Provably Insecure Mutual Authentication Protocols: The Two-Party Symmetric-Encryption Case

Jim Alves-Foss
Center for Secure and Dependable Software
University of Idaho
Moscow, ID 83844-1010

Abstract

In practice, users will rely on a wide variety of communication protocols to conduct their work over the Internet. This paper discusses the security ramifications of using multiple authentication protocols. We demonstrate multi-protocol attacks and how they can be realized to defeat otherwise secure authentication protocols. We highlight this discussion with examples of attacks on a proposed symmetric key-based authentication protocols. We present a model of communication that reflects the existence of this type of attack, and demonstrate that a class of authentication protocols can never be secure in the presence of this type of attack.

1 Introduction

The widespread use of electronic commerce and other networked applications that require a high level of authentication will benefit from the establishment of secure authentication and key distribution protocols. With such protocols in place, it will be easy for users to establish trusted, secure communication with other participants. Unfortunately, although a large number of authentication protocols have been designed, a very large percentage of them have been subsequently found to be flawed and insecure. This has resulted in efforts by many researchers to develop techniques for analyzing these protocols for security flaws. One such technique, introduced by Bellare and Rogaway [BR93], uses the concepts of provable security [GMW84, BM84, Yao82] to analyze authentication protocols.

Given the verification techniques that have been developed, several "secure" protocols have been created. Unfortunately there is a class of attacks, *multi-protocol attacks* that are successful against a large class of these "secure" authentication protocols. These attacks result from the fact that end users of the protocol (called players) can share keys between multiple protocols [KSW97, AF98, TH99]. Thus the installation of an insecure or possibly *tailored protocol* (called chosen protocols in [KSW97]) on player B's machine could result in an attacker being able to masquerade as B to other players.

For the purposes of this paper, we define a symmetric-key (shared key) authentication protocol as any protocol that relies on the use of symmetric-key encryption to validate the identity of protocol

participants. We assume that the definition of the protocols and the generation of symmetric key pairs is disjoint and that a symmetric-key can be used in more than one protocol.[1] We concede the security of the symmetric key within the user's machine.

In a secure environment, composition of secure components may result in unforeseen difficulties. As data aggregation may result in data that has a higher security classification than any of its parts, McCullough showed that the composition of secure systems may result in a system that is less secure that its constituent parts [McC88]. This paper discusses another type of composition, the composition of multiple protocols in a real-world environment. We demonstrate multi-protocol attacks and how they can be realized to defeat otherwise secure authentication protocols. We highlight this discussion with examples of attacks on a proposed symmetric key-based authentication protocols, other examples can be found in [KSW97, AF98, TH99]. We present a model of communication that reflects the existence of this type of attack, and demonstrate a class of authentication protocols that can never be secure in the presence of this type of attack.

2 Formal Model

Bellare and Rogaway present a formalization of cryptographic protocols and a formal model for adversaries [BR93], henceforth termed BR protocols and models. Bellare and Rogaway have used this formalization, and variants of it, in the specification and verification of a series of *provably secure* protocols [BR93, BR95]. In addition, others have extended this work in many areas [SR96, Luc97, BWM97, BWJM97]. In this section we extend the BR model to include interactions between multiple protocols.

2.1 Protocol Specifications

The BR protocols are defined as an efficiently computable function Π on the following inputs [BR93]:

1^k	—	security parameter
$i \in I$	—	identity of sender
$j \in I$	—	identity of (intended) partner
$a \in \{0,1\}^*$	—	secret information of i – typically a long lived key
$conv \in \{0,1\}^*$	—	conversation so far
$r \in \{0,1\}^\omega$	—	random coin flips

where the value returned by $\Pi(1^k, i, j, a, conv, r)$ is the triple (m, δ, σ) consisting of

$m \in \{0,1\}^* \cup \{*\}$	—	messages sent to j
$\delta \in \{A, R, *\}$	—	the decision
$\sigma \in \{0,1\}^* \cup \{*\}$	—	private output such as a distributed session key

where the private output is string-valued only when the decision is A (accept). If the decision is to reject the exchange, R, or no decision has yet been made, the private output is *.

