk_{X,Y}$ and must pass the $mdkNonce_X$ along to X. Now we must verify that X satisfies the retrieve precondition at the point he executes the function. This point is not directly obvious, but still simple to verify. Since X is a Candidate MP Y will pass the TN_x to X.

Once X has the $xxKey_X$ and the TN_x, X can derive the $pmk_{X,Z}$ for all Z and thus can derive $pmk_{X,Y}$.

In order to achieve our security goals we needed to modify the Four-Way Handshake protocol from the current MSA specification. Currently, the MA only sends a single nonce that was received from the MKD in the first message. The Candidate MP uses this nonce as both the TN_x and as Y's contribution to the $ptk_{X,Y}$. However, if this is done then Y can not contribute to the $ptk_{X,Y}$ and thus we can not show key freshness. Thus we recommend an extra nonce be added to the Four Way Handshake protocol. Then Y's nonce can be used to derive the $ptk_{X,Y}$ and the TN_x can be used for its sole purpose of deriving the $pmkmkd_X$ and the $mkdk_{X,T}$.

$$\begin{array}{l}\boldsymbol{\Theta}_{\mathbf{4WAY,INIT,1}} := \boldsymbol{\Phi}_{\mathbf{PLE,INIT,1}}\\ \boldsymbol{\Theta}_{\mathbf{4WAY,INIT,2}} := \boldsymbol{\Phi}_{\mathbf{PLE,INIT,2}}\end{array}$$

$$\begin{array}{l}\boldsymbol{\Theta}_{\mathbf{4WAY,MA,1}} := \boldsymbol{\Phi}_{\mathbf{PLE,MA,1}}\\ \boldsymbol{\Theta}_{\mathbf{4WAY,MA,2}} := \boldsymbol{\Phi}_{\mathbf{PLE,MA,2}}\end{array}$$

$$\begin{array}{l}\boldsymbol{\Gamma}_{\mathbf{4WAY,1}} := \text{Honest}(\hat{X}) \wedge \text{Send}(X, m)\wedge\\ (\text{Contains}(m, HASH_{ptk_{Z,Y}}(\text{“IAUTH6”}))\vee\\ \text{Contains}(m, HASH_{ptk_{Y,Z}}(\text{“IAUTH7”}))\vee\\ \text{Contains}(m, HASH_{ptk_{Z,Y}}(\text{“IAUTH8”}))) \supset\\ \hat{Z} = \hat{X}\end{array}$$

$$\begin{array}{l}\boldsymbol{\Phi}_{\mathbf{4WAY,AUTH,INIT}} :=\\ \text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset\\ (\Theta_{4WAY,INIT,2} \wedge \Theta_{4WAY,MA,2}\wedge\\ \text{Send}(Y, \text{IATH5})) < \text{Rcve}(X, \text{IATH5}) <\\ \text{Send}(X, \text{IATH6}) < \text{Rcve}(Y, \text{IATH6}) <\\ \text{Send}(Y, \text{IATH7}) < \text{Rcve}(X, \text{IATH7}) <\\ \text{Send}(X, \text{IATH8})\end{array}$$

$$\begin{array}{l}\boldsymbol{\Phi}_{\mathbf{4WAY,PTKD,INIT}} :=\\ \text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset\\ \text{Has}(X, ptk_{X,Y}) \wedge \text{Has}(Y, ptk_{X,Y})\end{array}$$

$$\begin{array}{l}\boldsymbol{\Phi}_{\mathbf{4WAY,GTKD,INIT}} :=\\ \text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset\\ \text{Has}(X, gtk_Y) \wedge \text{Has}(Y, gtk_X)\end{array}$$

$$\begin{array}{l}\boldsymbol{\Phi}_{\mathbf{4WAY,KF,INIT}} :=\\ \text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset\\ \text{new } (\hat{X}, x) \wedge x \subseteq ptk_{X,Y} \wedge \text{FirstSend}(Y, y, \text{IATH5})\wedge\\ \text{new } (\hat{Y}, y) \wedge y \subseteq ptk_{X,Y} \wedge \text{FirstSend}(X, x, \text{IATH6})\end{array}$$

$$\begin{array}{l}\boldsymbol{\Phi}_{\mathbf{4WAY,INFO,INIT}} :=\\ \text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset\\ SELECT(INFO_X, INFO_Y) = CS, pmkN\wedge\\ \text{Has}(X, CS, pmkN) \wedge \text{Has}(Y, CS, pmkN)\end{array}$$

Figure 8: The Preconditions, Invariants and Security Goals of the Four-Way Handshake

We now give our security goals taken from [20], which were based on the security goals given in the 802.11i standard. We adapted them slightly for the slight differences in the protocol.

1. Confirm the existence of the PMK at the peer; this is realized in the matching conversation security goals , $\Phi_{4WAY,AUTH,INIT}$ and $\Phi_{4WAY,AUTH,MA}$.

2. Ensure that the *ptk* is fresh; this is realized in the key freshness security goal, $\Phi_{4WAY,KF,INIT}$ and $\Phi_{4WAY,KF,MA}$.

3. Synchronize the installation of the session keys into the MAC; realized in the *ptk* delivery goal, $\Phi_{4WAY,PTKD,INIT}$ and $\Phi_{4WAY,PTKD,MA}$.

4. Transfer the GTK from the the MA to the Candidate MP and from the Candidate MP to the MA, realized in *gtk* delivery goal, $\Phi_{4WAY,GTKD,INIT}$ and $\Phi_{4WAY,GTKD,MA}$.

5. Confirm the selection of miscellaneous security information, realized in verified information exchange goal, $\Phi_{4WAY,INFO,INIT}$ and $\Phi_{4WAY,INFO,MA}$.

We now give the formal security theorem below, as usual the first two enumerations represent the security goals of the Candidate MP and the MA respectively and the third enumeration is a verification that the protocol does not violate its invariants.

Theorem 3.

(i) $\Gamma_{4WAY,1} \wedge \Gamma_{4WAY,SI,\{1,2,3,4\}} \vdash$
$\Theta_{4WAY,INIT,\{1,2\}}[\textbf{4WAY:INIT}]_X$
$\Phi_{4WAY,\{AUTH,PTK,GTKD,KF,INFO\},INIT}$

(ii) $\Gamma_{4WAY,1} \wedge \Gamma_{4WAY,SI,\{1,2,3,4\}} \vdash$
$\Theta_{4WAY,MA,\{1,2\}}[\textbf{4WAY:MA}]_Y$
$\Phi_{4WAY,\{AUTH,PTK,GTKD,KF,INFO\},MA}$

(iii)4WAY $\vdash \Gamma_{4WAY,1} \wedge \Gamma_{4WAY,SI,\{1,2,3,4\}}$

We walk through the proof briefly here, a full version is in the appendix in Section B.1.1.

Candidate MPs's View From precondition $\Theta_{4WAY,INIT,1}$, the Candidate MP has the messages he sent and received in the Peer Link Establishment. Also on protocol completion MA knows he sent and received the messages of the Four-Way Handshake. From the receipt of the MIC in IATH7, the MA know whoever sent the message has the $ptk_{X,Y}$. From the security invariants on the system he knows that if everyone with access to the *pmk* is behaving honestly (X, Y, T) then IATH7 must have come from one of these principals. But, from the invariants on the protocol he can tell using the message identifier in the MIC of IATH7 and the order of the nonce in the $ptk_{X,Y}$ that the MIC came from Y. Since every protocol's message identifiers are unique (with the exception of TLS), X uses the message identifiers and knows that Y must be participating in the Four Way Handshake. Thus as part of the Four Way Handshake X knows Y must have earlier verified the MIC that X sent in IATH6, before sending IATH7. Since Y must have hashed both the MIC in message 7 and the MIC in message 6 (to verify it), X knows he indeed shares all the variables with Y. Thus X can now confirm that every message sent and received by Y in both the Peer Link Establishment and the Four Way Handshake match X's. With some more temporal tricks X can get matching conversations for the Four-Way Handshake protocol. While we can not necessarily order the messages of the Peer Link Establishment protocol, we can still ensure that the messages are identical. Thus we have established $\Phi_{4WAY,AUTH,INIT}$ and the remaining security goals follow without much work.

MA's View The MA receives IATH8 and with similar methods as given in the proof of the Candidate MP's view Y can tell that X hashed the MIC in IATH8. Then the MA knows that X is indeed participating in the Four-Way Handshake and thus must have computed the MIC in IATH7 in order to verify it before X sent IATH8. Again from the MIC in IATH6 and the invariants on the system Y can prove that X sent IATH6. Thus Y can verify that he and X share all the same variables. And with a few more timing tricks we can ensure $\Phi_{4WAY,AUTH,MA}$ holds and the remaining security goals follow.

Invariants. The invariants are straightforward and it can easily be verified they hold over the basic sequences.

We are now done describing the MSA Authentication protocol in the case of a Candidate MP and a MA.

5.1.4 MSA Authentication without a Candidate MP

In the previous subsubsections we looked at MSA Authentication when one of the parities is a Candidate MP. However, the MSA Authentication protocol can be run between a MP or MA and a MP or MA, so long as one of the parties is a MA. We continue to assume, without loss of generality that Y is the MA. This fundamentally changes nothing in the security proofs; it is just a different path through the initial handshake. There is no TLS step in this case, the protocol proceeds from PLE directly to 4WAY. The composition is proven in Section 8.

5.2 Mesh Key Holder Security Handshake

The MKHSH is used to set up a session key, the $mptk_{X,T}$ between a MP (X) and the MKD (T). Once MKHSH completes successfully the MP becomes an MA. We give the strands, preconditions and invariants in figure 9. The security goals are nearly identical to those of the Four-Way Handshake and thus have been moved to the appendix in Section C. The security invariant has been previously given in Figure 3.

```
MKHSH:INIT = $(X, \hat{T}, INFO_X)$
[new $x$; send "MKH1", $x, \hat{X}, \hat{T}, INFO_X$;
rcve "MKH2", $x, t, \hat{X}, \hat{T}, INFO_T, mic_0$;
mtch $SELECT(INFO_X, INFO_T)/CS$;
mtch $\mathrm{HASH}_{mkdk_{X,T}}(x, t)/mptk_{X,T}$;
mtch $mic_0/\mathrm{HASH}_{mptk_{X,T}}($
    "MKH2", $x, t, \hat{X}, \hat{T}, INFO_T)$;
mtch $mic_1/\mathrm{HASH}_{mptk_{X,T}}($"MKH3", $x, t, \hat{X}, \hat{T}, INFO_X)$;
send "MKH3", $x, t, \hat{X}, \hat{T}, INFO_X, mic_1$;
rcve "MKH4", $x, t, \hat{X}, \hat{T}, INFO_T, mic_2$;
mtch $mic_2/\mathrm{HASH}_{mptk_{X,T}}($"MKH4", $x, t, \hat{X}, \hat{T}, INFO_T)]_X$

MKHSH:RESP = $(T, INFO_T)$
[rcve "MKH1", $x, \hat{X}, \hat{T}, INFO_X$;
mtch $SELECT(INFO_X, INFO_T)/CS$;
new $t$; mtch $HASH_{mkdk_{X,T}}(x, t)/mptk_{X,T}$;
mtch $\mathrm{HASH}_{mptk_{X,T}}($
    "MKH2", $x, t, \hat{X}, \hat{T}, INFO_T)/mic_0$;
send "MKH2", $x, t, \hat{X}, \hat{T}, INFO_T, mic_0$;
rcve "MKH3", $x, t, \hat{X}, \hat{T}, INFO_X, mic_1$;
mtch $mic_1/\mathrm{HASH}_{mptk_{X,T}}($"MKH3", $x, t, \hat{X}, \hat{T}, INFO_X)$;
mtch $mic_2/\mathrm{HASH}_{mptk_{X,T}}($"MKH4", $x, t, \hat{X}, \hat{T}, INFO_T)$;
send "MKH4", $x, t, \hat{X}, \hat{T}, INFO_T, mic_2]_T$

$\mathbf{\Theta_{MKHSH,1}} := \mathrm{Has}(X, mkdk_{X,T}) \wedge \mathrm{Has}(T, mkdk_{X,T})$

$\mathbf{\Gamma_{MKHSH,1}} := \mathrm{Honest}(\hat{X}) \wedge \mathrm{Send}(X, m) \wedge$
$(\mathrm{Contains}(m, \mathrm{HASH}_{mptk_{X,T}}($"MKH2"$, (\hat{Y}, \hat{Z}))) \vee$
$\mathrm{Contains}(m, \mathrm{HASH}_{mptk_{X,T}}($"MKH3"$, (\hat{Z}, \hat{Y}))) \vee$
$\mathrm{Contains}(m, \mathrm{HASH}_{mptk_{X,T}}($"MKH4"$, (\hat{Y}, \hat{Z}))) \supset$
$\hat{Z} = \hat{X}$
```

Figure 9: Strands, preconditions and invariants of the MKHSH

This is a Four Way Handshake that is used for setting up a key. From the point of view of this paper it is very similar to the Four-Way Handshake in MSA presented earlier. Once again, it is possible to tell who sent which message based solely on the MIC of the message. However in the MKHSH protocol it is done in a more straightforward way than in the Four-Way Handshake. In the MKHSH protocol, each message that is MIC'd contains the MA's address ordered before the MKD's address in the MIC (this is capture in the invariant $\Gamma_{MKHSH,1}$). Thus a reflection attack is not possible. Still the ideas are similar enough that the proof is almost identical. We give the formal security theorem below.

Theorem 4.

(i) $\Gamma_{MKHSH,1} \wedge \Gamma_{MKHSH,SI,1} \vdash$
$\Theta_{MKHSH,INIT,1}[\textbf{MKHSH:MA}]_X$
$\Phi_{MKHSH,\{AUTH,MPTK,GTKD,KF,INFO\},MA}$

(ii) $\Gamma_{MKHSH,1} \wedge \Gamma_{MKHSH,SI,1} \vdash$
$\Theta_{MKHSH,MA,1}[\textbf{MKHSH:MKD}]_T$
$\Phi_{MKHSH,\{AUTH,MPTKD,GTKD,KF,INFO\},MKD}$

(iii)MKHSH $\vdash \Gamma_{MKHSH,1} \wedge \Gamma_{MKHSH,SI,1}$

The proof is nearly identical to the proof of the Four-Way Handshake and the differences are neither enlightening nor interesting. It has been omitted.

5.3 Group Key Handshake

If a MA or MP has a security connection (shares a session key) and wants to update its group transfer key, it must distribute its new group transfer key to every principal it has a connection with. To do this it will run the Group Key Handshake protocol.

The protocol is a simple two-round protocol, in which a sender, X, sends its group transfer key to a receiver, Y and the receiver acknowledges the reception. The messages are protected by MICs using the a portion of the $ptk_{X,Y}$ and the gtk_X is encrypted again using the $ptk_{X,Y}$. The sender maintains a message counter for every principal it has a connection with, and it updates this counter after every key exchange message sent.

In examining the protocols we found the current MSA Group Key Handshake is vulnerable to a reflection attack as in the messages there is no indication of who the sender or the intended receiver of the message was. The protocol was adopted from 802.11i and was not modified to ensure security in a mesh network. We modify the Group Key Handshake to include the sender and receiver's addresses ordered inside the MICs. This stops the reflection attacks and allows us to prove the protocol secure.

The Group Key Handshake protocol contains the "return to global" function that we introduced in Section 3.4. It will be used to return the new message counter value, represented by $ngtkc_{X,Y}$, to the global level. Thus, once the basic sequence is complete, any role that uses the $gtkc_{X,Y}$ value will be using the updated global message counter number $globalgtkc_{X,Y} = ngtkc_{X,Y}$. Recall that in PCL basic sequences are autonomous steps and thus we are guaranteed that the value that is updated at the global level will not be used until after the basic sequence that updated it completes.

We give the strands, preconditions and invariants in Figure 10 and note that GRP1HASH and GRP2HASH represent the hashes in the first and second messages of the group key handshake, instantiated with the appropriate principals variables. The security invariants can be found in Figure 3. While $\Gamma_{GKH,2}$ is listed as an invariant it could be considered a security invariant as it has security properties within itself. It essentially states that a message with a higher counter was sent out after a message with a lower counter. The stated security goals are simply matching conversations and group key delivery; they are straightforward and have been moved to Section D. This leads to a theorem about the correctness of the Group Key Handshake, similar to previous theorems.

Theorem 5.

(i) $\Gamma_{GKH,\{1,2\}} \wedge \Gamma_{GKH,SI,\{1,2\}} \vdash$
$\Theta_{GKH,\{1,2\}}[\textbf{GKH:SNDR}]_X$
$\Phi_{GKH,\{AUTH,GTKD\},SNDR}$

(ii) $\Gamma_{GKH,\{1,2\}} \wedge \Gamma_{GKH,SI,\{1,2\}} \vdash$
$\Theta_{GKH,\{1,2\}}[\textbf{GKH:RCVR}]_Y$
$\Phi_{GKH,AUTH,RCVR}$

(iii)GKH $\vdash \Gamma_{GKH,\{1,2\}} \wedge \Gamma_{GKH,SI,\{1,2\}}$

We have omitted the proofs (i) and (ii) as they are nearly identical to the proof given of the Delete protocol in section E. However, the proof that $\Gamma_{GKH,2}$ holds over every basic sequence of the Group Key Handshake is one of the most technically challenging proofs we present. The proof requires that the precondition $\Theta_{GKH,2}$ holds after every basic sequence, thus we additionally show this property in the proof. The proof is quite original as we are using the new function introduced in this paper, return(). We give the formal proof in Section D in the appendix, but give an overview below.

