Securing Address Registration in Location/ID Split Protocol using ID-Based Cryptography

Mahdi Aiash[1], Ameer Al-Nemrat and David Preston[2]

[1] School of Science and Technology, Middlesex University, London, UK
M.Aiash@mdx.ac.uk
[2] School of Architecture, Computing and Engineering, University of East London, London, UK
A.Al-Nemrat,D.Preston@uel.ac.uk

Abstract. The Locator/ID Separation Protocol (LISP) is a routing architecture that provides new semantics for IP addressing. In order to simplify routing operations and improve scalability in future Internet, the LISP separates the device identity from its location using two different numbering spaces. The LISP also, introduces a mapping system to match the two spaces. In the initial stage, each LISP-capable router needs to register with a Map Server, this is known as the Registration stage. However, this stage is vulnerable to masquerading and content poisoning attacks. Therefore, a new security method for protecting the LISP Registration stage is presented in this paper. The proposed method uses the ID-Based Cryptography (IBC) which allows the mapping system to authenticate the source of the data. The proposal has been verified using formal methods approach based on the well-developed Casper/FDR tool.

Keywords: Location/ID Split Protocol, Casper/FDR, Future Internet, Address Registration, Masquerading attacks

1 Introduction

Since the public Internet first became part of the global infrastructure, its dramatic growth has created a number of scaling challenges. Among the most fundamental of these is helping to ensure that the routing and addressing system continues to function efficiently even as the number of connected devices continues to increase. An IETF working group along with the research group at Cisco, are working on the Locator/ID Separation Protocol (LISP) [1]. Unlike IP addresses, which combine hosts' locations and devices IDs in a single namespace, the LISP separates hosts' locations and identities. The LISP specifies an architecture and mechanism for replacing the addresses currently used by IP with two separate name spaces: Endpoint IDs (EIDs), used within EID sites, and Routing Locators (RLOCs), used on the transit networks such as the Internet infrastructure. To achieve this separation, LISP defines protocol mechanisms for EID-to-RLOC mapping. Furthermore, LISP assumes the existence of a mapping system in the form of distributed database to store and propagate those mappings globally. The functionality of the mapping system goes through two stages:

1. Registration Stage: in this stage, the Map Server learns the EIDs-to-RLOC mappings from an authoritative LISP-Capable Router and publishes them in the database.
2. Addresses resolving Stage: the Map Server (MS) accepts Map-Requests from routers, looks up the database and returns the requested mapping.

These two stages will be explained in more details in section 2.2.

Currently, the research concentrates mainly on defining the LISP overall architecture as well as the structure of the LISP packets such as the Map-Register, Map-Notify and Map-Reply. However, the security-related research is still at an early stage, the research in [2] [3] have highlighted the potential threats to be addressed at a later stage of the research. Therefore, this paper investigates the security side of implementing the LISP architecture. Our main concern here is the security of the address Registration stage, where an LISP-capable router publishes all its hosts' EIDs to the Map Server via a Map-Register as will be discussed in section 1. For a secure Registration, two information elements are critical: the hosts' EIDs and the the router's address (RLOC). Indeed, a malicious router might spoof different RLOC and supply wrong EID- prefixes to the MS. This is very similar to poisoning attacks against Domain Name Server (DNS) or routing tables [6]. To stop such attacks, we need to thwart masquerading threats; therefore, a new approach based on the ID-Based Cryptography (IBC) [5] is proposed in this paper. The IBC helps to certify the messages sender as the real owner of the RLOC that will update the Map Server. The main advantage of using the IBC over traditional Public Key Infrastructure is that since the public key will be derived from the nodes' identifiers, IBC eliminates the need for a public key distribution infrastructure, more details are in section 2.3. The proposed solution is formally verified using formal methods approach based on the well-developed Casper/FDR tool [7].

The rest of the paper is organised as follows: Section 2 describes some related work in the literature. Section 3 presents the proposed security protocol along with its formal analysis. The paper concludes in Section 4.

2 Related Work

2.1 An Overview of The LISP

To improve routing scalability while facilitating flexible address assignment in multi-homing and mobility scenarios, the LISP describes changes to the Internet architecture in which IP addresses are replaced by routing locators (RLOCs) for routing through the global Internet and by endpoint identifiers (EIDs) for identifying network sessions between devices [8]. As shown in Fig 1, three essential components exist in the LISP environment: the LISP sites (EID space), the non-LISP sites (RLOC space), and the LISP Mapping System which comprises Map Servers and databases.