[1] This is not an unreasonable assumption when considering cases where a symmetric key may be based on a password, when using a mechanism where the key is valid for fixed limited time period, or when using prearranged or fixed security association.

2.2 Communication Model

This section presents a brief introduction to the communication model presented by Bellare and Rogaway [BR93]. This model makes the powerful assumption that *all* communication passes through the adversary. In other words, the adversary has complete control of the network; creating, modifying, relaying, delaying and deleting messages at will. Given such a powerful adversary, protocols proven secure under this model should be secure under most real-world implementations.

In this model of communication, all active participants (other than the adversary Eve) are called *players*. The notation $\Pi^s_{i,j}$ denotes player i communicating with remote player j using the specified protocol, Π. The superscript s denotes a specific instance of the protocol run, a *session*. All sessions, whether occurring concurrently or over time, are considered separate. The only information shared between sessions would be long-lived entities such as keys.

If players i and j are communicating in their respective sessions s_i and s_j, a *benign* adversary will relay all messages between $\Pi^{s_i}_{i,j}$ and $\Pi^{s_j}_{j,i}$. This models normal communication, and is necessary since we assume that the adversary has complete control of the communication medium. Given this assumption, we are implicitly assuming that the act of the adversary relaying messages does not constitute an attack.

A non-benign adversary has at her disposal an infinite collection of *oracles*[2], $\Pi^s_{i,j}$ for all s, i and j. Eve has the ability to start any of these oracles at any time, can send messages to any oracle, and can relay any of the oracles' messages to any other oracle. From the responses of the oracles, Eve learns the message and whether or not the oracle accepted or rejected. However, Eve does not learn the oracle's private output (except under certain circumstances to model Eve penetrating a player's computer, called *opening* the oracle).

A provably secure protocol is one that results in Eve having a negligible probability of successful attack against a session in which she has not explicitly opened either player's oracle. This is modeled through the concept of *matching conversations*, where conversations of the two honest players are synchronized. Specifically, the only way a player will decide to accept is if there is a matching player and all communication between players has been relayed as if Eve was benign.

The BR communication model allows for a very powerful adversary, avoiding many of the hidden assumptions present in the design of some published protocols. Their claim is that the security of protocols analyzed through their model is based on the minimal assumption that pseudo random functions (PRF) exist. This assumption permits mathematical analysis of the security of protocols, resulting in specific probabilities of attackers subverting the protocols. That is, these probabilities are correct if the model is a correct model of the environment in which the protocol will execute and it addresses the goals of the protocol [Bel98].

In the following section we present an attack against the "provably secure" BR protocol MAP1 [BR93]. This attack is an instance of a *multi-protocol attack* [AF98], also called *chosen-protocols attacks* [KSW97]. This class of attacks works against this protocol unless we make one of the following assumptions:

1. The adversary may only interact with sessions of the same protocol and may not concurrently interact with sessions of different protocols. This is the assumption of the BR model, but

[2] These oracles differ from traditional oracles in that they maintain state to model active players in the environment.

1. $A \rightarrow B : R_A$
2. $B \rightarrow A : \{B.A.R_A.R_B\}_a$
3. $A \rightarrow B : \{A.R_B\}_a$

Figure 1: MAP1 Protocol

Protocol EVE1	*Protocol EVE2*
1. $A \rightarrow B : R_A$	1. $A \rightarrow B : \{B.A.R_A.R_B\}_a$
2. $B \rightarrow A : \{A.B.R_A.R_B\}_a$	2. $B \rightarrow A : \{A.R_B\}_a$
3. $A \rightarrow B : \{A.R_B\}_a$	

Figure 2: Attack Protocols

does not accurately reflect reality, unless there is a mechanism to guarantee that a player can identify which protocol was used to generate a received message. No such mechanism has been presented in the literature.

2. The keys used for authentication are specifically tied to a particular protocol and cannot be used for any other protocol. The literature has not yet presented a secure mechanism to implement such a restriction.