Proof Sketch: GKH $\vdash \boldsymbol{\Theta_{GKH,2} \wedge \Gamma_{GKH,2}}$**, SNDR.** First we need to show the precondition holds over all

```
GKH:SNDR =
$(X, \hat{Y}, gtkc_{X,Y}, ptk_{X,Y}, gtk_X)$
[mtch $ngtkc_{X,Y}$/Increment($gtk_{X,Y}$);
return $ngtkc_{X,Y}$; mtch $enc_1$/ENC$_{ptk_{X,Y}}$($gtk_X$);
mtch $mic_1$/HASH$_{ptk_{X,Y}}$(
    "GKH1", $ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1$);
send "GKH1", $ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1$;
rcve "GKH2", $ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X}$;
mtch $mic_2$/HASH$_{ptk_{X,Y}}$(
    "GKH2", $ngtkc_{X,Y}, \hat{Y}, \hat{X}$)]$_X$

GKH:RCVR =
$(Y, \hat{X}, gtkc'_{X,Y}, ptk_{X,Y})$
[rcve "GKH1", $ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1$;
IsLess($gtkc'_{X,Y}, ngtkc_{X,Y}$);
mtch $enc_1$/ENC$_{ptk_{X,Y}}$($gtk_X$);
mtch $mic_1$/HASH$_{ptk_{X,Y}}$(
    "GKH1", $ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1$);
return $ngtkc_{Y,T}$;
mtch $mic_2$/HASH$_{ptk_{X,Y}}$(
    "GKH2", $ngtkc_{X,Y}, \hat{Y}, \hat{X}$);
send "GKH2", $ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X}$]$_Y$
```

$\mathbf{\Theta_{GKH,1}}$:= Has$(X, ptk_{X,Y})$ ∧ Has$(Y, ptk_{X,Y})$

$\mathbf{\Theta_{GKH,2}}$:= Send(X, m) ∧ Contains$(m, gtkc'_{X,Y})$ ⊃ IsLess$(gtkc'_{X,Y}$, Increment$(globalgtkc_{X,Y}))$

$\mathbf{\Gamma_{GKH,1}}$:= Honest$(\hat{X})$ ∧ Send(X, m)∧ (Contains$(m$, Hash$_{ptk}$("GKH1", $\hat{Z}, \hat{Y}$))∨ Contains$(m$, Hash$_{ptk}$("GKH2", $\hat{Y}, \hat{Z}$))) ⊃ $\hat{Z} = \hat{X}$

$\mathbf{\Gamma_{GKH,2}}$:= Honest$(\hat{X})$ ∧ Send(X, m_0) ∧ Send(X, m_1)∧ Contains$(m_0$, HASH$_{ptk_{X,Y}}(ngtkc'_{X,Y}$, GRP1HASH)∧ = Contains$(m_1$, HASH$_{ptk_{X,Y}}(ngtkc_{X,Y}$, GRP1HASH)∧ IsLess$(gtkc'_{X,Y}, gtkc_{X,Y})$ ⊃ Send(X, m_0) < Send(X, m_1)

$\mathbf{\Gamma_{GKH,3}}$:= Honest$(\hat{X})$ ∧ Send(X, m_0) ∧ Send(X, m_1)∧ Contains$(m_0$, HASH$_{ptk_{X,Y}}(ngtkc'_{X,Y}$, GRP2HASH))∧ Contains$(m_1$, HASH$_{ptk_{X,Y}}(ngtkc_{X,Y}$, GRP2HASH))∧ IsLess$(gtkc'_{X,Y}, gtkc_{X,Y})$ ⊃ Send(X, m_0) < Send(X, m_1)

Figure 10: Strands, preconditions and Invariants of the Group Key Handshake

basic sequences. The precondition essentially states that any counter value that has been sent out in the past is smaller than the incremented value of the global counter, i.e. $ngtkc_{X,Y}$ < Increment($globalgtkc_{X,Y}$). We need to show this precondition holds over the first basic sequence, i.e. we need to show that if the precondition is true at the start of the protocol it is still true after the send of the first message GKH1. The only counter value sent out in the first basic sequence was $newgtkc_{X,Y}$. Recall, $globablgtkc_{X,Y} = newgtkc_{X,Y}$ from the return function. Thus we have $ngtkc_{X,Y}$ < Increment($globalgtkc_{X,Y}$), by the definition of Increment.

Now assuming the precondition and the invariant held before the first basic sequence we must show that the invariant still holds. Recall that the invariant essentially states that a message with a higher counter was first sent out after a message with a lower counter. From the precondition we know that *all messages sent prior to this basic sequence have a counter number smaller than* $newgtkc_{X,Y}$. This value is GTK1's message counter value. Thus, GTK1 did not violate the invariant and the invariant still holds after the basic sequence. The proof of the second basic sequence is trivial as it does not contain a send.

Proof Sketch: GKH ⊢ $\mathbf{\Theta_{GKH,2}} \wedge \mathbf{\Gamma_{GKH,3}}$, RCVR. We assume that the precondition holds and we need to show that it holds over the first basic sequence. The message sent out during the first basic sequence has a counter value of $ngtkc_{X,Y}$, which is the same value as the current global counter value. Thus $ngtkc_{X,Y}$ < Increment($globalgtkc_{X,Y}$) continues to hold.

We can show the invariant holds over the first basic sequence in much the same was as in the SNDR case. We know from the precondition that all messages sent prior to the start of this basic sequence contain a smaller counter value than the value of the message we send during this first basic sequence. Thus we do not violate the invariant sending the message of the first basic sequence. This completes the proof sketch.

5.4 Push, Pull and Delete

The Push, Pull and Delete protocols are protocols between a MA and the MKD for key management of the *pmk*'s. We give the strands, preconditions and invariants of the all three protocols below, some of the invariants have been moved to the appendix as they are similar to $\Gamma_{PPD,2}$. The security invariants can be found in Figure 3.

```
PULL:MA = $(Y, \hat{T}, \hat{X}, pmkN_{X,Y}, plc_{Y,T})$
[mtch $nplc_{Y,T}$/Increment($plc_{Y,T}$);
return $nplc_{Y,T}$;
mtch $mic_1$/HASH$_{mptk_{Y,T}}$(
    $\hat{Y}, \hat{T}$, "PULL1", $nplc_{Y,T}, \hat{X}, pmkN_{X,Y}$);
send $\hat{T}, \hat{Y}$, "PULL1", $nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1$;
rcve $\hat{Y}, \hat{T}$, "PULL2", $nplc_{Y,T}, \hat{X}, pmkN_{X,Y}$,
    $TN_x, enc_1, mic_2$;
mtch $enc_1$/ENC$_{mptk_{Y,T}}$($pmk_{X,Y}$);
mtch $mic_2$/HASH$_{mptk_{Y,T}}$($\hat{Y}, \hat{T}$, "PULL2",
```

$nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, TN_x, enc_1)]_Y$

PULL:MKD $= (T, pmk_{X,Y}, plc'_{Y,T})$
[rcve $\hat{T}, \hat{Y}$, "PULL1", $nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1$;
IsLess($plc'_{Y,T}, nplc_{Y,T}$);
mtch mic_1/HASH$_{mptk_{Y,T}}$(
$\hat{Y}, \hat{T}$, "PULL1", $nplc_{Y,T}, \hat{X}, pmkN_{X,Y}$);
mtch enc_1/ENC$_{mptk_{Y,T}}$($pmk_{X,Y}$);
return $nplc_{Y,T}$;
mtch mic_2/HASH$_{mptk_{Y,T}}$($\hat{Y}, \hat{T}$, "PULL2",
$nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, TN_x, enc_1$);
send $\hat{Y}, \hat{T}$, "PULL2", $nplc_{Y,T}, \hat{X}, pmkN_{X,Y}$,
$TN_x, enc_1, mic_2]_T$

PUSH:MA $= (Y, plc_{Y,T} = nplc_{Y,T}, pdc'_{Y,T})$
[rcve $\hat{Y}, \hat{T}$, "PUSH1", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0$;
mtch mic_0/HASH$_{mptk_{Y,T}}$(
$\hat{Y}, \hat{T}$, "PUSH1", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}$);
IsLess($pdc'_{Y,T}, npdc_{Y,T}$);
return $npdc_{Y,T}$;
PULL:MA$]_Y$

PUSH:MKD =
$(T, \hat{Y}, \hat{X}, pmkN_{X,Y}, pmk_{X,Y}, pdc_{Y,T}, plc'_{Y,T})$
[mtch $npdc_{Y,T}$/Increment($pdc_{Y,T}$);
return $npdc_{Y,T}$;
mtch mic_0/HASH$_{mptk_{Y,T}}$(
$\hat{Y}, \hat{T}$, "PUSH1", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}$);
send $\hat{Y}, \hat{T}$, "PUSH1", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0$;
PULL:MKD$]_T$

DEL:MA := (Y)
[rcve $\hat{Y}, \hat{T}$, "DEL1", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0$;
mtch mic_0/HASH$_{mptk_{Y,T}}$(
$\hat{Y}, \hat{T}$, "DEL1", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}$);
IsLess($pdc'_{Y,T}, npdc_{Y,T}$);
return $npdc_{Y,T}$;
mtch mic_1/HASH$_{mptk_{Y,T}}$(
$\hat{Y}, \hat{T}$, "DEL2", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}$);
send $\hat{T}, \hat{Y}$, "DEL2", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1]_Y$

DEL:MKD := $(T, \hat{Y}, \hat{X}, pmkN_{X,Y}, pmk_{X,Y})$
[mtch $npdc_{Y,T}$/Increment($pdc_{Y,T}$);
return $npdc_{Y,T}$;
mtch mic_0/HASH$_{mptk_{Y,T}}$(
$\hat{Y}, \hat{T}$, "DEL1", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}$);
send $\hat{Y}, \hat{T}$, "DEL1", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0$;
rcve $\hat{T}, \hat{Y}$, "DEL2", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1$;
mtch mic_1/HASH$_{mptk_{Y,T}}$(
$\hat{Y}, \hat{T}$, "DEL2", $npdc_{Y,T}, \hat{X}, pmkN_{X,Y}$)$]_T$

$\mathbf{\Theta_{PPD,1}}$:= Has($T, mptk_{Y,T}$) $\wedge$ Has($Y, mptk_{Y,T}$)

$\mathbf{\Theta_{PPD,2}}$:= Send(Y, m) $\wedge$ Contains($m, plc'_{Y,T}$) $\supset$
IsLess($plc'_{Y,T}$, Increment($globalplc_{Y,T}$))$\wedge$
Send(T, m) $\wedge$ Contains($m, pdc'_{Y,T}$) $\supset$
IsLess($pdc'_{Y,T}$, Increment($npdc_{Y,T}$))

$\mathbf{\Gamma_{PPD,1}}$:= Honest(Y) $\wedge$ Send(Y, m)$\wedge$
(Contains(m, HASH$_{mptk_{Y,T}}$($\hat{Z}, \hat{X}$, "PULL1"))$\vee$
Contains(m, HASH$_{mptk_{Y,T}}$($\hat{X}, \hat{Z}$, "PULL2"))$\vee$
Contains(m, HASH$_{mptk_{Y,T}}$($\hat{X}, \hat{Z}$, "PUSH1"))$\vee$
Contains(m, HASH$_{mptk_{Y,T}}$($\hat{X}, \hat{Z}$, "DEL1"))$\vee$
Contains(m, HASH$_{mptk_{Y,T}}$($\hat{Z}, \hat{X}$, "DEL2"))) $\supset$
$\hat{Z} = \hat{Y}$

$\mathbf{\Gamma_{PPD,2}}$:= Honest($\hat{Z}$)$\wedge$
(Send(Z, m_0) $\wedge$ Send(Z, m_1)$\wedge$
Contains(m_0, HASH$_{mptk_{Z,X}}$($plc'_{Z,X}$, PULL1HASH))$\wedge$
Contains(m_1, HASH$_{mptk_{Z,X}}$($plc_{Z,X}$, PULL1HASH))$\wedge$
IsLess($plc'_{Y,T}, plc_{Y,T}$) $\supset$ Send(T, m_0) $<$ Send(T, m_1))

(1)

The Pull protocol is the protocol used by the retrieve() function to obtain a key if it is not cached locally. Thus, the Pull protocol can be run mid-protocol during either the MSA Authentication or during our Abbreviated Handshake we present later. Of course, it can also be run independently. The protocol consists of the MA sending a request for a key by name and the MKD delivering that key. The Pull protocol uses a counter in a similar fashion to the Group Key Handshake, which is maintained by the MA as he initiates the Pull protocol.

The Push protocol is a three message protocol, initiated by the MKD, to instruct the MA to update a key. The first messages is sent by the MKD and contains the name of the key to update. Once, the MA receives this message, he instantiates a Pull protocol using the key name delivered by the MKD. Thus the Push protocol could be thought of as a one message protocol, followed by the Pull protocol. However for ease of presentation we chose not to always separate the protocol this way. The first message of the Push protocol has a separate counter that is maintained by the MKD as the MKD initiates the Push protocol. However the second and third messages contain the counters of the Pull protocol.

The Delete protocol consists of the MKD sending the name of a key to delete, followed by an acknowledgement from the MA. The delete protocol uses the same counter as the Push protocol in both its messages.

The security goals of these protocols are mostly straightforward, however the authentication goal for the MKD:PUSH is not. For the the authentication goal we will consider PUSH as a one message protocol, followed by a two message protocol. It is fine to separate it in this way as this is an accurate representation of the protocol. Thus we will apply our generalized matching conversation definition to this set of two protocols. In the ideal world (without any adversaries) it is impossible to prove that the MA received PULL1. This is because there is nothing connecting the PUSH message sent with the next message sent in the system, PULL1. It is possible that the message PUSH1 was lost and the MA decided independently to run the Pull protocol. Thus we have no hopes of every proving the reception of PUSH1 from the point of view of the MKD. However, this is irrelevant, for as long as the MKD receives the PULL1 (which is fundamentally similar to a PUSH2) message after it sends the PUSH1 message, it can send the correct key to the MA. Thus our definition in this case proves to be sufficient. Also it is maximal if when we consider PUSH as a one message protocol followed by a two message protocol, which we believe was the intention of the designers of the MSA. We give the authentication goal for the MKD:PUSH below,

$$\begin{aligned}&\mathbf{\Phi_{PUSH,MA,MKD}} := \\ &\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\ &(\text{Send}(T, \text{PUSH1}) \wedge \text{Send}(Y, \text{PULL1})) < \\ &\text{Rcve}(T, PULL1) < \text{Send}(Y, PULL2).\end{aligned}$$

The Push, Pull, and Delete protocols also have similar timing invariants as the Group Key Handshake. These guarantee that the MA and the MKD will always be certain which messages were sent in which order. The proof of this invariant is nearly identical to the proof given in the Group Key Handshake and is thus omitted. Since we were able to show this invariant over all three protocols, the protocols do in fact compose arbitrarily with each other. Thus, there is no combination of steps, lost messages and restarts that will cause a message sent out after another message to have a lower counter value. Note that this is only the case if the basic sequences are implemented as autonomous steps as modeled by PCL. If the basic sequences are not autonomous steps than the ordering security invariant does not hold, but the rest of the proofs (mutual authentication, key secrecy, etc.) are still valid.

We present the formal requirement of Push, Pull and Delete below. We use the short hand PPD:MA to indicate we have independently proven the statement for the MA running each of the three protocols. We use similar notation for the MKD.

Theorem 6.

(i) $\Gamma_{PPD,\{1,2\}} \wedge \Gamma_{PPD,SI,\{1,2\}} \vdash$
$\Theta_{PPD,\{1,2\}}[\textbf{PPD:MA}]_X$
$\Phi_{PPD,\{AUTH,PTKD\},PPD}$

(ii) $\Gamma_{PPD,\{1,2\}} \wedge \Gamma_{PPD,SI,\{1,2\}} \vdash$
$\Theta_{PPD,\{1,2\}}[\textbf{PPD:MKD}]_T$
$\Phi_{PPD,AUTH,MKD}$

(iii)PPD $\vdash \Gamma_{PPD,\{1,2\}} \wedge \Gamma_{PPD,SI,\{1,2\}}$

6 Abbreviated Handshake

The Abbreviated Handshake is a protocol that is to be used in place of a MSA Authentication between a MA and a MP, so long as one of the parties is a MA. The 802.11s task group has received submissions interested in creating an Abbreviated Handshake [8, 25]. The idea is that it is a shorter protocol, as not as many steps are necessary when both parties have already joined the mesh. This allows for node mobility within the mesh, with minimal extra messages and bandwidth cost.

As in the MSA Authentication the parties will exchange information important to the successful completion of the protocol. This information is given in the form of $INFO$ and the select() function deterministically selects the correct option from the shared info. As in the Peer Link Establishment protocol, both principals may start communication at the same time. However, the protocols are not totally symmetric, for if both parties are MAs the decision as who will retrieve the key from the MKD is based on the relative values of the MAC addresses.

Also we found it unnecessary to require that both parities always act as if they are in the case where they have started a connection to each other at the same time (the SIMO case). When one party is obviously the initiator the messages exchanged are similar to the 802.11i Four-Way Handshake. The advantage of this protocol over the SIMO case is that it allows the opening party to always receive the last message. The reason for this design decision is that we imagine in most common situations the party who initiated the connection will be the first to initiate communication once the session is established. Thus a Four-Way Protocol allows the initiating party to confirm that the protocol was completed successfully. It is impossible for both parties to confirm the completion of the protocol as one party must send the last message and will never know if that message arrives (two army problem).

(Not) acknowledging the receipt of the *gtk*. In the Simultaneous open we do not confirm the receipt of the *gtk* at the peer as we believe it is unnecessary. We give our justification below.