- **The LISP sites (EID space):** they represent customer end-sites in exactly the same way that end-sites are defined today. However, the IP addresses in the EID space are not advertised to the non-LISP sites, but are published into the LISP Mapping Systems which perform the EID-to-RLOC mapping. The LISP functionality is deployed on the site's gateway or edge routers. Therefore, based on their roles, two types of routers are defined: firstly, the Ingress Tunnel Routers (ITRs) which receive packets from hosts and send LISP packets toward the Map Server. Secondly, the Egress Tunnel Routers (ETRs) which receive LISP packets from the Map Server and pass them to hosts [1] [8].
- **Non-LISP sites (RLOC space):** they represent current sites where the IP addresses are advertised and used for routing purpose.
- **LISP Mapping Systems:** These are represented by Map Servers (MS) and a globally distributed database that contains all known EID prefixes to RLOC mappings. Similar to the current Domain Name System (DNS), the Mapping systems are queried by LISP-capable devices for an EID-to-RLOC mapping.

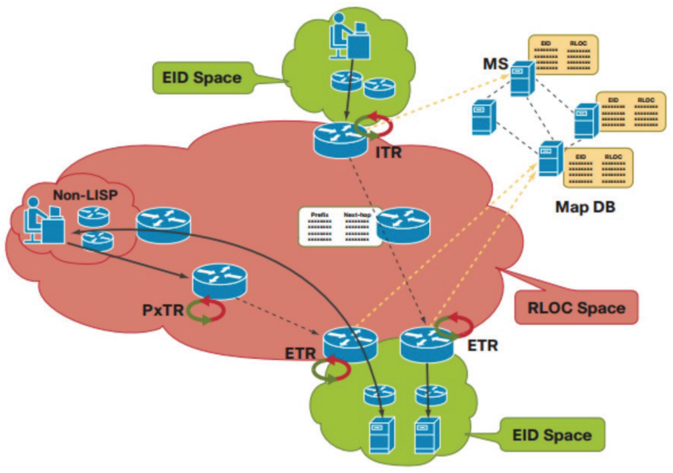


Fig. 1. The LISP Network Architecture Design [8]

2.2 Interactions With Other LISP Components

The functionality of the LISP goes through two stages:

1. **The EID Prefix Configuration and ETR Registration Stage:**
   As explained in [9], an ETR publishes its EID-prefixes on a Map Server (MS) by sending LISP Map-Register messages which include the ETR's RLOC and

a list of its EID-prefixes. Initially, it has been presumed that prior to sending a Map-Register message, the ETR and the Map Server must be configured with a shared secret or other relevant authentication information. Upon the receipt of a Map-Register from an ETR, the Map Server checks the validity of the Map-Register message and acknowledges it by sending a Map-Notify message. When registering with a Map-Server, an ETR might request a no-proxy reply service which implies that the Map Server will forward all the EID-to-RLOC mapping requests to the relevant ETR rather than dealing with them.
Since no security protocol has been proposed yet to authenticate the ETR and secure the connection with the MS, the registration stage, shown in Fig 2, is vulnerable to serious security threats such as replay and poisoning attacks. Therefore, a security protocol will be introduced in section 3.

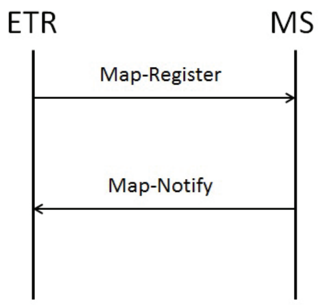


Fig. 2. The ETR Registration Process

2. **The Address Resolving Stage:** Once a Map Server has EID-prefixes registered by its client ETRs, it will accept and process Map-Requests. In response to a Map-Request (sent from an ITR), the Map Server first checks to see if the required EID matches a configured EID-prefix. If there is no match, the Map Server returns a negative Map-Reply message to the ITR. In case of a match, the Map Server re-encapsulates and forwards the resulting Encapsulated Map-Request to one of the registered ETRs which will return Map-Replay directly to the requestion ITR as shown in Fig 3.
The LISP working group in [1] has defined the structure of all the LISP Packets including the Map-Request, the Map-Notify, the Map-Register and the MAP-Reply. However, for the security analysis in section 3, only security-related parameters of the LISP messages are explicitly mentioned.