Note that we are not attacking the underlying cryptographic primitives used by the protocol, and therefore are not disavowing the PRF assumption. What we are doing is modifying the model under which the protocols were proved correct. This new model weakens the assumptions in the BR model by permitting the adversary access to a wide collection of possible protocols, instead of limiting the adversary to the single protocol being analyzed. This model reflects the reality of players potentially using multiple protocols required by different applications, with the same key.

3 Breaking a Provably Secure Protocol

Bellare and Rogaway present the provably secure protocol, MAP1 [BR93] for mutual authentication of two parties. A brief outline of the protocol is presented in Fig. 1. Consider protocol EVE1 (see Fig. 2), where message 2 differs from message 2 in MAP1 by transposing the A and the B. This protocol can also be proven secure using the BR model.

Figure 3 outlines one possible attack against MAP1 using EVE1. In this attack, Eve, masquerading as B (denoted E_B) intercepts messages from A in protocol MAP1 and relays them as messages in protocol EVE1 back to A. Eve then waits for A's response, which she then returns back to A in the MAP1 protocol. B can be offline, and this attack will still work, as long as A has installed the two protocols.

Figure 4 outlines another possible attack against MAP1 using EVE2. In this attack, Eve, masquerading as A (denoted E_A) intercepts messages from B in protocol MAP1 and relays them as messages in protocol EVE2 back to B. Eve then waits for B's response, which she then returns back to B in the MAP1 protocol. A can be offline, and this attack will still work, as long as B has installed the two protocols.

Protocol MAP1	*Protocol EVE1*
1. $A \rightarrow E_B : R_A$	
	1. $E_B \rightarrow A : R_A$
	2. $A \rightarrow E_B : \{B.A.R_A.R_{A'}\}_a$
2. $E_B \rightarrow A : \{B.A.R_A.R_{A'}\}_a$	
3. $A \rightarrow E_B : \{A.R_{A'}\}_a$	

Figure 3: Attack against A in MAP1 protocol, with B offline

Protocol MAP1	*Protocol EVE2*
1. $E_A \rightarrow B : R_A$	
2. $B \rightarrow E_A : \{B.A.R_A.R_B\}_a$	
	1. $E_A \rightarrow B : \{B.A.R_A.R_B\}_a$
	2. $B \rightarrow E_A : \{A.R_B\}_a$
3. $E_A \rightarrow B : \{A.R_B\}_a$	

Figure 4: Attack against B in MAP1 protocol, with A offline

The purpose of this demonstration is to show that we can take different pairs of protocols, even if they are both provably secure, and generate a successful attack against one of them. We have also generated other attacks where Eve interacts with both players, each running a different protocol. Many possible attacks exists using multi-protocol techniques [AF98, KSW97, TH99]. Unfortunately, manually generating these attacks does not help the situation. What we need is an understanding of the properties of a protocol that make it either secure or insecure. In the following sections we take a step towards this understanding in the case of two-party mutual authentication protocols using symmetric-key encryption.

4 A Modified Communication Model

The following communication model uses the same protocol specification format as the BR model and follows the format of the BR communication model. The difference lies in the fact that we allow the adversary to interact with oracles modeling *all* protocols with a particular message format.

The notation $\Pi(n)^s_{i,j}$ denotes player i communicating with player j using the specified protocol, $\Pi(n)$. As with the BR model, the superscript s denotes a specific instance of the protocol run, a *session*, and all sessions are considered separate. The only information shared between sessions and protocols would be long-lived entities such as keys.

If players i and j are communicating in their respective sessions s_i and s_j, a *benign* adversary will relay all messages between $\Pi(n)^{s_i}_{i,j}$ and $\Pi(n)^{s_j}_{j,i}$. This models normal communication, and is necessary since we assume that the adversary has complete control of the communication medium. As in the BR model, relaying is not considered an attack.

A non-benign adversary has at her disposal an infinite collection of *oracles*, $\Pi(n)^s_{i,j}$ for all s, n, i and j. Eve has the ability to start any of these oracles at any time, can send messages to any oracle, and can relay any of the oracles' messages to any other oracle, even between protocols.

From the responses of the oracles, Eve learns the message and whether or not the oracle accepted or rejected. However, Eve still does not learn the oracle's private output.