1. It is given in 802.11i (see Section 8.4.8 in [1]) that a desired property of the Four-Way Handshake is to *transfer* the *gtk* from the authenticator to the

supplicant. The standard does not state confirming the reception of the *gtk* at the supplicant as a goal. We believe this choice of language was intentional. It seems natural to have the same requirement for 802.11s.

2. In MSA it is functionally meaningless for the party receiving the last message of a *ptk* establishment protocol to *acknowledge* the receipt of the *gtk* during the protocol. This stems from the restriction that a party is forbidden to use the *gtk* until the *ptk* protocol has successfully completed at that node. Assume that party A is the party receiving a last message and party B is the party sending a last message. Once party B has completed its role in the protocol, B can not know if A has received the last message. Thus upon completion of any *ptk* establishment protocol it is always unknown to the sender of the last message if the receiver of the last message is using the *gtk* (because it's unknown whether A is certainly using the *ptk*). Thus an acknowledgement of the *gtk* by the party receiving the last message is functionally meaningless. This is inherent as long as the *gtk* can not be installed until the completion of the protocol that establishes the *ptk*. It is functionally meaningful, however, to tie *gtk* installation (establishing a *gtk* session) to the completion of a *ptk* establishment protocol.

This issue is further complicated in the peer-to-peer network of 802.11s as both parties exchange *gtk*s during the *ptk* establishment protocol. Since there must always be one party that receives the last message of a protocol, there will always be one party who is unable to confirm the use of the *gtk* at the peer. Thus, if confirming the installation of the *gtk* at the peer is a necessary function of the network, a completely independent *gtk* confirmation protocol **must** be used after a successful *ptk* establishment protocol (like the abbreviated handshake). Acknowledging the reception of the *gtk* during the *ptk* establishment protocol will always be meaningless for one party. We note, though, that both parties can tie *gtk* usage to *ptk* usage – an honest node will only install the *gtk* if and only if it has also installed the *ptk*. And, of course, the *gtk* is like all other values and is verified to be correct.

Independent of our design decision, we feel that the distinction between the acknowledgement of the receipt of the *gtk* and the acknowledgement of the installation (the **use** of the *gtk*) is important to establish. This distinction should be useful to both the members of the 802.11s task group as well of those who make use of the system; without it those individuals may make design and protocol decisions assuming that principals within the system are indeed using the *gtk* when they are in fact not. However, *gtk* use can be tied to *ptk* use, so that other methods of determining *gtk* installation are possible.

Generalized Matching Conversations For ABBH:SIMO We cannot use the matching conversations definition from [2] for the ABBH:SIMO protocol. Thus, we apply the generalized matching conversations definition. Let principal X be the principal from whose view we are establishing the proof of generalized matching conversation and Y be the other principal. SIMO1X and SIMO5X will represent X's messages, similarly for Y's messages. We need to determine the maximal timing relations in the ideal world (no adversaries) only running ABBH:SIMO. As X cannot confirm if Y has received SIMO5X (it may be the last message sent), SIMO5X is not part of X's maximal conversation. Note that every message must be sent by the correct party before they are received by the other party in this ideal world. Thus we get Send(X,msg)<Receive(Y,msg) and Send(Y,msg)<Receive(X,msg) for every message. Now we simply list what actions must happen after other actions leaving out the trivial receives after sends we showed above and other redundant information.

$$\begin{aligned}&\text{Send}(X, \text{SIMO1X}) < \text{Receive}(X, \text{SIMO1Y}) < \\ &\quad(\text{Send}(X, \text{SIMO5X}) \wedge \text{Receive}(X, \text{SIMO5Y}))\\ &\text{Send}(Y, \text{SIMO1Y}) < \text{Receive}(Y, \text{SIMO1X}) < \\ &\quad\text{Send}(Y, \text{SIMO5Y})\end{aligned}$$

This temporal ordering is inherently maximal for X's view of an arbitrary run of the ABBH:SIMO protocol, so it satisfies the definition of generalized matching conversations for X (Y's view will be identical). The enforcing of send orders within one node can be accomplished by waiting for acknowledgements from the MAC layer before proceeding. We enforce that X has sent its message 1 before receiving a message 1 to separate the SIMO case from the INIT/RESP case. If a node X has not sent its message 1, it replies to a message 1 with a message 2.

We present the strands, preconditions, invariants and security goals in the SIMO case below. One subtlety of the SIMO strand is the ordering of the nonces x and y when creating the *ptk*. Other messages have clear initiators and responders, so the ordering can be determined from that. For SIMO, the ordering is dependent on the values of the nonces themselves, with the smaller nonce ordered first. This could also be applied to 4WAY without changing the validity of the corresponding proofs. This might be simpler from an implementation and standardizations perspective. The remaining security goals have been moved to the appendix in Section F. We once again use the retrieve function, thus we assume the relevant preconditions and postcondition associated with the basic sequences surrounding the function.

ABBH:INIT $= (X, \hat{Y}, INFO_X, gtk_X)$
[new x; send $\hat{Y}, \hat{X}$, "ABBH1", $INFO_X, x$;
rcve $\hat{X}, \hat{Y}$, "ABBH2", $INFO_Y, y, x, enc_0, mic_0$;
mtch $SELECT(INFO_X, INFO_Y)/CS, pmkN$;
mtch $RETRIEVE(pmkN)/pmk$;
mtch $HASH_{pmk}(x, y)/ptk_{X,Y}$;
mtch $enc_0/\mathrm{ENC}_{ptk_{X,Y}}(gtk_Y)$;
mtch $mic_0/\mathrm{HASH}_{ptk_{X,Y}}($
    $\hat{X}, \hat{Y}$, "ABBH2", $INFO_Y, y, x, enc_0)$;
mtch $enc_1/\mathrm{ENC}_{ptk_{X,Y}}(gtk_X)$;
mtch $mic_1/\mathrm{HASH}_{ptk_{X,Y}}(\hat{Y}, \hat{X}$, "ABBH3",
    $INFO_X, x, y, enc_1)$;
send $\hat{Y}, \hat{X}$, "ABBH3", $INFO_X, x, y, enc_1, mic_1$;
rcve $\hat{X}, \hat{Y}$, "ABBH4", y, x, mic_2;
mtch $mic_2/\mathrm{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y}$, "ABBH4", $y, x)]_X$

ABBH:RESP $= (Y, INFO_Y, gtk_Y)$
[rcve $\hat{Y}, \hat{X}$, "ABBH1", $INFO_X, x$;
mtch $SELECT(INFO_X, INFO_Y)/CS, pmkN$;
mtch $RETRIEVE(pmkN)/pmk$;
new y; mtch $HASH_{pmk}(x, y)/ptk_{X,Y}$;
mtch $\mathrm{ENC}_{ptk_{X,Y}}(gtk_Y)/enc_0$;
mtch $\mathrm{HASH}_{ptk_{X,Y}}($
    $\hat{X}, \hat{Y}$, "ABBH2", $INFO_Y, y, x, enc_0)/mic_0$;
send $\hat{X}, \hat{Y}$, "ABBH2", $INFO_Y, y, x, enc_0, mic_0$;
rcve $\hat{Y}, \hat{X}$, "ABBH3", $INFO_X, x, y, enc_1, mic_1$;
mtch $enc_1/\mathrm{ENC}_{ptk_{X,Y}}(gtk_X)$;
mtch $mic_1/\mathrm{HASH}_{ptk_{X,Y}}(\hat{Y}, \hat{X}$, "ABBH3",
    $INFO_X, x, y, enc_1)$;
mtch $mic_2/\mathrm{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y}$, "ABBH4", $y, x)$;
send $\hat{X}, \hat{Y}$, "ABBH4", $y, x, mic_2]_Y$

ABBH:SIMO $= (X, \hat{Y}, INFO_X, gtk_X)$
[new x; send $\hat{Y}, \hat{X}$, "ABBH1", $INFO_X, x$;
rcve $\hat{X}, \hat{Y}$, "ABBH1", $INFO_Y, y$;
mtch $SELECT(INFO_X, INFO_Y)/CS, pmkN$;
mtch $RETRIEVE(pmkN)/pmk$;
mtch $HASH_{pmk}(x, y)/ptk_{X,Y}$;
(mtch $\mathrm{ENC}_{ptk_{X,Y}}(gtk_X)/enc_0$;
mtch $\mathrm{HASH}_{ptk_{X,Y}}(\hat{Y}, \hat{X}$,
    "ABBH5", $INFO_X, x, y, enc_0, INFO_Y)/mic_0$;
send $\hat{Y}, \hat{X}$, "ABBH5", $INFO_X, x, y, enc_0, mic_0)$:
(rcve $\hat{X}, \hat{Y}$, "ABBH5", $INFO_Y, y, x, enc_1, mic_1$;
mtch $enc_1/\mathrm{ENC}_{ptk_{X,Y}}(gtk_Y)$;
mtch $mic_1/\mathrm{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y}$,
    "ABBH5", $INFO_Y, y, x, enc_1, INFO_X))]_X$

$\mathbf{\Theta_{ABBH,1}} := \mathrm{Has}(X, pmk_{X,Y}) \wedge \mathrm{Has}(Y, pmk_{Y,X}) \wedge$
$(\mathrm{Has}(X, mptk_{X,T}) \vee \mathrm{Has}(Y, mptk_{X,T}))$

$\mathbf{\Gamma_{ABBH,1}} := \mathrm{Honest}(\hat{X}) \wedge \mathrm{Send}(X, m) \wedge$
$(\mathrm{Contains}(m, \mathrm{Hash}_{ptk}((\text{"ABBH2"}, \hat{Y}, \hat{Z}))) \vee$
$\mathrm{Contains}(m, \mathrm{Hash}_{ptk}((\text{"ABBH3"}, \hat{Y}, \hat{Z}))) \vee$
$\mathrm{Contains}(m, \mathrm{Hash}_{ptk}((\text{"ABBH4"}, \hat{Y}, \hat{Z}))) \vee$
$\mathrm{Contains}(m, \mathrm{Hash}_{ptk}((\text{"ABBH5"}, \hat{Y}, \hat{Z}))) \vee$
$\hat{Z} = \hat{X}$

$\mathbf{\Phi_{SIMO,MA}} :=$
$\mathrm{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$(\mathrm{Send}(X, \mathrm{SIMO1X}) < \mathrm{Rcve}(Y, \mathrm{SIMO1X})) \wedge$
$(\mathrm{Send}(Y, \mathrm{SIMO1Y}) < \mathrm{Rcve}(X, \mathrm{SIMO1Y})) \wedge$
$(\mathrm{Send}(Y, \mathrm{SIMO5Y}) < \mathrm{Rcve}(X, \mathrm{SIMO5Y})) \wedge$
$(\mathrm{Send}(X, \mathrm{SIMO1X}) < \mathrm{Rcve}(X, \mathrm{SIMO1Y}) <$
$(\mathrm{Send}(X, \mathrm{SIMO5X}) \wedge \mathrm{Rcve}(X, \mathrm{SIMO5Y})) \wedge$
$(\mathrm{Send}(Y, \mathrm{SIMO1Y}) < \mathrm{Rcve}(Y, \mathrm{SIMO1X}) <$
$\mathrm{Send}(Y, \mathrm{SIMO5Y}))$

$\mathbf{\Phi_{SIMO,PTKD}} :=$
$\mathrm{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\mathrm{Has}(X, ptk_{X,Y}) \wedge \mathrm{Has}(Y, ptk_{X,Y})$

$\mathbf{\Phi_{SIMO,GTKD}} :=$
$\mathrm{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
$\mathrm{Rcve}(Y, SIMOX5) \supset$
$\mathrm{Has}(X, gtk_Y) \wedge \mathrm{Has}(Y, gtk_X)$

$\mathbf{\Phi_{SIMO,KF}} :=$
$\mathrm{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$(\mathrm{new}\ (\hat{X}, x) \wedge x \subseteq ptk_{X,Y} \wedge \mathrm{new}\ (\hat{Y}, y) \wedge y \subseteq ptk_{X,Y}) \wedge$
$\mathrm{FirstSend}(X, x, \hat{X}, x, \mathrm{SIMO1X}) \wedge$
$\mathrm{FirstSend}(Y, y, \hat{Y}, y, \mathrm{SIMO1Y})$

$\mathbf{\Phi_{SIMO,INFO}} :=$
$\mathrm{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$SELECT(INFO_X, INFO_Y) = CS, pmkN \wedge$
$\mathrm{Has}(X, CS, pmkN) \wedge \mathrm{Has}(Y, CS, pmkN)$

We give the main theorem of this section below,

Theorem 7.

(i) $\Gamma_{ABBH,1} \wedge \Gamma_{ABBH,SI,\{1,2\}} \vdash$
$\Theta_{ABBH,1}[\textbf{ABBH:INIT}]_X$
$\Phi_{ABBH,\{AUTH,PTKD,GTKD,KF,INFO\},INIT}$

(ii) $\Gamma_{ABBH,1} \wedge \Gamma_{ABBH,SI,\{1,2\}} \vdash$
$\Theta_{ABBH,1}[\textbf{ABBH:RESP}]_Y$
$\Phi_{ABBH,\{AUTH,PTKD,GTKD,KF,INFO\},RESP}$

(iii)$\Gamma_{ABBH,1} \wedge \Gamma_{ABBH,SI,\{1,2\}} \vdash$
$\Theta_{ABBH,1}[\textbf{ABBH:SIMO}]_X$
$\Phi_{ABBH,\{AUTH,PTKD,GTKD,KF,SIMO\},SIMO}$

(iv)ABBH $\vdash \Gamma_{ABBH,\{1,2\}} \wedge \Gamma_{ABBH,SI,\{1,2\}}$

We omit the proof of ABBH:INIT and ABBH:RESP because it is nearly identical to the proof of the Four-

Way Handshake. The desired security goals are also nearly identical, proving mutual authentication, key freshness, session key possession, GTK transfer, and information selection. The proof of the ABBH:SIMO security goals is given in the appendix in section F. We walk through the generalized matching conversations proof below.

Proof Sketch Generalized Authentication, SIMO We only need to show the proof from a single point of view as the roles are symmetric. Let principal X be the principal from whose view we are establishing the proof from and let Y be the other principal. As the proof assumes X has completed the protocol successfully, we know that SIMO1X was sent before SIMO5X and SIMO1Y was received before SIMO5Y. Thus to complete the proof we must show that Y sent exactly SIMO1Y before SIMO5Y and received exactly SIMO1X before sending SIMO5Y. As in previous proofs, we can determine the MIC in SIMO5Y could have only been sent by Y if X, Y and T are honest. Since all the variables used in the protocol are contained in the MIC of SIMO5Y, we know that X and Y share identical variables. Now using the honesty of Y we are sure that Y sent SIMO1Y and received SIMO1X before sending SIMO5Y and that it was sent exactly as X received it. Again if Y is honest since X and Y share variables, then Y must have received SIMO1X exactly as X had sent it. This gives us generalized authentication.

7 Proof of Key Hierarchy

This section is presented here, for ease of exposition. In reality, a lot of the effort of proving these results had to be done before the proofs of the correctness of the protocols. None of the results in this section rely on any any previous results. But they do rely on the protocol definitions given in section 5 and 6. Indeed, those protocol definitions are critical for this section.

We use the SafeNet construction of [23] to prove these results. At its core, SafeNet shows that particular values are always "protected" by a set of keys, that is, no adversary without a key from that particular set of keys will not be able to derive the particular value. This allows us to reason about the possession of that value based on who has access to particular keys. The main theorem of this section shows that all protocols within the MSA definition satisfy the presented security invariants.

We do add one axiom to the SafeNet structure. We claim new Axiom **SAF5** as: SafeMsg($HASH_s(M), s, \mathcal{K}$). This follows from the fact that using information as the key part of a keyed hash does not reveal any information about that information. This is almost a direct consequence of **SAF4** (which states SafeMsg($HASH(M), s, \mathcal{K}$)). Depending on how the key is used in the hash, they could even be equivalent, but we prefer clarity with a new axiom.

Theorem 8. *Let MSA represent all the protocols in the Mesh Security Architecture and $\Gamma_{SI,ALL}$ represent all the security invariants in figure 3. Then $\Gamma_{SI,ALL}$ are invariants of MSA. Formally,*

$$MSA \vdash \Gamma_{SI,ALL}$$

Proof sketch: This theorem is proven in two steps. The first step is a massive induction over all the basic sequences of all the protocols that could be run by any participant in a mesh. This induction guarantees that all messages sent are "safe", in that critical information is protected by the listed keys. In the MSA case, the critical information is another key, lower in the hierarchy. From this, we argue the invariant nature of the SafeNet over MSA. Then, we use the **POS** and **POSL** axioms [23] to state who can potentially have access to the critical information (that is, various other keys). By proceeding in this way throughout the entire key hierarchy, we can establish all the necessary security invariants. We give a flavor of this type of induction in Section G of the appendix. The full proof is generally unenlightening and we do not provide it. We note that this proof does not depend on any of the analysis done in proceeding sections. It is simply induction over all basic sequences and application of secrecy axioms.

8 Composition

The MSA architecture allows for significant variation in how protocols compose together [4]. Once an established state is reached, many protocols (which may have been run previously to reach the established state) may be chosen. Reaching an established state may take a variety of paths, depending on the authentication mechanism (TLS or PSK) used. Error-handling strategies will cause protocols to restart, or, potentially, different protocols to be run. This introduces a complex state diagram and complexities of composition.

While staged composition proofs have been presented [22, 20], the presentation of each has differed. We provide a slightly different presentation of similar ideas in Section 8.1. Readers primarily interested in the proof of MSA can safely skip this section and proceed to Section 8.2 where the overall MSA security theorem is presented.

