Finite-State Security Analysis of OTR Version 2

Joseph Bonneau
Stanford University
Stanford, CA
jbonneau@stanford.edu

Andrew Morrison
Stanford University
Stanford, CA
asm@cs.stanford.edu

ABSTRACT

Off-the-Record messaging is a protocol for enabling secure, authenticated, deniable messaging with perfect forward secrecy, specifically over instant messaging networks. In this paper we describe the results of a finite-state security analysis of the OTR protocol. In addition to finding several security issues in the process of modeling the protocol, our model has discovered security problems in both the authenticated key exchange and data exchange phases of the protocol. The security problem during data exchange leads to an attack where by an active attacker can modify a message without detection by either party or disruption of the protocol. In addition to describing the attacks found, we describe possible solutions where appropriate.

1. INTRODUCTION

The OTR protocol [2] aims to provide a digital communication mechanism resembling a casual private conversation in which the honest principals involved may be assured *secrecy*, *authentication* and more interestingly *perfect forward secrecy* and *deniability*. A full description of the current protocol, version 2.3, is provided in [1]. A brief outline of the protocol is provided for reference.

Figure 1 shows the initial authenticated key exchange, modeled after SIGMA. Authentication is provided by a public key infrastructure and signatures. The initial AKE authenticates each principal and establishes a Diffie-Hellman shared secret between the two. Signatures are used only in the AKE phase, which allows an initial authentication and allows deniability in the data exchange phase. The use of signatures means that neither party can deny that a conversation did occur between them, it is the contents of the conversation which are intended to be deniable.

Illustrated in Figure 2 is the OTR data exchange protocol. Each message is sent encrypted using AES in counter mode, turning AES into a stream cipher which makes messages

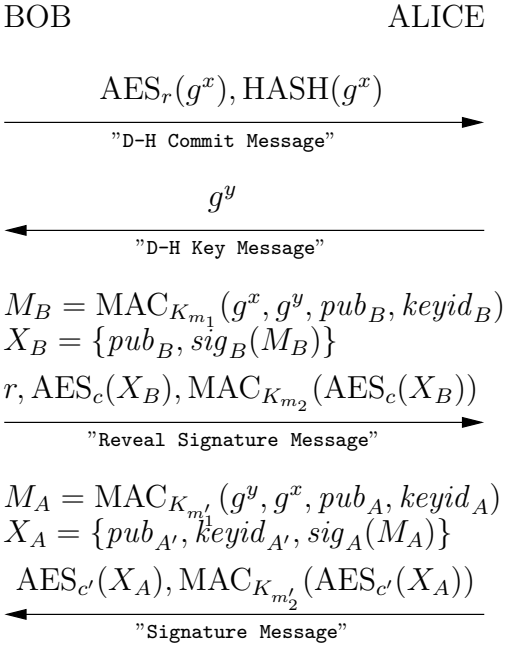

Figure 1: OTR authenticated key exchange protocol

malleable to a third party. Malleability is desirable so that it may be claimed that any third party in possession of the proper MAC keys is capable of altering messages.

Two principals communicating via OTR are constantly attempting to re-key, both to ensure forward secrecy and to enable deniability. At any given time, the two are using key material derived from a Diffie-Hellman shared secret $g^{x_i y_i}$. Each data exchange message sent by Alice contains a new proposed Diffie-Hellman value $g^{x_{i+1}}$. Once Bob has seen this value, he will use $g^{x_{i+1} y_i}$ to encrypt future messages, as well as sending a new exponent $g^{x_{i+1}}$ of his own. This re-keying system has two useful properties.

First, the principals can securely erase old Diffie-Hellman exponents as soon as they will no longer be used. This ensures perfect forward secrecy, as a future break-in to either principal's machine cannot recover the old keys, and therefore cannot decrypt stored messages sent with old keys. Constantly re-keying makes the "window of vulnerability" small, that is, a break-in will only allow an attacker to read

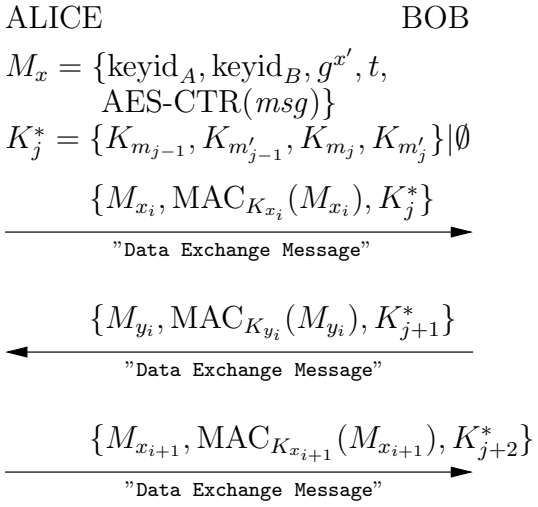

Figure 2: OTR data exchange protocol

the last few messages encrypted with the current keys instead of the whole conversation. Additionally, re-keying allows the publication of the MAC keys associated with expired Diffie-Hellman shared secrets. Because the MAC keys are derived from a hash of the encryption keys, they can be published without allowing third parties to decrypt old messages. Once old MAC keys have been published, any observer can modify the messages and re-MAC them. So, either Alice or Bob can deny sending a given message, claiming the contents were modified by a third party.