4.1 Proofs Using the Model

In the BR model of communication, proofs about the security of the protocols relies on the form of messages in the protocol. Since we permit the adversary to have access to a wide range of protocols, we must permit proofs to reason about the form of messages within any of these protocols.

For the purposes of this paper we will limit our discussion to a subset of two-party authentication protocols using symmetric encryption, whose messages do not contain nested encryption. Specifically, all messages sent between players will consist of messages, $m \in M$ where the set M is defined by:

- all $m \in \{0,1\}^*$ are messages in M, these may include player identifiers, nonces or other message text. They may not include any messages that are considered encrypted by the authenticating players.
- if $m, n \in M$ then $m.n \in M$ where $m.n$ denotes composition of messages m and n and each of m and n are considered *fields* of the message.
- if $m \in M$, and is not an encrypted message, then $\{m\}_k$ is a message denoting the encipherment of m using symmetric key k.

In the BR model, all messages are represented as unformatted strings of bits. Although this is a very general model, it prohibits reasoning about the interpretation players give to those bits in terms of fields of the message and to the wide range of message formats used in different protocols. In our model we define messages as consisting of a collection of fields (represented by the concatenation of messages, each of which is a string of bits) and additionally a message may be encrypted with a shared key.

Proofs using the BR model focus on critical messages that, if accepted by a player, will result in that player accepting the communication. Secure protocols reach this final stage only with benign adversaries. The proofs demonstrate that such critical messages can only come from a matching player and not from the adversary interacting with another oracle. Proofs using our model follows the same approach, except that the adversary is not restricted to only using messages of the attacked protocol.

5 Proof of Insecurity

In this section we present an outline of a proof for the following theorem:

Theorem: *All two-party mutual authentication protocols using symmetric key encryption and the restricted message format specified above are not secure.*

Before we proceed with the proof, we need to discuss how authentication protocols work, and how the protocol players reach a belief in the identity of the other player. In the case of symmetric-key encryption systems, the proof of current knowledge of the symmetric key is often sufficient to prove identity. This typically results in protocols such as MAP1, where there is an exchange of encrypted messages containing random nonces. These values are used, as in MAP1, to avoid replay attacks by validating to the players that the other has recently used the symmetric-key to encrypt or decrypt these values. This *validation information* is individually denoted as fields in our communication model.

For our proof we assume that the adversary has prior knowledge of the protocol to be attacked and the players that use this protocol. The adversary then either designs or chooses a specific attack protocol to use in a multi-protocol attack. We assume that the adversary is able to get this protocol installed on at least one of the player's machines.

Consider an authentication protocol written with messages following our message format. For a player, A, to accept a communication, the last message they receive must be either:

1. encrypted - containing a nonce or other information that validates the recency of the encryption.
2. not encrypted - containing a previously encrypted value that the other player had to successfully decipher.

Case 1. In case 1, encrypted message, there are two possible subcases to consider. The first subcase, 1.1, is that the validation information was sent in a previously unencrypted message. In this case, the adversary generates a protocol that enables an oracle to take the unencrypted message and then generate an encrypted message of the form required by the accepting player, as we do in protocol EVE1. The second subcase, 1.2, is that the validation information was sent in a previously encrypted message. In this case, the adversary generates a protocol that enables an oracle to take the encrypted message, decipher it and then generate an encrypted message of the form required by the accepting player, as we do in protocol EVE2.

Formally, we are saying that the final message, m received by the accepting player, A, is of the form $m = \{m_1.m_2 \ldots m_m\}_K$, where some subset, V, of the m_i's are used for validation.

Case 1.1. For each $m_i \in V$ used for validation, there exists a previous message, m' of the form $m'_1.m'_2 \ldots .m_i \ldots m'_{n'}$ that was sent by A. Eve can generate a protocol, Π_{eve}, that takes any portion of this message (since it was transmitted unencrypted) and generates message m. If the m_i's were sent in multiple messages, then Π_{eve} will use data from the multiple prior messages.