8.1 Consistent Composition

We utilize the definitions of role-prefix, staged role, and staged composition from [20]. Additionally, to add simplicity to our exposition, we use Γ to denote the conjunction of all invariants within a staged composition of protocols. That is, Γ is the totality of all the invariants from each of the protocols Q_i that make up a composition of protocols Q. This will allow us to state the following theorem more succinctly.

Theorem 9. *Let Q be a staged composition of protocols* $Q_1, Q_2, ..., Q_n$. *Then* $\vdash_Q$ *KOHonest* $\supset \Phi$, *if for all* $RComp(\langle P_1, P_2, \ldots, P_n\rangle) \in Q$, *all of the following hold:*

(Invariant Induction)

(i) $\forall i.\forall S \in BS(Q_i). \vdash \Theta_{P_i} \wedge \Gamma[S]_X\Gamma$

(Precondition Inductions)

(ii) $Q_1 \otimes Q_2 \otimes \cdots \otimes Q_n \vdash Start(X) \supset \Theta_{P_1}$

(iii)$Q_1 \otimes Q_2 \otimes \cdots \otimes Q_n \vdash \forall i.\Theta_{P_i}[P_i]_X\Theta_{P_{i+1}}$

(iv) $\forall i.\forall S \in \bigcup_{j\geq i} BS(P_j).\Theta_{P_i}[S]_X\Theta_{P_i}$

The invariant induction step fundamentally states that every basic sequence that could be reached by a run through Q maintains the full invariant list (Γ), as long as that's step's preconditions hold. The precondition induction rules enforce three facts. The first states that the start of protocol Q causes the base precondition to hold. This starts the induction. The second states that each step passes from its preconditions to the next step's preconditions. Note that step P_i has postcondition Φ_i, so this step is saying that $\Phi_i \supset \Theta_{P_{i+1}}$, to guarantee the progress moving forward. The third precondition invariant allows the backwards steps, by guaranteeing that no step ever violates any step's preconditions (that is, once something is true, it's true forever).

We note that this theorem is dependent upon basic sequences, as its fundamental building block. The protocols themselves, while useful distinctions in understanding and modeling the system, are not critical. In particular, the Q_is could be single basic sequences and the entire theorem continues to make sense. This allows us to model things at the basic sequence level. This level of granularity has been suggested before [20], but we make it explicit.

This allows, for example, the behavior of the $RETRIEVE$ description. As its own basic sequence, $RETRIEVE$ allows two different paths through a larger staged composition. In one path, the basic sequence simply returns a locally stored value. In the other path, an entire protocol might be run. Since that protocol composes consistently at basic sequence breaks in the initial protocol, it simply denotes an alternate method of staging the composition of the protocols. It is important, however, that the $RETRIEVE$ action always be treated as its own basic sequence. This provides the necessary preconditions at the basic sequence level. Inductively proving actions without proper consideration of $RETRIEVE$ would lead to false results. In all protocols using $RETRIEVE$, the invariants and various preconditions in the protocol were proven against all possible interpretations of $RETRIEVE$.

8.2 Composition in MSA

We wish to apply Theorem 9 to the protocols of the MSA proposal. We view the protocols of staged composition as the protocols given previously. As mentioned, we consider arbitrary breaks at the basic sequence level, for mid-protocol composition as well as overall composition. We need to prove that all protocols within MSA (comprising PLE, TLS, 4WAY, MKHSH, GKH, PULL, PUSH, DEL, ABBH, SIMO) satisfy the necessary conditions for composition.

Theorem 10. *Let Q be a specific composition of protocols from MSA and* $RComp(\langle P_1, P_2, \ldots, P_n\rangle) \in Q$ *and* $\Gamma = \Gamma_{TLS,SI,\{1,2\}} \wedge \Gamma_{4WAY,SI,\{1,2,3,4\}} \wedge \Gamma_{PPD,SI,\{1,2\}} \wedge \Gamma_{MKHSH,SI,1} \wedge \Gamma_{ABBH,SI,\{1,2\}} \wedge \Gamma_{GKH,SI,\{1,2\}} \wedge \Gamma_{TLS,\{1,2\}} \wedge \Gamma_{4WAY,1} \wedge \Gamma_{MKHSH,1} \wedge \Gamma_{GKH,\{1,2\}} \wedge \Gamma_{PPD,\{1,2\}} \wedge \Gamma_{ABBH,1}$. *Then:*

(i) $\forall i.\forall S \in BS(Q_i). \vdash \Theta_{P_i} \wedge \Gamma[S]_X\Gamma$

(ii) Θ_{P_1}

(iii)$\Phi_{4WAY} \vdash \Theta_{MKHSH} \wedge \Theta_{GKH}$

$\Phi_{MKHSH} \vdash \Theta_{PUSH} \wedge \Theta_{PULL} \wedge \Theta_{DEL}$

$\Phi_{MKHSH} \vdash \Theta_{ABBH} \wedge \Theta_{SIMO}$

$\Phi_{ABBH} \vdash \Theta_{GKH}$

(iv) $\forall i.\forall S \in \bigcup_{j\geq i} BS(P_j).\Theta_{P_i}[S]_X\Theta_{P_i}$

Proving all the protocols securely compose is a lengthy induction process, which we omit for space purposes. We will give brief justifications of each step without formal mention of the details.

First, we prove that invariant induction holds across all the protocols. In previous sections, we proved each particular protocol's invariants hold across all its basic sequences. Since nearly all the invariants are tied to specific protocol messages, most of these are extremely straightforward. Those which are not so tied are the TLS invariants $\Gamma_{TLS,1}$ and $\Gamma_{TLS,2}$ and the two ordering invariants $\Gamma_{GKH,2}$ and $\Gamma_{PPD,2}$. Since the TLS invariants deal with public-key operations not used by other protocols, they, too, are straightforward. The ordering invariants involve counters specific to the GKH

protocol and the *PUSH*, *PULL*, and *DEL* protocols respectively. Those invariants were proven across those counters in their respective sections and no other protocol uses those counters, so the invariants are not violated.

Second, we prove that the induction starts correctly, that the base preconditions are held at the start of a session at any node X. This is a precondition assumption about information available (either public-key credentials or a PSK). This precondition is inherent in the definition of the start of the protocol.

Third, we prove that each step's postconditions match the preconditions of following steps. We have been noting this throughout the paper. Specifically, where multiple paths lead to the same condition, we have verified that the necessary preconditions are all met for any stage. Since some initial handshake (comprising PLE, TLS, 4WAY or PLE, 4WAY with PSK) must occur first, we show that 4WAY implies the ability to participate in fuller mesh protocols (as a mesh point – including MKHSH and GKH). Note that at this point, a node could attempt ABBH or SIMO, but it is not guaranteed successfully complete, even in the presence of no adversary, as one side or the other must have the correct PMK. Completing MKHSH makes a node a Mesh Authenticator (MA) and guarantees a full variety of protocols (PUSH, PULL, DEL, ABBH, SIMO).

Finally, we prove that no step (or combination of steps) in any protocol violates any precondition. All preconditions are expressed expressly in terms of Has, Send, and Receive. Once these states are true, they do not change, from a security perspective. So, the induction to prove the continuing truth of all preconditions is pretty trivial.

We have now shown that we can apply Theorem 9 to the protocols presented in this paper. In particular, we have that any reasonable progression through protocols is secure. In particular, we have shown the following theorem holds.

Theorem 11. *A mesh of nodes, all of which conform to the proposed specification of MSA (and the minor modifications of this document), guarantee all the listed security goals of this paper.*

More can be said, too. It isn't that the whole mesh must comply. As long as the players in a given protocol are honest (and the MKD is honest), the security of that protocol is ensured. Fundamentally, the Mesh Security Architecture is sound.

9 Conclusions and Future Work

We have proven the security of the MSA, under standard assumptions. We have provided and justified a few recommendations which we hope will be implemented in the final draft of the standard, if this architecture is chosen. We also hope that providing a security proof during the design and review process will lead to additional efforts in that regard. We feel that protocol design is important and an analysis of a system should be done before implementation, not after. Along the way, we made a number of contributions to PCL.

The most important, from our perspective, is the ability to handle simultaneity, with the introduction of action groups and associated axioms and proof techniques. The use of SafeNet and other axioms to handle an entire key hierarchy is also novel, but it is a good application of existing techniques in an interesting way. The return(), select(), and retrieve() actions were also designed to extend naturally to examinations of other architectures.

The importance of proving security as invariants instead of simply as end goals was introduced in [23], but the importance of it was not stressed. We feel making this implicit difference explicit should allow for clearer and better security proofs in the future. Some security goals are rightly protocol postconditions. Others must be verified throughout the run of a protocol.

This paper also takes a deeper dive into the details of the protocols than is often undertaken. While examining only the security components (nonces, keys, etc.) simplifies analysis, it also leaves a gap. Our experience leads us to believe that gaps in analysis are often dangerous, as they lead to assumptions about security, implementation difficulties, and unforeseen attack vectors. Some level of abstraction is necessary, but adding a model for authenticated information exchange is critical for many applications.

This paper opens opportunities for applying PCL to other peer-to-peer protocols, where ordering may not be as strict as in server-client models. Other protocol systems, particularly those on standard-track, would be natural candidates for additional analysis.

The language of PCL also has some avenues for further development. The concept of breaking basic sequences was introduced in this paper. The natural question arises of what might happen if basic sequences were not autonomous units but could be broken. This change would invalidate existing PCL theorems, but would close certain potential loopholes. Combining this with pre- and post-conditions at a smaller granularity could potentially lead to additional breakthroughs in the area of proving protocol correctness.

References

[1] IEEE standard 802.11-2007. local and metropolitan area networks – specific requirements – part 11: Wireless LAN medium access control and physical layer specifications.

[2] M. Bellare and P. Rogaway. Entity authentication and key distribution. In *CRYPTO*, pages 232–249, 1993.

[3] N. Borisov, I. Goldberg, and D. Wagner. Intercepting mobile communications: The insecurity of 802.11. http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html, 2001.

[4] T. Braskich and S. Emeott. Clarification and update of MSA overview and MKD functionality text. https://mentor.ieee.org/802.11/documents doc 11-07/2119r1, July 18 2007.

[5] T. Braskich and S. Emeott. Initial MSA comment resolution. https://mentor.ieee.org/802.11/documents doc 11-07/0564r2, May 16 2007.

[6] T. Braskich and S. Emeott. Key distribution for MSA comment resolution. https://mentor.ieee.org/802.11/documents doc 11-07/0618r0, May 14 2007.

[7] T. Braskich and S. Emeott. Mesh key holder protocol improvements. https://mentor.ieee.org/802.11/documents doc 11-07/1987r1, June 27 2007.

[8] T. Braskich, S. Emeott, and D. Kuhlman. Security requirements for an abbreviated msa handshake. https://mentor.ieee.org/802.11/documents doc 11-07/0770r0, May 15 2007.

[9] A. Datta, A. Derek, J.C.Mitchell, and B.Warinschi. Computationally sound compositional logic for key exchange protocols. In *Proceedings of 19th IEEE Computer Security Foundations Workshop*, pages 321–334, 2006.

[10] A. Datta, A. Derek, J. Mitchell, and D. Pavlovic. A derivation system for security protocols and its logical formalization, 2003.

[11] A. Datta, A. Derek, J. C. Mitchell, and D. Pavlovic. Secure protocol composition. In *FMSE*, pages 11–23, 2003.

[12] A. Datta, A. Derek, J. C. Mitchell, and D. Pavlovic. A derivation system and compositional logic for security protocols. *J. Comput. Secur.*, 13(3):423–482, 2005.

[13] A. Datta, A. Derek, J. C. Mitchell, and A. Roy. Protocol composition logic (pcl). *Electr. Notes Theor. Comput. Sci.*, 172:311–358, 2007.

[14] A. Datta, J. Mitchell, F. Muller, and D. Pavlovic. Authentication for mobile ipv, 2002.

[15] W. Diffie, P. C. van Oorschot, and M. J. Wiener. Authentication and authenticated key exchanges. *Des. Codes Cryptography*, 2(2):107–125, 1992.

[16] N. Durgin, J. Mitchell, and D. Pavlovic. A compositional logic for proving security properties of protocols, 2002.

[17] N. Durgin, J. Mitchell, and D. Pavlovic. A compositional logic for proving security properties of protocols. *J. Comput. Secur.*, 11(4):677–721, 2004.

[18] D. Gollmann. What do we mean by entity authentication? In *SP '96: Proceedings of the 1996 IEEE Symposium on Security and Privacy*, page 46, Washington, DC, USA, 1996. IEEE Computer Society.

[19] J. Haasz and S. Hampton. Amendment: Mesh networking. http://standards.ieee.org/board/nes/projects/802-11s.pdf, 2006.

[20] C. He, M. Sundararajan, A. Datta, A. Derek, and J. C. Mitchell. A modular correctness proof of ieee 802.11i and tls. In *ACM Conference on Computer and Communications Security*, pages 2–15, 2005.

[21] H. Krawczyk. Sigma: The 'sign-and-mac' approach to authenticated diffie-hellman and its use in the ike-protocols. In *CRYPTO*, pages 400–425, 2003.

[22] A. Roy, A. Datta, A. Derek, and J. C. Mitchell. Inductive proof method for computational secrecy. 2006.

[23] A. Roy, A. Datta, A. Derek, J. C. Mitchell, and J-P. Seifert. Secrecy analysis in protocol composition logic. In *Proceedings of 11th Annual Asian Computing Science Conference*, 2006.

[24] J. Walker. Unsafe at any key size; an analysis of the wep encapsulation, 2000.

[25] M. Zhao, J. Walker, and W. Steven Conner. Overview of abbreviated handshake protocol. https://mentor.ieee.org/802.11/documents doc 11-07/1998r01, June 14 2007.

Appendix

A Proof of Security Goals of TLS

A.1 Proof of Security for the Client

Matching Conversations, Client:

$\mathbf{AA1}, \mathbf{ARP}, \mathbf{AA4}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{Send}(X, \hat{X}, \hat{T}, Nx, VerSU_x) <\\ \text{Rcve}(X, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) <\\ \text{Send}(X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) <\\ \text{Rcve}(X, \hat{T}, \hat{X}, hash_3)\end{array} \tag{2}$$

$\mathbf{ARP}, \mathbf{HASH3}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{Rcve}(X, \hat{T}, \hat{X}, hash_3) \supset\\ \exists Z.\text{Computes}(Z, \text{Hash}_{xxKey_X}(handShake2, \text{“server”}))\wedge\\ \text{Send}(Z, \text{Hash}_{xxKey_X}(handShake2, \text{“server”}))\wedge\\ (\text{Send}(Z, \text{Hash}_{xxKey_X}(handShake2, \text{“server”})) <\\ \text{Rcve}(X, \hat{T}, \hat{X}, hash_3))\end{array} \tag{3}$$

$\mathbf{\Gamma_{TLS,SI,1}}, \mathbf{HASH1}$

$$\begin{array}{l}\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset\\ \text{Computes}(Z, \text{Hash}_{xxKey_X}(handShake2, \text{“server”})) \supset\\ \text{Has}(Z, xxKey_X) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}\end{array} \tag{4}$$

$\mathbf{3}, \mathbf{4}, \mathbf{AA1}, \mathbf{\Gamma_{TLS,2}}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset\\ \text{Send}(Z, \text{Hash}_{xxKey_X}(\hat{Y}, \hat{T}, Ny, VerSU_y,\\ \quad \hat{T}, \hat{Y}, Nz, VerSU_t, cert_1,\\ \quad \hat{Y}, \hat{T}, cert_2, sig_2, enc_2, hash_2, \hat{T}, \hat{Y}, \text{“server”})) \supset\\ \hat{Z} = \hat{T}\end{array} \tag{5}$$

$\mathbf{3}, \mathbf{5}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset\\ \text{Computes}(T, \text{Hash}_{xxKey_X}(handShake2, \text{“server”}))\end{array} \tag{6}$$

$\mathbf{6}, \mathbf{HASH1}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{Has}(T, xxKey_X) \wedge \text{Has}(T, handShake2)\end{array} \tag{7}$$

$\mathbf{6}, \mathbf{7}, \mathbf{\Gamma_{TLS,2}}, \mathbf{\Phi_{HONESTY}}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset\\ \text{Rcve}(T, \hat{X}, \hat{T}, Nx, VerSU_x) <\\ \text{Send}(T, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) <\\ \text{Rcve}(T, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) <\\ \text{Send}(T, \hat{T}, \hat{X}, hash_3)\end{array} \tag{8}$$

$\mathbf{3}, \mathbf{8}\Theta_{TLS}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset\\ \text{Send}(T, \hat{T}, \hat{X}, hash_3) <\\ \text{Rcve}(X, \hat{T}, \hat{X}, hash_3)\end{array} \tag{9}$$

$\mathbf{FS1}, \mathbf{AN3}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{FirstSend}(X, xxKey_X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2)\end{array} \tag{10}$$

$\mathbf{8}, \mathbf{10}, \mathbf{FS2}\Theta_{TLS}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{Send}(X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) <\\ \text{Rcve}(T, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2)\end{array} \tag{11}$$

$\mathbf{FS1}, \mathbf{AN3}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{FirstSend}(X, Nx, \hat{X}, \hat{T}, Nx, VerSU_x)\end{array} \tag{12}$$

$\mathbf{8}, \mathbf{12}, \mathbf{FS2}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{Send}(X, \hat{X}, \hat{T}, Nx, VerSU_x) <\\ \text{Rcve}(T, \hat{X}, \hat{T}, Nx, VerSU_x)\end{array} \tag{13}$$

$\mathbf{FS1}, \mathbf{AN3}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{Honest}(\hat{T}) \supset \text{FirstSend}(T, Nt, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1)\end{array} \tag{14}$$