2.3 ID-Based Cryptography (IBC)

The IBC is a cryptographic scheme was first proposed by Adi Shamir [5]. The scheme enables users to communicate securely and verify each other's signature

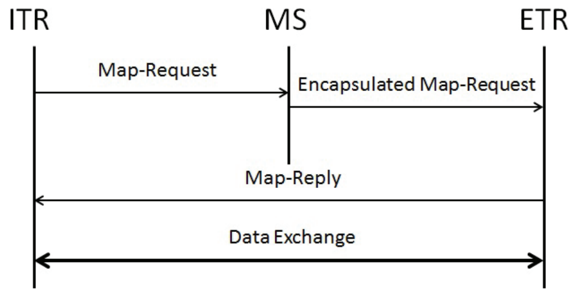


Fig. 3. The No Proxy Map Server Processing

without exchanging public or private keys. However, the scheme requires the presence of Trusted Key Generation (TKG) centres.

IBC's Operation: Unlike the normal Public Key Infrastructure (PKI) where a TKG randomly generates pairs of public/private keys, each node in IBC chooses its identifier (address or name) as a public key. Practically, any publicly known information that uniquely identifies the node could be used as a public key. The TKG generates the corresponding private key and securely distributes it to the node.

When a node (A) wants to communicate with another node (B), node A will sign the message using its private key and encrypt the result with the node B's public key. Upon receiving the message, node B will decrypt the message using its private key and verify the signature using node A's public key.

The IBC represents an efficient and easy to implement system which removes some of the overhead encountered in PKI for key management and digital certificate issuance/revocation. However, the security of the IBC is based on the secrecy of the private key. To deal with this issue, the node needs to combine additional information such as timestamps to their identifiers when generating the public key. This procedure will guarantee a periodic update of the public key. However, it introduces a key-management problem where all users must have the most recent public key for the node.

2.4 Verifying Security Protocols using Casper/FDR

Previously, analysing security protocols used to go through two stages. Firstly, modelling the protocol using a theoretical notation or language such as the CSP [10]. Secondly, verifying the protocol using a model checker such as Failures-Divergence Refinement (FDR) [11]. However, describing a system or a protocol using CSP is a quite difficult and error-prone task; therefore, Gavin Lowe [7] has developed the CASPER/FDR tool to model security protocols, it accepts a simple and human-friendly input file that describes the system and compiles it into CSP code which is then checked using the FDR model checker. Casper/FDR has been used to model communication and security protocols as in [12], [13]. The CASPER's input file that describes the systems consists of eight headers as explained in Table 1.

Table 1. THE HEADERS OF CASPER'S INPUT FILE

The Header	Description
# Free Variables	Defines the agents, variables and functions in the protocol
# Processes	Represents each agent as a process
# Protocol Description	Shows all the messages exchanged between the agents
# Specification	Specifies the security properties to be checked
# Actual Variables	Defines the real variables, in the actual system to be checked
# Functions	Defines all the functions used in the protocol
# System	Lists the agents participating in the actual system with their parameters instantiated
# Intruder Information	Specifies the intruder's knowledge and capabilities

3 The Proposed Solution

This section discusses our proposal of using the IBC protocol to secure the Registration procedure of the LISP.

3.1 System Definition

As shown in Fig 2, and based on the notations in Table 2, the secure Registration procedure using the IBC goes as follows:
`Msg1. TKG→ ETR : {SK(ETR)}{K1}`
`Msg2. TKG → MS : {SK(MS)}{K2}`

The TKG provides the two communicating parties (ETR, MS) with their private keys SK(ETR), SK(MS) in messages 1 and 2. These messages are encrypted using the pre-shared secret keys K1, K2, respectively.