Case 1.2. For each $m_i \in V$ used for validation, there exists a previous message, m' of the form $\{m'_1.m'_2 \ldots .m_i \ldots m'_{n'}\}_K$ that was sent by A. Eve can generate a protocol, Π_{eve}, that takes this and generates message m. If the m_i's were sent in multiple messages, then Π_{eve} will require receipt of all of these message to generate m.

In either case, or in a combination of these cases, Eve is able to construct a protocol Π_{eve} that can successfully attack the target protocol.

Case 2. In case 2, unencrypted message, there is only the situation that the validation information was previously transmitted in an enciphered message (otherwise straight eavesdropping is possible). Using a technique similar to protocol EVE2, the attacker creates a protocol that enables an oracle to accept messages in the format of the enciphered message, decipher it and then respond with a plaintext version of the enciphered validation information field. The adversary can take this plaintext message and create a message of the form required by the accepting player.

Formally, we are saying that the final message, m received by the accepting player, A, is of the form $m = m_1.m_2 \ldots m_m$, where some subset, V, of the m_i's are used for validation. For each $m_i \in V$ used for validation, there exists a previous message, m' of the form $\{m'_1.m'_2 \ldots .m_i \ldots .m'_{n'-1}.m'_{n'}\}_a$ that was sent by A. Eve can generate a protocol, Π_{eve}, that takes this and generates message m. If the m_i's were sent in multiple messages, then Π_{eve} will require receipt of all of these message to generate m.

Therefore, regardless of the form of the final message, or the form of previous messages containing the validation information, Eve can create protocol Π_{eve} and use it to attack the target protocol using a multi-protocol attack.

6 Conclusion

In this paper, we have discussed the use of multi-protocol attacks against two-party mutual authentication protocols using symmetric-key encryption. We have shown, by example, how these attacks work. We have presented a model of communication that takes into consideration the existence of multi-protocol attacks, and permits a powerful adversary to use them. We also present a proof outline for the theorem that all two-party mutual authentication protocols using symmetric key encryption and a limited message format are insecure.

Although we limited our discussion in this paper to authentication protocols, there exists the even more important concept of key-distribution, often coupled with authentication. Most of the verification techniques can be applied to key distribution protocols as well. However, further work is needed in the context of the research presented here before we address key distribution protocols.

This paper presented the initial stages of a larger research project where we are investigating several classes of protocols under the assumptions of multi-protocol attacks. These classes include public-key authentication protocols [AF98] and both shared-key and public-key key-distribution protocols. From our preliminary results it appears that the multi-protocol attack is valid against a very wide range of these protocols, if not all of them. The protocol design practices discussed by Kelsey et. al. [KSW97] do not alleviate the problem (as discussed in [AF98]). As a matter of fact, all design practices, such as including protocol identification numbers, direction bits, or checksums, are not sufficient. Each of these practices only works if all protocols use them. The power of the multi-protocol attack is that it is sufficient against otherwise secure protocols by using a tailored protocol designed specifically to defeat the protocol that is under attack. The tailored protocol does not follow the design rules of secure protocol, and as such can use the same protocol numbers and other identification schemes as the secure protocols. The only security against multi-protocol attacks is to either restrict the use of a key to a specific protocol (or family of secure protocols), to establish a mechanism whereby only secure (and certified) protocols can be used on the end-user's machine, or to develop an unforgeable mechanism to uniquely identify for which protocol the

encryption was performed.

References

[AF98] J. Alves-Foss. Multi-protocol attacks and the public key infrastructure. In *Proc. National Information System Security Conference*, pages 566–576, October 1998.

[Bel98] M. Bellare. Practice-oriented provable-security. In *Proc. First International Workshop in Information Security (ISW97)*, 1998.

[BM84] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13(4):850–864, November 1984.

[BR93] M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Advances in Cryptology – Crypto 93 Proceedings*, pages 232–249, 1993.

[BR95] M. Bellare and P. Rogaway. Provably secure session key distribution – the three party case. In *Proc. 25th Annual Symposium on the Theory of Computing*, pages 57–66, 1995.