$\mathbf{8}, \mathbf{14}, \mathbf{FS2}\Theta_{TLS}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{Honest}(\hat{T}) \supset \text{Send}(T, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) <\\ \text{Rcve}(X, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1)\end{array} \tag{15}$$

$\mathbf{8}, \mathbf{9}, \mathbf{11}, \mathbf{13}, \mathbf{15}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset\\ \text{Send}(X, \hat{X}, \hat{T}, Nx, VerSU_x) <\\ \text{Rcve}(T, \hat{X}, \hat{T}, Nx, VerSU_x) <\\ \text{Send}(T, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) <\\ \text{Rcve}(X, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) <\\ \text{Send}(X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) <\\ \text{Rcve}(T, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) <\\ \text{Send}(T, \hat{T}, \hat{X}, hash_3) <\\ \text{Rcve}(X, \hat{T}, \hat{X}, hash_3)\end{array} \tag{16}$$

Key Delivery, Client:

$\mathbf{AA1}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{Has}(X, xxKey_X)\end{array} \tag{17}$$

$\mathbf{7}, \mathbf{17}, \mathbf{\Theta_{TLS}}$

$$\begin{array}{l}[\mathbf{TLS}:\mathbf{CLNT}]_X\\ \text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset\\ \text{Has}(X, xxKey_X) \wedge \text{Has}(T, xxKey_X)\end{array} \tag{18}$$

A.2 Proof of Security Goals for the Server

Security Goals of the Server:

$$\begin{array}{l}
\mathbf{\Phi_{TLS,MA,SRVR}} := \\
\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset \\
\text{Send}(X, \hat{X}, \hat{T}, Nx, VerSU_x) < \\
\text{Rcve}(T, \hat{X}, \hat{T}, Nx, VerSU_x) < \\
\text{Send}(T, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) < \\
\text{Rcve}(X, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) < \\
\text{Send}(X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) < \\
\text{Rcve}(T, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) < \\
\text{Send}(T, \hat{T}, \hat{X}, hash_3)
\end{array}$$

$$\begin{array}{l}
\mathbf{\Phi_{TLS,KD,SRVR}} := \\
\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset \\
\text{Has}(\hat{X}, xxKey_X) \wedge \text{Has}(\hat{T}, xxKey_X)
\end{array}$$

Matching Conversations, Server:

$$\begin{array}{l}
\mathbf{AA1, ARP, AA4, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{Rcve}(T, \hat{X}, \hat{T}, Nx, VerSU_x) < \\
\text{Send}(T, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) < \\
\text{Rcve}(T, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) < \\
\text{Send}(T, \hat{T}, \hat{X}, hash_3)
\end{array} \tag{19}$$

$$\begin{array}{l}
\mathbf{ARP, VER, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{Rcve}(X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) \supset \\
\exists Z.\text{Computes}(Z, SIG_{priv_X}(handShake1)) \wedge \\
\text{Send}(Z, SIG_{priv_X}(handShake1)) \wedge \\
(\text{Send}(Z, SIG_{priv_X}(handShake1)) < \\
\text{Rcve}(T, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2))
\end{array} \tag{20}$$

$$\begin{array}{l}
\mathbf{\Gamma_{TLS,SI,1}, HASH1} \\
\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset \\
\text{Computes}(Z, SIG_{priv_X}(handShake1)) \supset \\
\text{Has}(Z, priv_X) \supset \hat{Z} = \hat{X}
\end{array} \tag{21}$$

$$\begin{array}{l}
\mathbf{21, \Gamma_{TLS,1}, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset \\
\text{Computes}(X, SIG_{priv_X}(handShake1))
\end{array} \tag{22}$$

$$\begin{array}{l}
\mathbf{22, SIG1, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{Has}(X, priv_X) \wedge \text{Has}(X, handShake1)
\end{array} \tag{23}$$

$$\begin{array}{l}
\mathbf{22, 23, \Gamma_{TLS,1}, \Phi_{HONESTY}, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_X \\
\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset \\
\text{Send}(X, \hat{X}, \hat{T}, Nx, VerSU_x) < \\
\text{Rcve}(X, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) < \\
\text{Send}(X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2)
\end{array} \tag{24}$$

$$\begin{array}{l}
\mathbf{20, 24, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset \\
\text{Send}(X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) < \\
\text{Rcve}(T, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2)
\end{array} \tag{25}$$

$$\begin{array}{l}
\mathbf{FS1, AN3, \Theta_{TLS}} \\
[\text{new } Nx;\ \text{send } \hat{X}, \hat{T}, Nx, VerSU_x]_T \\
\text{FirstSend}(T, Nt, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1)
\end{array} \tag{26}$$

$$\begin{array}{l}
\mathbf{24, 26, FS2, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{Send}(T, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) < \\
\text{Rcve}(X, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1)
\end{array} \tag{27}$$

$$\begin{array}{l}
\mathbf{FS1, AN3, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{Honest}(\hat{X}) \supset \text{FirstSend}(X, Nx, \hat{X}, \hat{T}, Nx, VerSU_x)
\end{array} \tag{28}$$

$$\begin{array}{l}
\mathbf{24, 28, FS2, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{Honest}(\hat{X}) \supset \text{Send}(X, \hat{X}, \hat{T}, Nx, VerSU_x) < \\
\text{Rcve}(T, \hat{X}, \hat{T}, Nx, VerSU_x)
\end{array} \tag{29}$$

$$\begin{array}{l}
\mathbf{24, 25, 27, 29, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset \\
\text{Send}(X, \hat{X}, \hat{T}, Nx, VerSU_x) < \\
\text{Rcve}(T, \hat{X}, \hat{T}, Nx, VerSU_x) < \\
\text{Send}(T, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) < \\
\text{Rcve}(X, \hat{T}, \hat{X}, Nt, VerSU_t, cert_1) < \\
\text{Send}(X, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) < \\
\text{Rcve}(T, \hat{X}, \hat{T}, cert_2, enc_2, sig_2, hash_2) < \\
\text{Send}(T, \hat{T}, \hat{X}, hash_3)
\end{array} \tag{30}$$

Key Delivery, Server:

$$\begin{array}{l}
\mathbf{AA1, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{Has}(T, xxKey_X)
\end{array} \tag{31}$$

$$\begin{array}{l}
\mathbf{23, 31, \Theta_{TLS}} \\
[\mathbf{TLS : SRVR}]_T \\
\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset \\
\text{Has}(X, xxKey_X) \wedge \text{Has}(T, xxKey_X)
\end{array} \tag{32}$$

B Proof of Security Goals of Four-Way Handshake

B.1 Proof Security Goals Four-Way, Candidate MP

$$\begin{array}{l}
\mathbf{AA1, ARP, AA4, \Theta_{4WAY}} \\
[\mathbf{4WAY : INIT}]_X \\
(\text{Send}(X, \hat{Y}, \hat{X}, INFO_X, \text{``IATH1''}) \wedge \\
\text{Rcve}(X, \hat{X}, \hat{Y}, INFO_Y, \text{``IATH1''}) \wedge \\
\text{Send}(X, \hat{Y}, \hat{X}, INFO_Y, \text{``IATH3''}) \wedge \\
\text{Rcve}(X, \hat{X}, \hat{Y}, INFO_X, \text{``IATH3''})) < \\
\text{Rcve}(X, \hat{X}, \hat{Y}, \text{``IAUTH5''}, y, TN_x) < \\
\text{Send}(X, \hat{Y}, \hat{X}, \text{``IAUTH6''}, x, mic_1, INFO_X, enc_1) < \\
\text{Rcve}(X, \hat{X}, \hat{Y}, \text{``IAUTH7''}, y, mic_2, INFO_Y, TN_x, enc_2) < \\
\text{Send}(X, \hat{Y}, \hat{X}, \text{``IAUTH8''}, mic_3)
\end{array} \tag{33}$$

$$\begin{array}{l}
\mathbf{ARP, VER, \Theta_{4WAY}} \\
[\mathbf{4WAY : INIT}]_X \\
\text{Rcve}(X, \hat{X}, \hat{Y}, \text{``IAUTH7''}, y, mic_2, INFO_Y, TN_x, enc_2) \supset \\
\exists Z.\text{Computes}(Z, \text{HASH}_{ptk_{X,Y}} \\
\quad (\text{``IAUTH7''}, y, pmkN, INFO_Y, TN_x, enc_2)) \wedge \\
\text{Send}(Z, \text{HASH}_{ptk_{X,Y}}
\end{array}$$

$(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2))\wedge$
$(\text{Send}(Z, \text{HASH}_{ptk_{X,Y}}$
$(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2)) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2))$ (34)

$\mathbf{\Gamma_{4WAY,SI,3}, HASH1}$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Computes}(Z, \text{HASH}_{ptk_{X,Y}}$
$(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2)) \supset$
$\text{Has}(Z, ptk_{X,Y}) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y} \vee \hat{Z} = \hat{T}$ (35)

$\mathbf{34, 35, AA1, \Gamma_{4WAY,1}, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(Z, \text{HASH}_{ptk_{X,Y}}$
$(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2)) \supset$
$\hat{Z} = \hat{Y}$ (36)

$\mathbf{34, 36, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Computes}(Y, \text{HASH}_{ptk_{X,Y}}$
$(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2))\wedge$
$\text{Send}(Y, \text{HASH}_{ptk_{X,Y}}$
$(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2))$ (37)

$\mathbf{37, \phi_{HONESTY}, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Computes}(Y, \text{HASH}_{ptk_{X,Y}}$
$(\text{“IAUTH6”}, x, pmkN, INFO_X, enc_1))$ (38)

$\mathbf{37, 38, HASH1, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Has}(Y, ptk_{X,Y})\wedge$
$\text{Has}(Y, TN_x, pmkN, INFO_X, INFO_Y, x, y)$ (39)

$\mathbf{34, 37, 39, \phi_{HONESTY}, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$(\text{Send}(Y, \hat{X}, \hat{Y}, INFO_Y, \text{“IATH1”})\wedge$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, INFO_X, \text{“IATH1”})\wedge$
$\text{Send}(Y, \hat{X}, \hat{Y}, INFO_X, \text{“IATH3”})\wedge$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, INFO_Y, \text{“IATH3”})) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2)$ (40)

$\mathbf{34, 40, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2)$ (41)

$\mathbf{FS1, AN3, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{FirstSend}(X, x, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1)$ (42)

$\mathbf{40, 42, FS2, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1) <$
$\text{Rcve}(X, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1)$ (43)

$\mathbf{FS1, AN3, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{Honest}(\hat{Y}) \supset$
$\text{FirstSend}(Y, y, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x)$ (44)

$\mathbf{40, 44, FS2, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{Honest}(\hat{Y}) \supset \text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x)$ (45)

$\mathbf{40, 41, 43, 45, \Theta_{4WAY}}$
$[\mathbf{4WAY : INIT}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$(\text{Send}(X, \hat{Y}, \hat{X}, INFO_X, \text{“IATH1”})\wedge$
$\text{Rcve}(X, \hat{X}, \hat{Y}, INFO_Y, \text{“IATH1”})\wedge$
$\text{Send}(X, \hat{Y}, \hat{X}, INFO_Y, \text{“IATH3”})\wedge$
$\text{Rcve}(X, \hat{X}, \hat{Y}, INFO_X, \text{“IATH3”})\wedge$
$\text{Send}(Y, \hat{X}, \hat{Y}, INFO_Y, \text{“IATH1”})\wedge$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, INFO_X, \text{“IATH1”})\wedge$
$\text{Send}(Y, \hat{X}, \hat{Y}, INFO_X, \text{“IATH3”})\wedge$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, INFO_Y, \text{“IATH3”})\wedge$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x)) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) <$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2) <$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“IAUTH8”}, mic_3)$ (46)

The other security goal proofs are straightforward and similar to that of ABBH in Section F, therefore we omit them here.

B.1.1 Proof Security Goals Four-Way, MA

Security Goals of the MA:
$\mathbf{\Phi_{4WAY,PTKD,MA}} :=$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Has}(X, ptk_{X,Y}) \wedge \text{Has}(Y, ptk_{X,Y})$

$\mathbf{\Phi_{4WAY,GTKD,MA}} :=$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Has}(X, gtk_Y) \wedge \text{Has}(Y, gtk_X)$

$\mathbf{\Phi_{4WAY,KF,MA}} :=$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$(\text{new } (\hat{X}, x) \wedge x \subseteq ptk_{X,Y} \wedge \text{new } (\hat{Y}, y) \wedge y \subseteq ptk_{X,Y})\wedge$
$\text{FirstSend}(Y, y, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x)\wedge$
$\text{FirstSend}(X, x, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1)$

$\mathbf{\Phi_{4WAY,AUTH,MA}} :=$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$(\text{Send}(X, \hat{Y}, \hat{X}, INFO_X, \text{“IATH1”})\wedge$
$\text{Rcve}(X, \hat{X}, \hat{Y}, INFO_Y, \text{“IATH1”})\wedge$
$\text{Send}(X, \hat{Y}, \hat{X}, INFO_Y, \text{“IATH3”})\wedge$
$\text{Rcve}(X, \hat{X}, \hat{Y}, INFO_X, \text{“IATH3”})\wedge$
$\text{Send}(Y, \hat{X}, \hat{Y}, INFO_Y, \text{“IATH1”})\wedge$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, INFO_X, \text{“IATH1”})\wedge$

$$\begin{array}{l}
\text{Send}(Y,\hat{X},\hat{Y},INFO_X,\text{“IATH3”})\wedge\\
\text{Rcve}(Y,\hat{Y},\hat{X},INFO_Y,\text{“IATH3”})\wedge\\
\text{Send}(Y,\hat{X},\hat{Y},\text{“IAUTH5”},y,TN_x))<\\
\text{Rcve}(X,\hat{X},\hat{Y},\text{“IAUTH5”},y,TN_x)<\\
\text{Send}(X,\hat{Y},\hat{X},\text{“IAUTH6”},x,mic_1,INFO_X,enc_1)<\\
\text{Rcve}(Y,\hat{Y},\hat{X},\text{“IAUTH6”},x,mic_1,INFO_X,enc_1)<\\
\text{Send}(Y,\hat{X},\hat{Y},\text{“IAUTH7”},y,mic_2,INFO_Y,TN_x,enc_2)<\\
\text{Rcve}(X,\hat{X},\hat{Y},\text{“IAUTH7”},y,mic_2,INFO_Y,TN_x,enc_2)<\\
\text{Send}(X,\hat{Y},\hat{X},\text{“IAUTH8”},mic_3)<\\
\text{Rcve}(Y,\hat{Y},\hat{X},\text{“IAUTH8”},mic_3)
\end{array}$$

$$\begin{array}{l}
\boldsymbol{\Phi}_{\mathbf{4WAY},\mathbf{CS},\mathbf{MA}}:=\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
SELECT(INFO_X,INFO_Y)=CS\wedge\\
\text{Has}(X,CS)\wedge\text{Has}(Y,CS)
\end{array}$$

Matching Conversation MA:
AA1, ARP, AA4, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
(\text{Send}(Y,\hat{X},\hat{Y},INFO_Y,\text{“IATH1”})\wedge\\
\text{Rcve}(Y,\hat{Y},\hat{X},INFO_X,\text{“IATH1”})\wedge\\
\text{Send}(Y,\hat{X},\hat{Y},INFO_X,\text{“IATH3”})\wedge\\
\text{Rcve}(Y,\hat{Y},\hat{X},INFO_Y,\text{“IATH3”}))<\\
\text{Send}(Y,\hat{X},\hat{Y},\text{“IAUTH5”},y,TN_x)<\\
\text{Rcve}(Y,\hat{Y},\hat{X},\text{“IAUTH6”},x,mic_1,INFO_X,enc_1)<\\
\text{Send}(Y,\hat{X},\hat{Y},\text{“IAUTH7”},y,mic_2,INFO_Y,TN_x,enc_2)<\\
\text{Rcve}(Y,\hat{Y},\hat{X},\text{“IAUTH8”},mic_3)
\end{array}\tag{47}$$

ARP, HASH3, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{Rcve}(Y,\hat{Y},\hat{X},\text{“IAUTH8”},mic_3)\supset\\
\exists Z.\text{Computes}(Z,\text{HASH}_{ptk_{X,Y}}(\text{“IAUTH8”}))\wedge\\
\text{Send}(Z,\text{HASH}_{ptk_{X,Y}}(\text{“IAUTH8”}))\wedge\\
(\text{Send}(Z,\text{HASH}_{ptk_{X,Y}}(\text{“IAUTH8”}))<\\
\text{Rcve}(Y,\hat{Y},\hat{X},\text{“IAUTH8”},mic_3))
\end{array}\tag{48}$$

$\boldsymbol{\Gamma}_{\mathbf{4WAY},\mathbf{SI},\mathbf{3}}$, HASH1

$$\begin{array}{l}
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Computes}(Z,\text{HASH}_{ptk_{X,Y}}(\text{“IAUTH8”}))\supset\\
\text{Has}(Z,ptk_{X,Y})\supset\hat{Z}=\hat{X}\vee\hat{Z}=\hat{Y}\vee\hat{Z}=\hat{T}
\end{array}\tag{49}$$

48, 49, AA1, $\boldsymbol{\Gamma}_{\mathbf{4WAY},\mathbf{1}}$, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Send}(Z,\text{HASH}_{ptk_{X,Y}}(\text{“IAUTH8”}))\supset\\
\hat{Z}=\hat{X}
\end{array}\tag{50}$$

48, 50, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Computes}(X,\text{HASH}_{ptk_{X,Y}}(\text{“IAUTH8”}))\wedge\\
\text{Send}(X,\text{HASH}_{ptk_{X,Y}}(\text{“IAUTH8”}))
\end{array}\tag{51}$$