`Msg3. ETR → MS : {Map-Register}{PK(MS)}, {h(Map-Register)}{SK(ETR)}`

The ETR sends an LISP Map-Register packets in Msg3. The content of this message is encrypted using the MS's public key (which is publicly known) and digitally signed using the private key of the ETR. As described in [9], the Map-Register packet includes the ETR's address (RLOC), a random number (n1) and a list of EID-Prefix, managed by the ETR.

```
Msg4. MS → ETR : {Map-Notify}{PK(ETR)}, {h(Map-Notify)}{SK(MS)}
```

Upon receiving msg3, the MS will use its private key SK(MS) to decrypt the message and then verify the signature using the ETR's public key PK(ETR). Finally, the MS will hash the included Map-Register and compare the result with the received signed value. Only if the two values are equal, the MS composes a Map-Notify packet as msg4 which includes the received random number (n1). This message is encrypted using the ETR's public key and digitally signed using the MS's private key. The ETR will check the included random number and only when the check succeeds, the ETR authenticates the MS.

Table 2. Notation

The Notation	Definition
TKG	The Trusted Ticket Granting
SK(ETR), SK(MS)	The Private keys of the ETR, MS, respectively. These keys are derived by the TKG
K1, K2	Pre-shared keys to secure the connections between the TKG and ETR, MS
ETR	The Egress Tunnel Router in the destination EID Space
MS	The Map Server
n1	A fresh random number
h(m)	Hash value of the message (m)
{m}{K}	The message (m) being encrypted with the key (K)

3.2 Formal Analysis Using Casper/FDR

To formally analyse the proposed solution, we simulate the system using Casper/FDR tool. The full Casper input file describing the system is included in the Appendix. For conciseness, only the #Protocol Description, the #Specification and the #Intruder Information headings are described here, while the rest are of a less significance in terms of understanding the verification process.

The #Protocol description heading defines the system and the transactions between the entities. It is worth pointing out that for security simulation we need to explicitly define the security parameters. Therefore, we mention the security-related contents of the Map-Register and Map-Notify as shown below in msg 3, 4. Where (m) and (m1) refer to Map-Register and Map-Notify packets, respectively. EIDPre refers to the EID-Prefix sent by the ETR.

```
#Protocol description
0. -> ETR : MS, TKG
1. TKG -> ETR : {SK(ETR)}{K1}
2. TKG -> MS : {SK(MS)}{K2}
```

```
3. ETR -> MS : {m,ETR, n1,EIDPre}{PK(MS)}, {h(m, ETR, n1, EIDPre)}{SK(ETR)}%z
```
$[decryptable(z, PK(ETR))]$
```
4. MS -> ETR : {m1, n1}{PK(ETR)}, {h(m1, n1)}{SK(MS)}%w
```
$[decryptable(w, PK(MS))]$

The security requirements of the system are defined under the # Specification heading. The lines starting with the keyword **Secret** define the secrecy properties of the protocol. The `Secret(MS, n1, [ETR])` specifies the n1 nonce as a secret between the ETR and the MS. The lines starting with the **Agreement** define the protocol's authenticity properties; for instance `Agreement(MS, ETR, [n1])` specifies that, the MS is correctly authenticated to the ETR using the random number n1. The `WeakAgreement(ETR, Ms)` assertion could be interpreted as follows: if ETR has completed a run of the protocol with MS, then MS has previously been running the protocol, apparently with ETR.

```
#Specification
Secret(MS, n1, [ETR])
WeakAgreement(ETR, MS)
WeakAgreement(MS, ETR)
Agreement(MS, ETR, [n1])
```

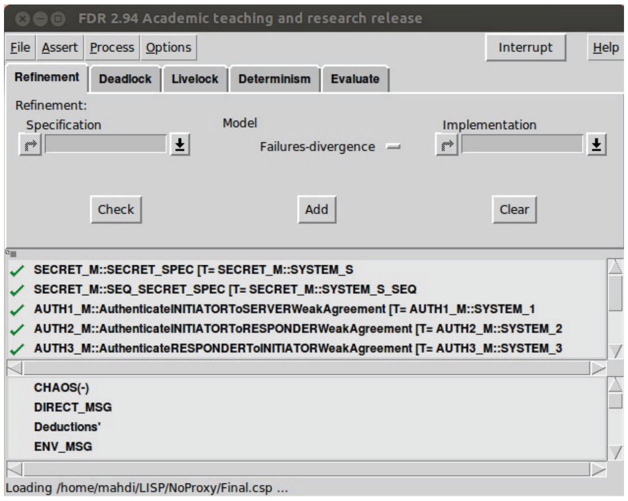
The # Intruder Information heading specifies the intruder's identity, knowledge and capability. The first line identifies the intruder as Mallory, the intruder knowledge defines the Intruder's initial knowledge, i.e., we assume the intruder knows the identity of the participants, its own private key and can fabricate Map-Register and Map-Notify messages.

```
#Intruder Information
Intruder = Mallory
IntruderKnowledge = {ETR, MS, Mallory, PK ,SK(Mallory), Map-Register,
Map-Notify}
```

After generating the CSP description of the systems using Casper and asking FDR to check the security assertions, no attack was found against the proposed solution as shown in Fig 4.