[BWJM97] S. Blake-Wilson, D. Johnson, and A. Menezes. Key exchange protocols and their security analysis. In *Proc. 6th IMA International Conference on Cryptography and Coding*, 1997.

[BWM97] S. Blake-Wilson and A. Menezes. Entity authentication and authenticated key transport protocols employing asymmetric techniques. In *Proc. 1997 Security Protocols Workshop*, pages 137–153, 1997.

[GMW84] S. Goldwasser, S. Micali, and A. Wigderson. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, April 1984.

[KSW97] J. Kelsey, B. Schneier, and D Wagner. Protocol interactions and the chosen protocol attack. In *Proc. 1997 Security Protocols Workshop*, pages 91–104, 1997.

[Luc97] S. Lucks. Open key exchange: How to defeat dictionary attackes without encrpyting public keys. In *Proc. 1997 Security Protocols Workshop*, pages 79–90, 1997.

[McC88] D. McCullough. Noninterference and the composability of security properties. In *Proc. IEEE Symposium on Security and Privacy*, pages 177–187, 1988.

[SR96] V. Shoup and A. Rubin. Session key distribution using smart cards. In *Advances in Cryptology – Eurocrypt 96 Proceedings*, 1996.

[TH99] W. Tzeng and C. Hu. Inter-protocol interleaving attacks on some authentication and key distribution protocols. *Information Processing Letters*, 69(6):297–302, March 1999.

[Yao82] A. Yao. Protocols for secure computation. In *Proc. Twenty Third Annual Symposium on the Foundations of Computer Science*, 1982.

Provably Insecure Mutual Authentication Protocols: The Two-party Symmetric Encryption Case

Jim Alves-Foss
Center for Secure and Dependable Software
University of Idaho
http://www.csds.uidaho.edu

A "Secure" Protocol

$$\xrightarrow{R_A}$$

$$A \xleftarrow{\{B,A,R_B,R_A\}_a} B$$

$$\xrightarrow{\{A,R_B\}_a}$$

"Protocol 1 - MAP1"

From: Bellare and Rogaway "Entity Authentication and Key distribution". In *Advances in Cryptology -- Crypto 93 Proc.*, pp 232-249, 1993

Simple Tailoring of a Protocol

$$\xrightarrow{R_A}$$

$$A \xleftarrow{\{A,B,R_B,R_A\}_a} B$$

$$\xrightarrow{\{A,R_B\}_a}$$

"Protocol 2 - Eve 1"

Modified version of MAP1 by swapping first two fields of message 1.

Attack Against A with B offline

$$A \xrightarrow{R_A} E_B$$

$$E_B \xrightarrow{R_A} A$$

$$E_B \xleftarrow{\{B,A,R_{A'},R_{A'}\}_a} A$$

$$A \xleftarrow{\{B,A,R_{A'},R_{A'}\}_a} E_B$$

$$A \xrightarrow{\{A,R_{A'}\}_a} E_B$$

Using Map1 *Using Eve1*

What are Multiprotocol Attacks?

- Multiprotocol Attack
  - Interleaves messages from two separate protocols to attack one of them.
  - The attacked protocol is subverted using either:
    - An incidental collision with another protocol.
    - A deliberately *tailored* protocol.
  - An attacker may successfully masquerade as client A to server B using protocol P, even if A does not support P.

Which Protocols are Susceptible

- Attacks in this talk are specific to shared-key protocols.
  - If a key is shared between two protocols.
  - If the protocols follow the format specified in this talk.
    - Two parties, shared-key encryption, no nested encryption, no other shared secrets
  - May not work against all shared-key protocols.

Protocol Specifications

- BR Protocols are defined as efficiently computable functions Π on the following inputs:
  - 1^k, $i \in I$, $j \in I$, $a \in \{0,1\}^*$, $conv \in \{0,1\}^*$, $r \in \{0,1\}^w$
  - i and j identify the participants, a is i's secret information, $conv$ is the conversation so far and r denote random coin flips
- Where $\Pi(1^k, i, j, a, conv, r)$ returns (m, δ, σ)
  - $m \in \{0,1\}^* \cup \{*\}$, $\delta \in \{A,R,*\}$, $\sigma \in \{0,1\}^* \cup \{*\}$
  - m is the message sent to j, δ is A(accept), R(reject) or don't know(*) status of i regarding the authentication, and σ is private output.