51, $\phi_{\mathbf{HONESTY}}$, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Computes}(X,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH7”},y,pmkN,INFO_Y,TN_x,enc_2))
\end{array}\tag{52}$$

ARP, HASH3, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{Rcve}(Y,\hat{Y},\hat{X},\text{“IAUTH6”},x,mic_1,INFO_X,enc_1)\supset\\
\exists Z.\text{Computes}(Z,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH6”},x,pmkN,INFO_X,enc_1))\wedge\\
\text{Send}(Z,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH6”},x,pmkN,INFO_X,enc_1))
\end{array}\tag{53}$$

$\boldsymbol{\Gamma}_{\mathbf{4WAY},\mathbf{SI},\mathbf{3}}$, HASH1

$$\begin{array}{l}
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Computes}(Z,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH6”},x,pmkN,INFO_X,enc_1))\supset\\
\text{Has}(Z,ptk_{X,Y})\supset\hat{Z}=\hat{X}\vee\hat{Z}=\hat{Y}\vee\hat{Z}=\hat{T}
\end{array}\tag{54}$$

53, 54, AA1, $\boldsymbol{\Gamma}_{\mathbf{4WAY},\mathbf{1}}$, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Send}(Z,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH6”},x,pmkN,INFO_X,enc_1))\supset\\
\hat{Z}=\hat{X}
\end{array}\tag{55}$$

53, 55, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Computes}(X,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH6”},x,pmkN,INFO_X,enc_1))\wedge\\
\text{Send}(X,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH6”},x,pmkN,INFO_X,enc_1))
\end{array}\tag{56}$$

51, 52, 56, HASH1, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Has}(X,ptk_{X,Y})\wedge\\
\text{Has}(X,TN_x,pmkN,INFO_X,INFO_Y,x,y)
\end{array}\tag{57}$$

48, 51, 57, $\phi_{\mathbf{HONESTY}}$, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
(\text{Send}(X,\hat{Y},\hat{X},INFO_X,\text{“IATH1”})\wedge\\
\text{Rcve}(X,\hat{X},\hat{Y},INFO_Y,\text{“IATH1”})\wedge\\
\text{Send}(X,\hat{Y},\hat{X},INFO_Y,\text{“IATH3”})\wedge\\
\text{Rcve}(X,\hat{X},\hat{Y},INFO_X,\text{“IATH3”}))<\\
\text{Rcve}(X,\hat{X},\hat{Y},\text{“IAUTH5”},y,TN_x)<\\
\text{Send}(X,\hat{Y},\hat{X},\text{“IAUTH6”},x,mic_1,INFO_X,enc_1)<\\
\text{Rcve}(X,\hat{X},\hat{Y},\text{“IAUTH7”},y,mic_2,INFO_Y,TN_x,enc_2)<\\
\text{Send}(X,\hat{Y},\hat{X},\text{“IAUTH8”},mic_3)
\end{array}\tag{58}$$

48, 58, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\supset\\
\text{Send}(X,\hat{Y},\hat{X},\text{“IAUTH8”},mic_3)<\\
\text{Rcve}(Y,\hat{Y},\hat{X},\text{“IAUTH8”},mic_3)
\end{array}\tag{59}$$

ARP, VER, $\boldsymbol{\Theta}_{\mathbf{4WAY}}$

$$\begin{array}{l}
[\mathbf{4WAY}:\mathbf{MA}]_Y\\
\text{KOHonest}(ptk_{X,Y},\{pmk_{X,Y},pmk_{Y,X}\})\wedge\\
\text{Rcve}(X,\hat{X},\hat{Y},\text{“IAUTH7”},y,mic_2,INFO_Y,TN_x,enc_2)\supset\\
\exists Z.\text{Computes}(Z,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH7”},y,pmkN,INFO_Y,TN_x,enc_2))\wedge\\
\text{Send}(Z,\text{HASH}_{ptk_{X,Y}}\\
\quad(\text{“IAUTH7”},y,pmkN,INFO_Y,TN_x,enc_2))\wedge\\
(\text{Send}(Z,\text{HASH}_{ptk_{X,Y}}
\end{array}$$

$\quad(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2)) <$
$\text{Rcve}(Y, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2)) \quad (60)$

$\mathbf{\Gamma_{4WAY,SI,3}, HASH1}$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Computes}(Z, \text{HASH}_{ptk_{X,Y}}$
$\quad(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2)) \supset$
$\text{Has}(Z, ptk_{X,Y}) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y} \vee \hat{Z} = \hat{T} \quad (61)$

$\mathbf{60, 61, AA1, \Gamma_{4WAY,1}, \Theta_{4WAY}}$
$[\mathbf{4WAY : MA}]_Y$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(Z, \text{HASH}_{ptk_{X,Y}}$
$\quad(\text{“IAUTH7”}, y, pmkN, INFO_Y, TN_x, enc_2)) \supset$
$\hat{Z} = \hat{Y} \quad (62)$

$\mathbf{60, 62, \Theta_{4WAY}}$
$[\mathbf{4WAY : MA}]_Y$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2) <$
$\text{Rcve}(Y, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2) \quad (63)$

$\mathbf{FS1, AN3, \Theta_{4WAY}}$
$[\mathbf{4WAY : MA}]_Y$
$\text{Honest}(\hat{X}) \supset$
$\text{FirstSend}(X, x, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) \quad (64)$

$\mathbf{58, 64, FS2, \Theta_{4WAY}}$
$[\mathbf{4WAY : MA}]_Y$
$\text{Honest}(\hat{X}) \supset$
$\text{Send}(X, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) <$
$\text{Rcve}(Y, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) \quad (65)$

$\mathbf{FS1, AN3, \Theta_{4WAY}}$
$[\mathbf{4WAY : MA}]_Y$
$\text{FirstSend}(Y, y, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) \quad (66)$

$\mathbf{58, 66, FS2, \Theta_{4WAY}}$
$[\mathbf{4WAY : MA}]_Y$
$\text{Send}(X, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) \quad (67)$

$\mathbf{58, 59, 63, 65, 67, \Theta_{4WAY}}$
$[\mathbf{4WAY : MA}]_Y$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$(\text{Send}(X, \hat{Y}, \hat{X}, INFO_X, \text{“IATH1”}) \wedge$
$\text{Rcve}(X, \hat{X}, \hat{Y}, INFO_Y, \text{“IATH1”}) \wedge$
$\text{Send}(X, \hat{Y}, \hat{X}, INFO_Y, \text{“IATH3”}) \wedge$
$\text{Rcve}(X, \hat{X}, \hat{Y}, INFO_X, \text{“IATH3”}) \wedge$
$\text{Send}(Y, \hat{X}, \hat{Y}, INFO_Y, \text{“IATH1”}) \wedge$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, INFO_X, \text{“IATH1”}) \wedge$
$\text{Send}(Y, \hat{X}, \hat{Y}, INFO_X, \text{“IATH3”}) \wedge$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, INFO_Y, \text{“IATH3”}) \wedge$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x)) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“IAUTH5”}, y, TN_x) <$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“IAUTH6”}, x, mic_1, INFO_X, enc_1) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“IAUTH7”}, y, mic_2, INFO_Y, TN_x, enc_2) <$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“IAUTH8”}, mic_3) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“IAUTH8”}, mic_3) \quad (68)$

The other security goal proofs are straightforward and similar to that of ABBH in Section F, therefore we omit them here.

C Security Goals of MKHSH

Security Goals of the MA:
$\mathbf{\Phi_{MKHSH,AUTH,MA}} :=$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(X, \text{“MKH1”}, x, \hat{X}, \hat{T}, INFO_X) <$
$\text{Rcve}(T, \text{“MKH1”}, x, \hat{X}, \hat{T}, INFO_X) <$
$\text{Send}(T, \text{“MKH2”}, x, t, \hat{X}, \hat{T}, INFO_T, mic_0) <$
$\text{Rcve}(X, \text{“MKH2”}, x, t, \hat{X}, \hat{T}, INFO_T, mic_0) <$
$\text{Send}(X, \text{“MKH3”}, x, t, \hat{X}, \hat{T}, INFO_X, mic_1) <$
$\text{Rcve}(T, \text{“MKH3”}, x, t, \hat{X}, \hat{T}, INFO_X, mic_1) <$
$\text{Send}(T, \text{“MKH4”}, x, t, \hat{X}, \hat{T}, INFO_T, mic_2) <$
$\text{Rcve}(X, \text{“MKH4”}, x, t, \hat{X}, \hat{T}, INFO_T, mic_2)$

$\mathbf{\Phi_{MKHSH,MPTKD,MA}} :=$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Has}(X, mptk_{X,T}) \wedge \text{Has}(T, mptk_{X,T})$

$\mathbf{\Phi_{MKHSH,GTKD,MA}} :=$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \text{Has}(T, gtk_X)$

$\mathbf{\Phi_{MKHSH,KF,MA}} := \text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$(\text{new } (\hat{X}, x) \wedge x \subseteq mptk_{X,T} \wedge \text{new } (\hat{T}, t) \wedge t \subseteq mptk_{X,T}) \wedge$
$\text{FirstSend}(X, x, \text{“MKH1”}, x, \hat{X}, \hat{T}, INFO_X) \wedge$
$\text{FirstSend}(T, t, \text{“MKH2”}, x, t, \hat{X}, \hat{T}, INFO_T, mic_0)$

$\mathbf{\Phi_{MKHSH,INFO,MA}} := \text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$SELECT(INFO_X, INFO_T) = CS \wedge$
$\text{Has}(X, CS) \wedge \text{Has}(T, CS)$

Security Goals of the MKD:
$\mathbf{\Phi_{MKHSH,AUTH,MKD}} :=$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(X, \text{“MKH1”}, x, \hat{X}, \hat{T}, INFO_X) <$
$\text{Rcve}(T, \text{“MKH1”}, x, \hat{X}, \hat{T}, INFO_X) <$
$\text{Send}(T, \text{“MKH2”}, x, t, \hat{X}, \hat{T}, INFO_T, mic_0) <$
$\text{Rcve}(X, \text{“MKH2”}, x, t, \hat{X}, \hat{T}, INFO_T, mic_0) <$
$\text{Send}(X, \text{“MKH3”}, x, t, \hat{X}, \hat{T}, INFO_X, mic_1) <$
$\text{Rcve}(T, \text{“MKH3”}, x, t, \hat{X}, \hat{T}, INFO_X, mic_1) <$
$\text{Send}(T, \text{“MKH4”}, x, t, \hat{X}, \hat{T}, INFO_T, mic_2)$

$\mathbf{\Phi_{MKHSH,MPTKD,MKD}} := \mathbf{\Phi_{MKHSH,MPTKD,INIT}}$

$\mathbf{\Phi_{MKHSH,GTKD,MKD}} := \mathbf{\Phi_{MKHSH,GTKD,MA}}$

$\mathbf{\Phi_{MKHSH,KF,MKD}} := \mathbf{\Phi_{MKHSH,KF,INIT}}$

$\mathbf{\Phi_{MKHSH,INFO,MKD}} := \mathbf{\Phi_{MKHSH,INFO,INIT}}$

D Group Key Handshake

D.1 Security Goals of the Group Key Handshake

Security Goals, Sender:

$\mathbf{\Phi_{GKH,KD,SNDR}} :=$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$ Has(Y, gtk_X)

$\mathbf{\Phi_{GKH,Auth,SNDR}} :=$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Send$(X,$ “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1) <$
Rcve$(Y,$ “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1) <$
Send$(Y,$ “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X}) <$
Rcve$(X,$ “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X})$

Security Goals, Receiver:
$\mathbf{\Phi_{GKH,Auth,RCVR}} :=$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Send$(X,$ “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1) <$
Rcve$(Y,$ “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1) <$
Send$(Y,$ “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X})$ (69)

We omit the proof of these security goals as it is nearly identical to the proof of delete.

D.2 Proof of ordering

In this section we present one of the most technically challenging proofs. We show that $\Gamma_{GKH,2}$ and $\Theta_{GKH,2}$ hold over every basic sequence of the Group Key Handshake.

Proof of ordering, sender:
AA2, **Start**(X)
$[]_X$
$\neg$Send$(X, m) \wedge \neg$Send$(Y, m) \supset \Theta_{GKH,2}, \Gamma_{GKH,2}$ (70)

AA1, **AA2**, $\mathbf{\Theta_{GKH,2}}$
[mtch $ngtkc_{X,Y}$/Increment$(gtk_{X,Y})$;
return $ngtkc_{X,Y}$; mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
“GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
send “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1]_X$
$\neg$Send$(X, m) \wedge$
IsLess$(currgtkc_{X,Y},$ Increment$(globalgtkc_{X,Y})) \supset$
$\Theta_{GKH,2}$ (71)

AA1, $\mathbf{\Theta_{GKH,2}}$, $\mathbf{\Gamma_{GKH,2}}$
[mtch $ngtkc_{X,Y}$/Increment$(gtk_{X,Y})$;
return $ngtkc_{X,Y}$; mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
“GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
send “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1]_X$
Honest$(\hat{X}) \wedge \diamondminus$ Send$(X, m) \wedge$
$m \neq$ “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1 \wedge$
Contains$(m, gtkc'_{X,Y}) \supset$
IsLess$(gtkc'_{X,Y},$ Increment$(gtkc_{X,Y}))$ (72)

AA4, $\mathbf{\Theta_{GKH,2}}$, $\mathbf{\Gamma_{GKH,2}}$
[mtch $ngtkc_{X,Y}$/Increment$(gtk_{X,Y})$;
return $ngtkc_{X,Y}$; mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
“GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
send “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1]_X$
Honest$(\hat{X}) \wedge \diamondminus$ Send$(X, m) \wedge$
$m \neq$ “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1 \supset$
Send$(X, m) <$
Send$(X,$ “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1)$ (73)

72, **73**, $\mathbf{\Theta_{GKH,2}}$, $\mathbf{\Gamma_{GKH,2}}$
[mtch $ngtkc_{X,Y}$/Increment$(gtk_{X,Y})$;
return $ngtkc_{X,Y}$; mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
“GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
send “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1]_X$
Honest$(\hat{X}) \wedge$ Send$(X, m_0) \wedge$ Send$(X, m_1) \wedge$
Contains$(m_0, gtkc'_{X,Y}) \wedge$ Contains$(m_1, gtkc_{X,Y}) \wedge$
IsLess$(gtkc'_{X,Y}, gtkc_{X,Y}) \supset$ Send$(X, m_0) <$ Send(X, m_1) (74)

74, **AA2**, $\mathbf{\Theta_{GKH,2}}$, $\mathbf{\Gamma_{GKH,2}}$
[mtch $ngtkc_{X,Y}$/Increment$(gtk_{X,Y})$;
return $ngtkc_{X,Y}$; mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
“GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
send “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1]_X$
Honest$(\hat{X}) \wedge \neg$Send$(Y, m) \supset \Gamma_{GKH,2}$ (75)

AA2, $\mathbf{\Theta_{GKH,2}}$, $\mathbf{\Gamma_{GKH,2}}$
[rcve “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X}$;
mtch mic_2/HASH$_{ptk_{X,Y}}($
“GKH2”$, ngtkc_{X,Y}, \hat{Y}, \hat{X})]_X$
$\neg$Send$(X, m) \wedge \neg$Send$(Y, m) \supset \Theta_{GKH,2} \wedge \Gamma_{GKH,2}$ (76)

Proof of ordering, receiver:
AA2 Start(Y)
$[]_Y$
$\neg$Send$(Y, m) \wedge \neg$Send$(X, m) \supset \Theta_{GKH,2}, \Gamma_{GKH,2}$ (77)

AA1, $\mathbf{\Theta_{GKH,2}}$
[rcve “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1$;
IsLess$(gtkc'_{X,Y}, ngtkc_{X,Y})$;
mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
“GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
return $ngtkc_{Y,T}$;
mtch mic_2/HASH$_{ptk_{X,Y}}($
“GKH2”$, ngtkc_{X,Y}, \hat{Y}, \hat{X})$;
send “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X}]_Y$
IsLess$(currgtkc_{X,Y},$ Increment$(globalgtkc_{X,Y}))$
$\supset \Theta_{GKH,2}$ (78)

AA1, $\mathbf{\Theta_{GKH,2}}$, $\mathbf{\Gamma_{GKH,2}}$
[rcve “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1$;
IsLess$(gtkc'_{X,Y}, ngtkc_{X,Y})$;
mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
“GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
return $ngtkc_{Y,T}$;
mtch mic_2/HASH$_{ptk_{X,Y}}($
“GKH2”$, ngtkc_{X,Y}, \hat{Y}, \hat{X})$;
send “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X}]_Y$
Honest$(\hat{Y}) \wedge \diamondminus$ Send$(Y, m) \wedge$
$m \neq$ “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X} \wedge$
Contains$(m, gtkc'_{X,Y}) \supset$
IsLess$(gtkc'_{X,Y}, currgtkc_{X,Y})$ (79)

AA4, $\mathbf{\Theta_{GKH,2}}$, $\mathbf{\Gamma_{GKH,2}}$
[rcve “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1$;

IsLess($gtkc'_{X,Y}, ngtkc_{X,Y}$);
mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
  “GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
return $ngtkc_{Y,T}$;
mtch mic_2/HASH$_{ptk_{X,Y}}($
  “GKH2”$, ngtkc_{X,Y}, \hat{Y}, \hat{X})$;
send “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X}]_Y$
Honest$(\hat{Y}) \wedge \diamondminus$ Send$(Y, m) \wedge$
$m \neq$ “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X} \supset$
Send$(Y, m) <$
Send$(Y,$ “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X})$ (80)