2. SECURITY PROPERTIES

In order to model and verify the OTR protocols, we first define a set of desired security properties. In the following sections we provide details of several security invariants which must hold at all phases of execution of the OTR protocols.

2.1 Secrecy

Secrecy means that no third party should be able to read the messages Alice and Bob are sending. A secrecy invariant is defined such that for a conversation between principals A and B, no other agent shall possess the pair $\{\text{AES}_K(m), K\}$ for any data message m sent using the OTR protocol.

2.2 Authentication

Authentication pertains to the initial AKE, and means that if the protocol completes, both principals are assured they are talking to the principal they believe they are speaking with. We define it as an invariant such that if a principal A believes the AKE protocol has completed successfully with principal B, it must be the case that B believes the protocol has completed successfully with A.

2.3 Perfect Forward Secrecy

Perfect forward secrecy shall be defined such that for a conversation between principals A and B, no other agent who gains possession of the tuple $\{\text{AES}_{K_t}(m_t), K_t\}$ at time t shall be able to to learn any information about the same tuple for time $t' < t - \epsilon$. This means that even if a malicious adversary Mallory gains possession of current key material being used during the protocol, for example by cracking into one of the principal's machines, she cannot read any messages sent with keys too far from before the current time.

2.4 Integrity

During the data exchange protocol, integrity means that no messages are altered in transit without detection. For any message $\{\text{AES}_K(m), K\}$ accepted by a principal B from A during the OTR protocol, is must be the case that A actually sent the message m to B.

2.5 Plausible Deniability

We define *plausible deniability* via two categories; *weak deniability* in which it may be proven that both A and B have all necessary key material to produce any given message and *strong deniability* in which it may be proven that principals other than A and B are capable of producing valid messages.

2.5.1 Weak Deniability

During data exchange, the ability to transmit a valid message requires knowledge of the current Diffie-Hellman shared secret $g^{x_i y_j}$. Weak deniability shall be defined as the property that both A and B possess the full set of necessary key material derived from this shared secret for any message at the time it is sent. Given two parties with the necessary key material, it cannot be proven that either one was the legitimate author. This property is inherently true of any symmetric-key cryptographic system, which are exclusively used in OTR after the AKE, which uses signatures.

2.5.2 Strong Deniability

We define strong deniability as the ability to claim that not only could A and B have created a given message, but anyone could have modified the message. This property may be modeled by the ability of an outside agent F (the forger, "Francis") to forge transcripts of a conversation based on what is heard on the network. In more concrete terms, every the full set of correct MAC keys used in the conversation must be available to anybody listening at any point on the network by the end of a conversation.

3. FOUND ATTACKS

The following set of attacks were discovered either via "pencil and paper" methods or by use of the Murφ finite-state model checker [3]. Through our analysis, flaws were found in version negotiations, the strong deniability property, message integrity during data exchange and in authentication during authenticated key exchange. No flaws were discovered pertaining to the secrecy, forward secrecy or weak deniability properties.

In all of our attacks, we assume the Dolev-Yao model, where a malicious attacker, Mallory, has complete control over the network. Mallory can send,block, re-send, or modify any message on the network, although she is limited to assuming the cryptographic primitives used are "perfect" and unbreakable. We use "Alice" and "Bob" in our descriptions to represtent two principals communicating via OTR, as well

as "Francis," a third party who attempts to forge conversations by passively listening to Alice and Bob's conversation on the network.

3.1 Version Rollback

Before authentication or communication occurs, protocol version negotiations take place in the clear over a presumably insecure channel. A version rollback attack exists if the attacker changes the set of protocols each principal is willing to use. For example, if both principals intend to state support OTR version 1 or 2, the attacker can change each to state support for version 1 only. Each principal will assume the other only supports version 1 and communicate with that version.

The fix for this attack is for both parties to state at the end of the AKE phase what their initial version preferences were. If the stated preferences do not match what was actually used, it is known that the preferences were tampered with, and the AKE will fail. However, this fix cannot be applied to previous versions. If Alice and Bob are tricked into using version 1, that version will not allow them to state and check each other's version preferences securely! Thus the only true fix for this problem is to deprecate versions 1 and 2 of the protocol, and ensure in all future versions that protocol preferences are securely verified before completion of the AKE.

3.2 Attack on Strong Deniability

Strong deniability, the claim that a third party listening on the network will learn the complete set of MAC keys and therfore be able to forge a different transcript of the conversation which is still cryptographically valid does not hold.