Security Considerations Despite the fact that no attack has been discovered against the proposed solution in section 3.2, this result needs to be considered carefully. The formal verification result is based on the system defined in 3.1. In this system, it is assumed that the ETR knows the authoritative MS in its network or domain. In a very similar way to the current Domain Naming System (DNS), where clients are preconfigured with the authoritative DNS server. However, we simulated the case when the ETR is not sure of the identity of its authoritative MS. The following attack against the `Secret(MS, n1, [ETR])`, `Agreement(MS, ETR, [n1])` and `WeakAgreement(ETR, MS)` assertions was discovered

```
0. -> ETR : Mallory, TKG
1a. TKG -> I_ETR : {SK(ETR)}{K1}
```



Failures-divergence

```
✓ SECRET_M::SECRET_SPEC [T= SECRET_M::SYSTEM_S
✓ SECRET_M::SEQ_SECRET_SPEC [T= SECRET_M::SYSTEM_S_SEQ
✓ AUTH1_M::AuthenticateINITIATORToSERVERWeakAgreement [T= AUTH1_M::SYSTEM_1
✓ AUTH2_M::AuthenticateINITIATORToRESPONDERWeakAgreement [T= AUTH2_M::SYSTEM_2
✓ AUTH3_M::AuthenticateRESPONDERToINITIATORWeakAgreement [T= AUTH3_M::SYSTEM_3

CHAOS(-)
DIRECT_MSG
Deductions'
ENV_MSG

Loading /home/mahdi/LISP/NoProxy/Final.csp ...
```

Fig. 4. The FDR Formal Verification

```
1b. I_TKG -> ETR : {SK(ETR)}{K1}
2a. TKG -> I_MS : {SK(MS)}{K2}
2b. I_TKG -> MS : {SK(MS)}{K2}
3a. ETR -> Mallory : {M, ETR, n1, EIDpre}{PK(Mallory)}, {h(M, ETR, N1,
EIDpre)}{SK(ETR)}
3b. I_ETR -> MS : {M, ETR, n1, EIDpre}{PK(MS)}, {h(M, ETR, n1, EIDpre)}
{SK(ETR)}
4. MS -> I_ETR : {M2, n1}{PK(ETR)}, {h(M2, n1)}{SK(MS)}
```

Where the notations I_ Ms, I_ETR and I_TKG represent the case where the Intruder impersonates the Ms, ETR and TKG, respectively. This is an active Man-in-the-Middle attack; the Intruder intercepts and replays messages 1 and 2. Since the ETR is not sure of the identity of the MS, the intruder manages to impersonate the MS and fools the ETR to use its (rather than the MS's) public key to encrypt message 3a. Consequently, the random number (n1) will be compromised, and the ETR will run the protocol mistakenly believing it is with the MS, while in reality it is with the Intruder. In order to stop such attacks, the ETRs should be configured to use the authoritative Map Server in its domain or network. This could be simply achieved during the network configuration in a similar way to configuring the default DNS server or the default Gateway in a network.

4 Conclusion

This paper analysed the security of the address Registration in the LISP protocol. We presented a new security method, based on the IBC, allowing a Map

Server to check the received information (i.e., the EID-Prefix) as well as providing source authentication. The proposed solution has been verified using formal methods approach based on the Casper/FDR tool.