$\mathbf{79}, \mathbf{80}, \mathbf{AA2}, \mathbf{\Theta_{GKH,2}}, \mathbf{\Gamma_{GKH,2}}$
[rcve “GKH1”$, ngtkc_{X,Y}, mic_1, \hat{X}, \hat{Y}, enc_1$;
IsLess($gtkc'_{X,Y}, ngtkc_{X,Y}$);
mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}($
  “GKH1”$, ngtkc_{X,Y}, \hat{X}, \hat{Y}, enc_1)$;
return $ngtkc_{Y,T}$;
mtch mic_2/HASH$_{ptk_{X,Y}}($
  “GKH2”$, ngtkc_{X,Y}, \hat{Y}, \hat{X})$;
send “GKH2”$, ngtkc_{X,Y}, mic_2, \hat{Y}, \hat{X}]_Y$
Honest$(\hat{Y}) \wedge \neg$Send$(X, m) \supset \Gamma_{GKH,2}$ (81)

E Push, Pull and Del

E.1 Push and Pull

Remainder of invariants of Push, Pull and Del:
$\mathbf{\Gamma_{PPD,3}} :=$ Honest$(\hat{Z}) \wedge$
(Send$(Z, m_0) \wedge$ Send$(Z, m_1) \wedge$
Contains$(m_0,$ HASH$_{mptk_{Z,X}}(plc'_{Z,X},$ PULL2HASH$))) \wedge$
Contains$(m_0,$ HASH$_{mptk_{Z,X}}(plc_{Z,X},$ PULL2HASH$))) \wedge$
IsLess$(plc'_{Z,X}, plc_{Z,X}) \supset$ Send$(Z, m_0) <$ Send$(Z, m_1))$

$\mathbf{\Gamma_{PPD,4}}$Honest$(\hat{Z}) \wedge$
(Send$(Z, m_0) \wedge$ Send$(Z, m_1) \wedge$
Contains$(m_0,$ HASH$_{mptk_{Z,X}}(pdc'_{Z,X},$ PUSH1HASH$))) \wedge$
Contains$(m_0,$ HASH$_{mptk_{Z,X}}(pdc_{Z,X},$ PUSH1HASH$))) \wedge$
IsLess$(pdc'_{Z,X}, pdc_{Z,X}) \supset$ Send$(Z, m_0) <$ Send$(Z, m_1))$

$\mathbf{\Gamma_{PPD,5}}$Honest$(\hat{Z}) \wedge$
(Send$(Z, m_0) \wedge$ Send$(Z, m_1) \wedge$
Contains$(m_0,$ HASH$_{mptk_{Z,X}}(pdc'_{Z,X},$ DEL1HASH$))) \wedge$
Contains$(m_0,$ HASH$_{mptk_{Z,X}}(pdc_{Z,X},$ DEL1HASH$))) \wedge$
IsLess$(pdc'_{Z,X}, pdc_{Z,X}) \supset$ Send$(Z, m_0) <$ Send$(Z, m_1))$

$\mathbf{\Gamma_{PPD,6}}$Honest$(\hat{Z}) \wedge$
(Send$(Z, m_0) \wedge$ Send$(Z, m_1) \wedge$
Contains$(m_0,$ HASH$_{mptk_{Z,X}}(pdc'_{Z,X},$ DEL2HASH$))) \wedge$
Contains$(m_0,$ HASH$_{mptk_{Z,X}}(pdc_{Z,X},$ DEL2HASH$))) \wedge$
IsLess$(pdc'_{Z,X}, pdc_{Z,X}) \supset$ Send$(Z, m_0) <$ Send$(Z, m_1))$

Goals of Push and Pull, MKD:
$\mathbf{\Phi_{Pull,Auth,MKD}} :=$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Send$(Y, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
Rcve$(T, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
Send$(T, \hat{Y}, \hat{T},$ “PULL2”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y},$
  $TN_x, enc_1, mic_2)$

$\mathbf{\Phi_{Push,Auth,MKD}} :=$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
(Send$(T, \hat{Y}, \hat{T},$ “PUSH1”$, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) \wedge$
Send$(Y, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)) <$
Rcve$(T, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
Send$(T, \hat{Y}, \hat{T},$ “PULL2”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y},$
  $TN_x, enc_1, mic_2)$

Proof Authentication PULL, MKD:
$\mathbf{AA1}, \mathbf{ARP}, \mathbf{AA4}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PULL} : \mathbf{MKD}]_T$
Honest$(\hat{T}) \supset$
Rcve$(T, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
Send$(T, \hat{Y}, \hat{T},$ “PULL2”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y},$
  $TN_x, enc_1, mic_2)$ (82)

$\mathbf{ARP}, \mathbf{HASH3}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PULL} : \mathbf{MKD}]_T$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Rcve$(T, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) \supset$
$\exists Z.$Computes$(Z,$ HASH$_{mptk_{Y,T}}($
  $\hat{Y}, \hat{T},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y})) \wedge$
Send$(Z,$ HASH$_{mptk_{Y,T}}($
  $\hat{Y}, \hat{T},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y})) <$
Rcve$(Y, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$ (83)

$\mathbf{\Gamma_{PPD,SI,1}}, \mathbf{HASH1}$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Computes$(Z,$ HASH$_{mptk_{Y,T}}($
  $\hat{Y}, \hat{T},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y})) \supset$
Has$(Z, mptk_{Y,T}) \supset \hat{Z} = \hat{T} \vee \hat{Z} = \hat{Y}$ (84)

$\mathbf{83}, \mathbf{84}, \mathbf{AA1}, \mathbf{\Gamma_{PPD,1}}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PULL} : \mathbf{MKD}]_T$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Send$(Z,$ HASH$_{mptk_{Y,T}}($
  $\hat{Y}, \hat{T},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y})) \supset$
$\hat{Z} = \hat{Y}$ (85)

$\mathbf{83}, \mathbf{85}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PULL} : \mathbf{MKD}]_T$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Computes$(Y,$ HASH$_{mptk_{Y,T}}($
  $\hat{Y}, \hat{T},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y})) \wedge$
Send$(Y,$ HASH$_{mptk_{Y,T}}($
  $\hat{Y}, \hat{T},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}))$ (86)

$\mathbf{86}, \mathbf{HASH1}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PULL} : \mathbf{MKD}]_T$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Has$(Y, mptk_{Y,T}) \wedge$
Has$(Y, \hat{Y}, \hat{T}, nplc_{Y,T},$ “PULL1”$, \hat{X}, pmkN_{X,Y})$ (87)

$\mathbf{83}, \mathbf{86}, \mathbf{87}, \mathbf{\phi_{HONESTY}}\Theta_{PPD,1,2}$
$[\mathbf{PULL} : \mathbf{MKD}]_T$
KOHonest$(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
Send$(Y, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
Rcve$(T, \hat{T}, \hat{Y},$ “PULL1”$, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$ (88)

$\mathbf{82}, \mathbf{88}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PULL} : \mathbf{MKD}]_T \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned} \tag{89}$$

Proof Authentication Push MKD:
$\mathbf{AA1}, \mathbf{ARP}, \mathbf{AA4}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PUSH} : \mathbf{MKD}]_T \\
&\mathrm{Honest}(\hat{T}) \supset \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) < \\
&\mathrm{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)
\end{aligned} \tag{90}$$

$\mathbf{89}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PUSH} : \mathbf{MKD}]_T \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned} \tag{91}$$

$\mathbf{90}, \mathbf{91}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PUSH} : \mathbf{MKD}]_T \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&(\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) \wedge \\
&\mathrm{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)) < \\
&\mathrm{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned} \tag{92}$$

Goals of Push and Pull, MA:

$$\begin{aligned}
&\mathbf{\Phi_{\{PUSH,PULL\},KD,MA}} := \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \mathrm{Has}(Y, pmk_{X,Y})
\end{aligned}$$

$$\begin{aligned}
&\mathbf{\Phi_{Pull,Auth,MA}} := \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2) < \\
&\mathrm{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned}$$

$$\begin{aligned}
&\mathbf{\Phi_{Push,Auth,MA}} := \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) < \\
&\mathrm{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) < \\
&\mathrm{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2) < \\
&\mathrm{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned}$$

Proof of Authentication PULL, MA:
$\mathbf{ARP}, \mathbf{HASH3}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PULL} : \mathbf{MA}]_Y \\
&\mathrm{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2) \supset \\
&\exists Z.\mathrm{Computes}(Z, \mathrm{HASH}_{mptk_{Y,T}}(\hat{Y}, \hat{T}, \text{“PULL2”}, \\
&\quad nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, TN_x, enc_1)) \wedge \\
&\mathrm{Send}(Z, \mathrm{HASH}_{mptk_{Y,T}}(\hat{Y}, \hat{T}, \text{“PULL2”}, \\
&\quad nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, TN_x, enc_1)) < \\
&\mathrm{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned} \tag{93}$$

$\mathbf{\Gamma_{PPD,SI,1}}, \mathbf{HASH1}$

$$\begin{aligned}
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Computes}(Z, \mathrm{HASH}_{mptk_{Y,T}}(\hat{Y}, \hat{T}, \text{“PULL2”}, \\
&\quad nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, TN_x, enc_1)) \supset \\
&\mathrm{Has}(Z, mptk_{Y,T}) \supset \hat{Z} = \hat{T} \vee \hat{Z} = \hat{Y}
\end{aligned} \tag{94}$$

$\mathbf{93}, \mathbf{94}, \mathbf{AA1}, \mathbf{\Gamma_{PPD,1}}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PULL} : \mathbf{MA}]_Y \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Send}(Z, \mathrm{HASH}_{mptk_{Y,T}}(\hat{Y}, \hat{T}, \text{“PULL2”}, \\
&\quad nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, TN_x, enc_1)) \supset \\
&\hat{Z} = \hat{T}
\end{aligned} \tag{95}$$

$\mathbf{93}, \mathbf{95}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PULL} : \mathbf{MA}]_Y \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Computes}(T, \mathrm{HASH}_{mptk_{Y,T}}(\hat{Y}, \hat{T}, \text{“PULL2”}, \\
&\quad nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, TN_x, enc_1)) \wedge \\
&\mathrm{Send}(T, \mathrm{HASH}_{mptk_{Y,T}}(\hat{Y}, \hat{T}, \text{“PULL2”}, \\
&\quad nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, TN_x, enc_1))
\end{aligned} \tag{96}$$

$\mathbf{96}, \mathbf{HASH1}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PULL} : \mathbf{MA}]_Y \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Has}(T, mptk_{Y,T}) \wedge \\
&\mathrm{Has}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned} \tag{97}$$

$\mathbf{89}, \mathbf{96}, \mathbf{97}, \phi_{\mathbf{HONESTY}}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PULL} : \mathbf{MA}]_Y \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned} \tag{98}$$

$\mathbf{93}, \mathbf{98}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PULL} : \mathbf{MA}]_Y \\
&\mathrm{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset \\
&\mathrm{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) < \\
&\mathrm{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2) < \\
&\mathrm{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, \\
&\quad TN_x, enc_1, mic_2)
\end{aligned} \tag{99}$$

Proof of Authentication PUSH MA:
$\mathbf{AA1}, \mathbf{ARP}, \mathbf{AA4}, \mathbf{\Theta_{PPD,1,2}}$

$$\begin{aligned}
&[\mathbf{PUSH} : \mathbf{MA}]_Y \\
&\mathrm{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) < \\
&\mathrm{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)
\end{aligned} \tag{100}$$

$\mathbf{99}, \mathbf{\Theta_{PPD,1,2}}$

$[\mathbf{PUSH} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
$\text{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y},$
$\quad TN_x, enc_1, mic_2) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y},$
$\quad TN_x, enc_1, mic_2)$ (101)

$\mathbf{ARP}, \mathbf{HASH3}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PUSH} : \mathbf{MA}]_Y$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) \supset$
$\exists Z.\text{Computes}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \wedge$
$\text{Send}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0)$ (102)

$\mathbf{\Gamma_{PPD,SI,1}}, \mathbf{HASH1}\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Computes}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \supset$
$\text{Has}(Z, mptk_{Y,T}) \supset \hat{Z} = \hat{T} \vee \hat{Z} = \hat{Y}$ (103)

$\mathbf{102}, \mathbf{103}, \mathbf{AA1}, \mathbf{\Gamma_{PPD,1}}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PUSH} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \supset$
$\hat{Z} = \hat{T}$ (104)

$\mathbf{102}, \mathbf{104}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PUSH} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Computes}(T, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \wedge$
$\text{Send}(T, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}))$ (105)

$\mathbf{105}, \mathbf{HASH1}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PUSH} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Has}(T, mptk_{Y,T}) \wedge$
$\text{Has}(T, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0)$ (106)

$\mathbf{102}, \mathbf{105}, \mathbf{106}, \phi_{\mathbf{HONESTY}}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PUSH} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0)$ (107)

$\mathbf{100}, \mathbf{101}, \mathbf{107}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{PUSH} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PUSH1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Send}(Y, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
$\text{Rcve}(T, \hat{T}, \hat{Y}, \text{“PULL1”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y},$
$\quad TN_x, enc_1, mic_2) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“PULL2”}, nplc_{Y,T}, \hat{X}, pmkN_{X,Y},$
$\quad TN_x, enc_1, mic_2)$ (108)

E.2 Delete

Security Goals Del, MA:
$\mathbf{\Phi_{Del,Auth,MA}} := \text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Send}(Y, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$

Proof of Authentication:
$\mathbf{AA1}, \mathbf{ARP}, \mathbf{AA4}\Theta_{PPD,1,2}$
$[\mathbf{DEL} : \mathbf{MA}]_Y$
$\text{Honest}(\hat{Y}) \supset$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Send}(Y, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$ (109)

$\mathbf{ARP}, \mathbf{HASH3}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) \supset$
$\exists Z.\text{Computes}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \wedge$
$\text{Send}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) <$
$\text{Rcve}(T, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0)$ (110)

$\mathbf{\Gamma_{PPD,SI,1}}, \mathbf{HASH1}\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Computes}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \supset$
$\text{Has}(Z, mptk_{Y,T}) \supset \hat{Z} = \hat{T} \vee \hat{Z} = \hat{Y}$ (111)

$\mathbf{110}, \mathbf{111}, \mathbf{AA1}, \mathbf{\Gamma_{PPD,1}}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \supset$
$\hat{Z} = \hat{T}$ (112)

$\mathbf{110}, \mathbf{112}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Computes}(T, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}) \wedge$
$\text{Send}(T, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}))$ (113)

$\mathbf{113}, \mathbf{HASH1}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Has}(T, mptk_{Y,T}) \wedge$
$\text{Has}(T, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0)$ (114)

$\mathbf{110}, \mathbf{113}, \mathbf{114}, \phi_{\mathbf{HONESTY}}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL} : \mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0)$ (115)

$\mathbf{109}, \mathbf{115}, \mathbf{\Theta_{PPD,1,2}}$

$[\mathbf{DEL}:\mathbf{MA}]_Y$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Send}(Y, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$ (116)

Security Goals Del, MKD:
$\mathbf{\Phi_{Del,Auth,MKD}} :=$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Send}(Y, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
$\text{Rcve}(T, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$

Proof of Auth, MKD:
$\mathbf{ARP}, \mathbf{HASH3}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL}:\mathbf{MKD}]_T$
$\text{Rcve}(T, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) \supset$
$\exists Z.\text{Computes}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \wedge$
$\text{Send}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) <$
$\text{Rcve}(T, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$ (117)

$\mathbf{\Gamma_{PPD,SI,1}}, \mathbf{HASH1}\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Computes}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \supset$
$\text{Has}(Z, mptk_{Y,T}) \supset \hat{Z} = \hat{T} \vee \hat{Z} = \hat{Y}$ (118)

$\mathbf{117}, \mathbf{118}, \mathbf{AA1}, \mathbf{\Gamma_{PPD,1}}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL}:\mathbf{MKD}]_T$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(Z, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \supset$
$\hat{Z} = \hat{Y}$ (119)

$\mathbf{117}, \mathbf{119}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL}:\mathbf{MKD}]_T$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Computes}(Y, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y})) \wedge$
$\text{Send}(Y, \text{HASH}_{mptk_{Y,T}}($
$\quad \hat{Y}, \hat{T}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}))$ (120)

$\mathbf{120}, \mathbf{HASH1}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL}:\mathbf{MKD}]_T$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Has}(Y, mptk_{Y,T}) \wedge$
$\text{Has}(Y, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$ (121)

$\mathbf{116}, \mathbf{120}, \mathbf{121}, \phi_{\mathbf{HONESTY}}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL}:\mathbf{MKD}]_T$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Send}(Y, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$ (122)

$\mathbf{117}, \mathbf{122}, \mathbf{\Theta_{PPD,1,2}}$
$[\mathbf{DEL}:\mathbf{MKD}]_T$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{Send}(T, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Rcve}(Y, \hat{Y}, \hat{T}, \text{“DEL1”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_0) <$
$\text{Send}(Y, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1) <$
$\text{Rcve}(T, \hat{T}, \hat{Y}, \text{“DEL2”}, npdc_{Y,T}, \hat{X}, pmkN_{X,Y}, mic_1)$ (123)

F ABBH

F.1 Security Goals Initiator and Responder, ABBH

Goals Initiator:
$\mathbf{\Phi_{ABBH,MA,INIT}} :=$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH2”}, INFO_Y, y, x, enc_0, mic_0) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH2”}, INFO_Y, y, x, enc_0, mic_0) <$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH3”}, INFO_X, x, y, enc_1, mic_1) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“ABBH3”}, INFO_X, x, y, enc_1, mic_1) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH4”}, y, x, mic_2) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH4”}, y, x, mic_2)$
$\mathbf{\Phi_{ABBH,PTKD,INIT}} := \mathbf{\Phi_{SIMO,PTKD}}$
$\mathbf{\Phi_{ABBH,GTKD,INIT}} := \mathbf{\Phi_{SIMO,GTKD}}$
$\mathbf{\Phi_{ABBH,KF,INIT}} := \mathbf{\Phi_{SIMO,KF}}$
$\mathbf{\Phi_{ABBH,INFO,INIT}} := \mathbf{\Phi_{SIMO,INFO}}$