"Proofs of Security"

- Secure protocols are pitted against an adversary who has at her disposal a collection or *oracles*, $\Pi_{i,j}^{s}$ (denoting session s between i and j)
- The adversary is in control of all communication and can transfer messages between active *players* and the oracles.
- A *secure* protocol is one that is secure against all such oracle attacks with probability equivalent to the security of the underlying cryptographic algor.

Our "Secure Protocols"

- We modify the BR notation of oracles to be $\Pi(n)_{i,j}^{s}$ where n denotes which different protocol is being executed.
- An adversary, in this approach, can relay messages between oracles of different protocols.

Proofs of Protocol Classes

- We present a notion of meta-proof where we prove that no protocol in a class of protocols is secure.
  - Messages consist of plaintext fields, and composition of fields.
  - Any field or message may be encrypted with the players shared key.
  - No nested encryption is permitted.
  - No other shared secrets exist between the players

The proof

- Consider when a player, *A*, will accept the authentication of player *B* in a protocol.
  - This acceptance occurs when *A* receives a message containing information that only *B* could have sent. The information is either
    - Case 1 - encrypted with the shared key or
    - Case 2 - sent in plaintext.

The cases

- Case 1 - encrypted message
  - Case 1.1 - the authentication information was sent in a previous plaintext message.
    - The adversary takes the plaintext information and feeds it to a tailored protocol that encrypts the information into a message of the correct format.
  - Case 1.2 - the authentication information was sent in a previously encrypted message.
    - The adversary feeds the previous message to a tailored protocol that takes either takes the information and encrypts it into a message of the correct format, or extracts the information and sends it directly to the adversary

The cases

- Case 2 - plaintext message
  - Only case - the authentication information was sent in a previously encrypted message.
    - The adversary feeds the previous message to a tailored protocol that takes either takes the information and encrypts it into a message of the correct format, or extracts the information and sends it directly to the adversary who then feeds it to another tailored protocol to create the appropriate message.

QED.

Protection Against Tailored Protocol Attacks

- Why do the attacks occur?
  1. Keys (even certified keys) may be shared between multiple protocols.
  2. Tailored (or chosen) protocol is installed on a victim's machine.

Protection Against Tailored Protocol Attacks

- How do we stop the attacks?
- Kelsey, et. al:
  - Limit the scope of the key
  - Uniquely identify each application, protocol, version and protocol step
  - All protocols should have a fixed unique identifier in a fixed position in the message
  - Tie the unique identifier to encryption
  - Include support in smartcards

Protection Against Tailored Protocol Attacks

- Do these work?
- For smartcards they may, but not for general computers.
  - Requirements that insist on a unique identifier assumes that protocols follow the rules, a *tailored* protocol need not follow the rules.
  - Without these identifiers, we can not limit key usage to a particular protocol.

Solution

- What is the solution?
  - We must limit key usage to protected/trusted subsystems.
  - The subsystems must only allow encryption by certified applications, (those that follow the rules).
  - Operating system security must be in place to protect subsystems and stored keys.

Challenges

- Develop specific guidelines for protocol message content identifiers
- Enforce guidelines, limitations, and trust model in key management and crypto packages for *protocols*
- Establish protocol certification authority
- Prevent user apps from accessing keys

Suggested Protocol Architecture 1

- Develop a protocol message specification language.
  - The protocol developer obtains certification of protocol message set, and releases to application developers
  - Protocol application submits certification to crypto library to establish protocol
  - Subsequent calls to crypto library specify protocol and message identifiers; crypto library performs operation ONLY if message format matches specification

Suggested Protocol Architecture 2

- Develop a protocol message specification language.
  - The protocol application submits full protocol specification to lower level (protocol API - PAPI)
  - All messages are created through calls to PAPI, which will include unique identification information in each message. PAPI accesses the crypto API.
  - No user application can directly access crypto library, MUST go through PAPI.