References

1. Farinacci, D., Fuller, V., Meyer, D., Lewis, D.: Locator/ID Separation Protocol (LISP). Internet-Draft , November 13, 2012.
2. Cisco Locator/ID Separation Protocol Security At-A-Glance.`http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6554/ps6599/ps10800/at_a_glance_c45-645204.pdf`. [Last Accessed on 13.01.13].
3. Maino, F., Ermagan, V., Cabellos, A., Saucez, A., Bonaventure, O.: LISP-Security (LISP-SEC). Internet-Draft , September 12, 2012.
4. Maino, F., Ermagan, V., Cabellos, A., Saucez, A., Bonaventure, O.: LISP-Security (LISP-SEC). Internet-Draft , September 12, 2012.
5. Shamir,A., Identity-based cryptosystems and signature schemes, in Proceedings of CRYPTO 84 on Advances in cryptology. New York, NY, USA: Springer-Verlag New York, Inc., 1985, pp. 4753.
6. Arends, R., Austein, R., Larson, M., D. Massey, D., Rose, S.: DNS Security Introduction and Requirements, Internet Engineering Task Force, RFC 4033, Mar. 2005
7. Lowe, G., Broadfoot, P., Dilloway, C., Hui, M. L.: Casper: A compiler for the analysis of security protocols, 1.12 ed., September 2009.
8. Cisco Locator/ID Separation Protocol Revolutionary Network Architecture to Power the Network. `http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6554/ps6599/ps10800/aag_c45-635298.pdf`. [Last Accessed on 13.01.13].
9. Farinacci, D., Fuller, V.: LISP Map Server Interface. Internet-Draft , March 4, 2012.
10. Goldsmith, M., Lowe, G., Roscoe, A.W., Ryan,P., Schneider, S.: The modelling and analysis of security protocols, PEARSON Ltd, 2010.
11. Formal Systems, Failures-divergence refinement. fdr2 user manual and tutorial, June 1993, Version 1.3.
12. Aiash, M., Mapp, G., Lasebae, A., Phan, P., Loo, J.: Casper: A formally verified AKA protocol for vertical handover in heterogeneous environments using Casper/FDR, EURASIP Journal on Wireless Communications and Networking 2012, 2012:57.
13. Aiash, M., Mapp, G., Lasebae, A., Phan, P., Loo, J.: A Formally Verified Device Authentication Protocol Using Casper/FDR. In 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 25-27 June 2012.

Appendix: The Casper description of the proposed solution

```
#Free variables
ETR, MS : Agent
na, nb, seq2, n1 : Nonce
```

```
PK: Agent -> PublicKey
SK: Agent -> PrivateKey
K1, K2: PreSharedKey
TKG: Server
m,m2, Ack: Messages
InverseKeys = (PK,SK), (K1, K1),(K2, K2)
h : HashFunction
EIDPre: EIDPrefix
hash1: HashValues
#Processes
INITIATOR(ETR, MS,TKG, K1,nb, m, Ack, n1, EIDPre)knows PK(ETR), PK(MS),
SK(ETR)
RESPONDER(MS,TKG, K2, m2) knows PK(ETR), PK(MS), SK(MS)
SERVER(TKG, ETR, MS, K1, K2, na) knows PK, SK(ETR), SK(MS)
#Protocol description
0. -> ETR : MS, TKG
1. TKG -> ETR : {SK(ETR)}{K1}
2. TKG -> MS : {SK(MS)}{K2}
3. ETR -> MS : {m,ETR, n1,EIDPre}{PK(MS)}, {h(m, ETR, n1, EIDPre)}{SK(ETR)}%z
[decryptable(z, PK(ETR))]
4. MS -> ETR : {m2, n1}{PK(ETR)}, {h(m2, n1)}{SK(MS)}%w
[decryptable(w, PK(MS))]
#Specification
WeakAgreement(ETR, MS)
Secret(MS, n1, [ETR])
WeakAgreement(MS, ETR)
Agreement(MS, ETR, [n1])
#Actual variables
etr, ms, Mallory : Agent
Na, Nb, Seq2, N1 : Nonce
k1, k2: PreSharedKey
tkg: Server
InverseKeys = (k1, k1),(k2, k2)
EIDpre: EIDPrefix
M, M2, ack: Messages
haash1: HashValues
#Functions
symbolic SK, PK
#System
INITIATOR(etr,ms, tkg, k1, Nb, M, ack, N1, EIDpre)
RESPONDER(ms,tkg,k2,M2)
SERVER(tkg, etr, ms, k1,k2, Na))
#Intruder Information
Intruder = Mallory
```

```
IntruderKnowledge = {etr, ms, Mallory, PK ,SK(Mallory), M, M2}
```