Goals Responder:
$\mathbf{\Phi_{ABBH,MA,RESP}} :=$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH2”}, INFO_Y, y, x, enc_0, mic_0) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH2”}, INFO_Y, y, x, enc_0, mic_0) <$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH3”}, INFO_X, x, y, enc_1, mic_1) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“ABBH3”}, INFO_X, x, y, enc_1, mic_1) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH4”}, y, x, mic_2)$
$\mathbf{\Phi_{ABBH,PTKD,RESP}} := \mathbf{\Phi_{SIMO,PTKD}}$
$\mathbf{\Phi_{ABBH,GTKD,RESP}} := \mathbf{\Phi_{SIMO,GTKD}}$
$\mathbf{\Phi_{ABBH,KF,RESP}} := \mathbf{\Phi_{SIMO,KF}}$
$\mathbf{\Phi_{ABBH,INFO,RESP}} := \mathbf{\Phi_{SIMO,INFO}}$

F.2 Proof Security Goals, SIMO

Generalized Authentication:
$\mathbf{AA1}, \mathbf{ARP}, \mathbf{AA4}, \mathbf{\Theta_{SIMO}}$
$[\mathbf{ABBH}:\mathbf{SIMO}]_X$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y) <$
$(\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1) \wedge$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH5”}, INFO_X, x, y, enc_0, mic_0))$ (124)

$\mathbf{ARP}, \mathbf{HASH3}, \mathbf{\Theta_{SIMO}}$
$[\mathbf{ABBH}:\mathbf{SIMO}]_X$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1) \supset$
$\exists Z.\text{Computes}(Z, \text{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y},$
$\quad \text{“ABBH5”}, INFO_Y, y, x, enc_1, INFO_X)) \wedge$
$\text{Sends}(Z, \text{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y},$

$\text{“ABBH5”}, INFO_Y, y, x, enc_1, INFO_X)) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1)$ (125)

$\Gamma_{\mathbf{ABBH,SI,1}}$HASH1
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Computes}(Z, \text{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y},$
$\text{“ABBH5”}, INFO_Y, y, x, enc_1, INFO_X)) \supset$
$\text{Has}(Z, ptk_{X,Y}) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y} \vee \hat{Z} = \hat{T}$ (126)

$125, 126, \mathbf{AA1}, \Gamma_{\mathbf{ABBH,1}}, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(Z, \text{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y},$
$\text{“ABBH5”}, INFO_Y, y, x, enc_1, INFO_X)) \supset$
$\hat{Z} = \hat{Y}$ (127)

$125, 127, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Computes}(Y, \text{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y},$
$\text{“ABBH5”}, INFO_Y, y, x, enc_1, INFO_X)) \wedge$
$\text{Send}(Y, \text{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y},$
$\text{“ABBH5”}, INFO_Y, y, x, enc_1, INFO_X))$ (128)

$128, \mathbf{HASH1}, \Theta_{\mathbf{ABBH}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{Has}(Y, ptk_{X,Y}) \wedge \text{Has}(Y,$
$\hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1)$ (129)

$128, 129, \phi_{\mathbf{HONESTY}}, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1)$ (130)

$125, 130, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1)$ (131)

$\mathbf{FS1}, \mathbf{AN3}, \theta_{\mathbf{ABBH}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{FirstSend}(X, x, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x)$ (132)

$132, \mathbf{FS2}, \theta_{\mathbf{ABBH}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x)$ (133)

$\mathbf{FS1}, \mathbf{AN3}, \theta_{\mathbf{ABBH}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{Honest}(\hat{Y}) \supset$
$\text{FirstSend}(Y, y, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y)$ (134)

$130, 134, \mathbf{FS2}, \theta_{\mathbf{ABBH}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y) <$
$\text{Rcve}(Y, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y)$ (135)

$124, 130, 131, 133, 135, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$(\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x)) \wedge$
$(\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y) <$
$\text{Rcve}(Y, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y)) \wedge$
$(\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1)) \wedge$
$(\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y) <$
$(\text{Rcve}(X, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1) \wedge$
$\text{Send}(X, \hat{Y}, \hat{X}, \text{“ABBH5”}, INFO_X, x, y, enc_0, mic_0))) \wedge$
$(\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y) <$
$\text{Rcve}(Y, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x) <$
$\text{Send}(Y, \hat{X}, \hat{Y}, \text{“ABBH5”}, INFO_Y, y, x, enc_1, mic_1))$ (136)

Proof ptk Delivery, SIMO:
$\mathbf{AA1}, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{Has}(X, ptk_{X,Y})$ (137)

$129, 137, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{Has}(X, ptk_{X,Y}) \wedge \text{Has}(Y, ptk_{X,Y})$ (138)

Proof gtk Delivery, SIMO:
$\mathbf{AA1}, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{Has}(X, gtk_Y)$ (139)

$\mathbf{AA1}, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
$\text{Rcve}(Y, SIMOX5) \supset \text{Has}(Y, enc_1)$ (140)

$138, 139, 140, \mathbf{DEC}, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
$\text{Rcve}(Y, SIMOX5) \supset$
$\text{Has}(X, gtk_Y) \wedge \text{Has}(Y, gtk_X)$ (141)

Proof Key Freshness, SIMO:
$ptk_{X,Y} = HASH_{pmk}(x, y)$ (142)

$\mathbf{FS1}, \mathbf{AN3}, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{Honest}(\hat{Y}) \supset \text{new } (\hat{Y}, y) \wedge$
$\text{FirstSend}(Y, y, \hat{X}, \hat{Y}, \text{“ABBH1”}, INFO_Y, y)$ (143)

$\mathbf{FS1}, \mathbf{AN3}, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{Honest}(\hat{X}) \supset \text{new } (\hat{X}, x) \wedge$
$\text{FirstSend}(X, x, \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x)$ (144)

$130, 142, 143, 144, \Theta_{\mathbf{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$

(new $(\hat{X}, x) \wedge x \subseteq ptk_{X,Y} \wedge$ new $(\hat{Y}, y) \wedge y \subseteq ptk_{X,Y}) \wedge$
FirstSend$(X, x, \hat{Y}, \hat{X}$, "ABBH1"$, INFO_X, x) \wedge$
FirstSend$(Y, y, \hat{X}, \hat{Y}$, "ABBH1"$, INFO_Y, y)$ (145)

Proof Info, SIMO:
$CS, pmkN = SELECT(INFO_X, INFO_Y)$ (146)

$\mathbf{AA1}, \mathbf{\Theta_{SIMO}}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
Has$(X, CS, pmkN)$ (147)

$\mathbf{129}, \mathbf{146}, \mathbf{147}, \mathbf{PROJ}\Theta_{SIMO}$
$[\mathbf{ABBH} : \mathbf{SIMO}]_X$
KOHonest$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$SELECT(INFO_X, INFO_Y) = CS, pmkN \wedge$
Has$(X, CS, pmkN) \wedge$ Has$(Y, CS, pmkN)$ (148)

G Secrecy Proof

G.1 Proof of SafeNets

NET0,
SafeNet$(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeNet$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeNet$(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$
$[]_X$
Honest$(\hat{X}) \supset$
SendsSafeMsg$(X, xxKey_X, \{priv_X, priv_T, xxKey_X\})$,
SendsSafeMsg$(X, ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\})$,
SendsSafeMsg$(X, gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$ (149)

SAF0
SafeNet$(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeNet$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeNet$(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$
[new x; send $\hat{Y}, \hat{X}$, "ABBH1"$, INFO_X, x]_X$
SafeMsg(ABBH1$, xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeMsg(ABBH1$, ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeMsg(ABBH1$, gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$ (150)

$\mathbf{150}, \mathbf{NET2}, \mathbf{NET3}$,
SafeNet$(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeNet$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeNet$(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$
[new x; send $\hat{Y}, \hat{X}$, "ABBH1"$, INFO_X, x]_X$
Honest$(\hat{X}) \supset$
SendsSafeMsg$(X, xxKey_X, \{priv_X, priv_T, xxKey_X\})$,
SendsSafeMsg$(X, ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\})$,
SendsSafeMsg$(X, gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$ (151)

NET2,
SafeNet$(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeNet$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeNet$(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$
[rcve $\hat{X}, \hat{Y}$, "ABBH2"$, INFO_Y, y, x, enc_0, mic_0$;
mtch $SELECT(INFO_X, INFO_Y)/CS, pmkN]_X$
Honest$(\hat{X}) \supset$
SendsSafeMsg$(X, xxKey_X, \{priv_X, priv_T, xxKey_X\})$,
SendsSafeMsg$(X, ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\})$,
SendsSafeMsg$(X, gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$ (152)

NET2,
SafeNet$(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeNet$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeNet$(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$
[mtch $RETRIEVE(pmkN)/pmk]_X$
Honest$(\hat{X}) \supset$
SendsSafeMsg$(X, xxKey_X, \{priv_X, priv_T, xxKey_X\})$,
SendsSafeMsg$(X, ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\})$,
SendsSafeMsg$(X, gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$ (153)

$\mathbf{SAF0}, \mathbf{SAF2}, \mathbf{SAF4}, \mathbf{SAF5}$,
SafeNet$(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeNet$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeNet$(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$
[mtch $HASH_{pmk}(x, y)/ptk_{X,Y}$;
mtch enc_0/ENC$_{ptk_{X,Y}}(gtk_Y)$;
mtch mic_0/HASH$_{ptk_{X,Y}}($
$\hat{X}, \hat{Y}$, "ABBH2"$, INFO_Y, y, x, enc_0)$;
mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}(\hat{Y}, \hat{X}$, "ABBH3",
$INFO_X, x, y, enc_1)$;
send $\hat{Y}, \hat{X}$, "ABBH3"$, INFO_X, x, y, enc_1, mic_1]_X$
SafeMsg(ABBH3$, xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeMsg(ABBH3$, ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeMsg(ABBH3$, gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$ (154)

$\mathbf{154}, \mathbf{NET2}, \mathbf{NET3}$,
SafeNet$(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeNet$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeNet$(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$
[mtch $HASH_{pmk}(x, y)/ptk_{X,Y}$;
mtch enc_0/ENC$_{ptk_{X,Y}}(gtk_Y)$;
mtch mic_0/HASH$_{ptk_{X,Y}}($
$\hat{X}, \hat{Y}$, "ABBH2"$, INFO_Y, y, x, enc_0)$;
mtch enc_1/ENC$_{ptk_{X,Y}}(gtk_X)$;
mtch mic_1/HASH$_{ptk_{X,Y}}(\hat{Y}, \hat{X}$, "ABBH3",
$INFO_X, x, y, enc_1)$;
send $\hat{Y}, \hat{X}$, "ABBH3"$, INFO_X, x, y, enc_1, mic_1]_X$
Honest$(\hat{X}) \supset$
SendsSafeMsg$(X, xxKey_X, \{priv_X, priv_T, xxKey_X\})$,
SendsSafeMsg$(X, ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\})$,
SendsSafeMsg$(X, gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$ (155)

NET2,
SafeNet$(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
SafeNet$(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge$
SafeNet$(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$
[rcve $\hat{X}, \hat{Y}$, "ABBH4"$, y, x, mic_2$;
mtch mic_2/HASH$_{ptk_{X,Y}}(\hat{X}, \hat{Y}$, "ABBH4"$, y, x)]_X$
Honest$(\hat{X}) \supset$
SendsSafeMsg$(X, xxKey_X, \{priv_X, priv_T, xxKey_X\})$,
SendsSafeMsg$(X, ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\})$,
SendsSafeMsg$(X, gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\})$ (156)

G.2 Proof Security Invariants Always Hold

POS
KOHonest$(priv_X, \{\}) \supset$
SafeNet$(priv_X, \{\}) \wedge$ Has$(Z, priv_X) \supset$
$\hat{Z} = \hat{X}$ (157)

$\mathbf{157, POS}$
$\text{KOHonest}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \supset$
$\text{SafeNet}(xxKey_X, \{priv_X, priv_T, xxKey_X\}) \wedge$
$\text{Has}(Z, xxKey_X) \supset \hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}$ (158)

$\mathbf{158, POS}$
$\text{KOHonest}(pmkmkd_X, \{xxKey_X\}) \supset$
$\text{SafeNet}(pmkmkd_X, \{xxKey_X\}) \wedge \text{Has}(Z, pmkmkd_X) \supset$
$\hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}$ (159)

$\mathbf{158, POS}$
$\text{KOHonest}(mkdk_{X,T}, \{xxKey_X\}) \supset$
$\text{SafeNet}(mkdk_{X,T}, \{xxKey_X\}) \wedge \text{Has}(Z, mkdk_{X,T}) \supset$
$\hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}$ (160)

$\mathbf{160, POS}$
$\text{KOHonest}(mptk_{X,T}, \{mkdk_{X,T}\}) \supset$
$\text{SafeNet}(mptk_{X,T}, \{mkdk_{X,T}\}) \wedge \text{Has}(Z, mptk_{X,T}) \supset$
$\hat{Z} = \hat{X} \vee \hat{Z} = \hat{T}$ (161)

$\mathbf{159, 161, POS}$
$\text{KOHonest}(pmk_{X,Y}, \{pmkmkd_X, mptk_{Y,T}\}) \supset$
$\text{SafeNet}(pmk_{X,Y}, \{pmkmkd_X, mptk_{Y,T}\}) \wedge \text{Has}(Z, pmk) \supset$
$\hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y} \vee \hat{Z} = \hat{T}$ (162)

$\mathbf{162, POS}$
$\text{KOHonest}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \supset$
$\text{SafeNet}(ptk_{X,Y}, \{pmk_{X,Y}, pmk_{Y,X}\}) \wedge \text{Has}(Z, ptk_{X,Y}) \supset$
$\hat{Z} = \hat{X} \vee \hat{Z} = \hat{Y} \vee \hat{Z} = \hat{T}$ (163)

$\mathbf{163, POS}$
$\text{KOHonest}(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\}) \supset$
$\text{SafeNet}(gtk_X, \{ptk_{X,Y_1}, \ldots ptk_{X,Y_n}\}) \wedge$
$\text{Has}(Z, gtk_X) \supset \text{Has}(Z, ptk_{X,Y_i})$ (164)

H Invariant Proofs

Proof that Invariants hold over INIT's Basic Sequences

Proof Invariants ABBH, INIT:
AA2,**Start**(X)
$[]_X$
$\neg(\text{Send}(X, m) \wedge$
$(\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH2”})) \vee$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH3”})) \vee$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH4”})) \vee$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH5”})))$
$\supset \Gamma_{ABBH,1}$ (165)

$\mathbf{AA5}, \mathbf{\Gamma_{ABBH,1}}$
$[\text{new } x;\ \text{send } \hat{Y}, \hat{X}, \text{“ABBH1”}, INFO_X, x]_X$
$\neg(\text{Send}(X, m) \wedge$
$(\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH2”})) \vee$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH3”})) \vee$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH4”})) \vee$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH5”})))$
$\supset \Gamma_{ABBH,1}$ (166)

$\mathbf{AA2}, \mathbf{\Gamma_{ABBH,1}}$
$[\text{rcve } \hat{X}, \hat{Y}, \text{“ABBH2”}, INFO_Y, y, x, enc_0, mic_0;$
$\text{mtch } SELECT(INFO_X, INFO_Y)/CS, pmkN]_X$
$\Gamma_{ABBH,1}$ (167)

$\mathbf{AA2}, \mathbf{\Gamma_{ABBH,1}}$
$[\text{mtch } RETRIEVE(pmkN)/pmk]_X$
$\Gamma_{ABBH,1}$ (168)

$\mathbf{AA1}, \mathbf{AA5}, \mathbf{\Gamma_{ABBH,1}}$
$[\text{mtch } HASH_{pmk}(x, y)/ptk_{X,Y};$
$\text{mtch } enc_0/\text{ENC}_{ptk_{X,Y}}(gtk_Y);$
$\text{mtch } mic_0/\text{HASH}_{ptk_{X,Y}}($
$\hat{X}, \hat{Y}, \text{“ABBH2”}, INFO_Y, y, x, enc_0);$
$\text{mtch } enc_1/\text{ENC}_{ptk_{X,Y}}(gtk_X);$
$\text{mtch } mic_1/\text{HASH}_{ptk_{X,Y}}(\hat{Y}, \hat{X}, \text{“ABBH3”},$
$INFO_X, x, y, enc_1);$
$\text{send } \hat{Y}, \hat{X}, \text{“ABBH3”}, INFO_X, x, y, enc_1, mic_1]_X$
$\text{Send}(X, m) \wedge$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH3”})) \wedge \hat{Z} = \hat{X} \wedge$
$(\neg(\text{Send}(X, m) \wedge$
$(\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH2”})) \vee$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH4”})) \vee$
$\text{Contains}(m, \text{Hash}_{ptk}((\hat{Z}, \hat{Y}), \text{“ABBH5”}))))) \supset$
$\Gamma_{ABBH,1}$ (169)

$\mathbf{AA2}, \mathbf{\Gamma_{ABBH,1}}$
$[\text{rcve } \hat{X}, \hat{Y}, \text{“ABBH4”}, y, x, mic_2;$
$\text{mtch } mic_2/\text{HASH}_{ptk_{X,Y}}(\hat{X}, \hat{Y}, \text{“ABBH4”}, y, x)]_X$
$\Gamma_{ABBH,1}$ (170)