Since no validy check is performed on the expired and published key material, Mallory can replace the MAC keys published by Bob with a set of random values based on some $f(g^r)$ as opposed to the correct key material, disallowing Francis from producing valid transcripts.

The fix for this attack is non-trivial. In addition to Mallory modifying or replacing the published MAC keys, upon which no integrity checks are performed, if either principal is dishonest they themselves can also neglect to publish the correct values. The first attempt at fixing the problem is for both parties to check the expird MAC keys published by the other party.

However, if two intruders have network control on both ends of the channel, they could alter all expired mac values over the network, and un-alter them so that the principal receiving the message will think they were published properly as seen in Figure 3. For example, if the two intruders have a shared secret, they could encrypt the published key material while being sent over the network, then decrypt prior to message delivery. Alice and Bob will think MAC keys are being published properly, but Francis will only see garbage MAC keys from the middle of the channel.

This attack is fundamental and a defense is not clear. The strong deniability property depends on Alice and Bob being able to broadcast their old MAC keys to every party on

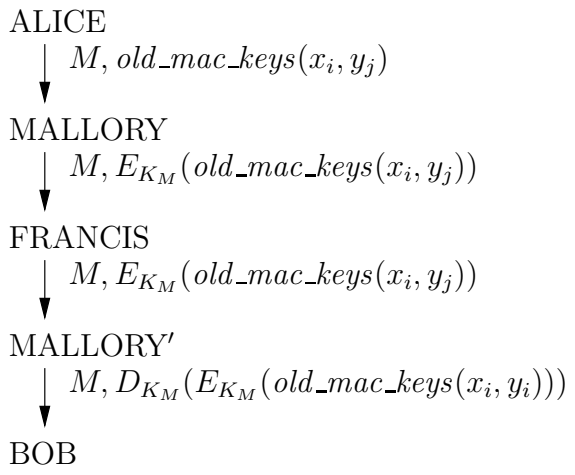

Figure 3: Weak Attack on strong deniability

the network. However, if an adversary has network control, they can always prevent anybody besides Alice and Bob from seeing the correct MAC keys. It is impossible for Alice and Bob to guarantee delivery of their expired MAC keys to every third party on an insecure network.

The only approach to broadcasting the keys given an insecure network is to securely send them to some trusted third party with whom an encrypted channel can be constructed. This trusted third party would have to act as a server which could enable others to access the keys. This does not seem like a realistic solution for the OTR protocol.

In practice, this attack is probably unlikely due to the level of network control which is required. However, it does weaken the claimed property of deniability, since it is never fully clear that Francis will be able to forge a conversation. In any case, it should be required by the protocol that both principals perform sanity checks on published MAC keys.

3.3 Authentication Failure

During the initial AKE phase of the protocol, a simple man-in-the-middle attack is able to convince Alice to commit to a key exchange without properly authenticating Bob. Figure 4 outlines the attack. Bob believes he is talking to Mallory,

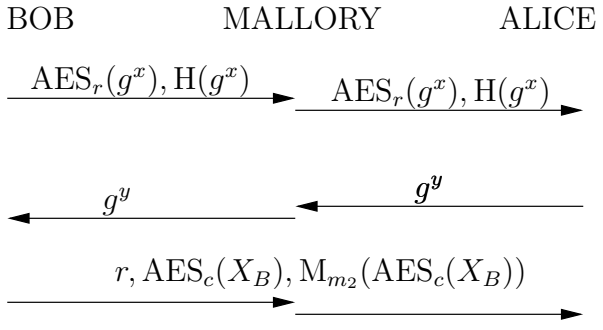

Figure 4: Attack on AKE authentication

who is forwarding Bob's messages to the victim, Alice, who

believes she is talking to Bob. The first three messages of the protocol are passed without modification, and the authentication invariant fails when Alice accepts the "Reveal Signature" message, and believes she has successfully finished an authenticated key exchange with Bob.

Mallory will be unable to complete the AKE with Bob, since she has not learned either Bob's x value or Alice's y value, and thus cannot encrypt them and sign them. If she forwards Alice's "Signature" message to Bob, Bob will know it was not signed with Mallory's public key. So, Bob will realize the AKE has failed with Mallory and not send or accept any messages.

Since Mallory has not learned either principal's Diffie-Hellman exponent, she will not be able to forge messages for Alice and thus Alice will not receive any messages in her newly opened OTR conversation. Alice may attempt to send messages for Bob, but Mallory will not be able to read them.

Thus the actual damage done by this attack is minimal, but nonetheless it is an authentication failure in that Alice believes she has had a successful AKE run with Bob. Bob not only does not think he has had a successful AKE with Alice, he does not know Alice received his AKE messages intended for Mallory, and may not even know who Alice is. It is possible Mallory could carry out the attack with many different Bob's, leaving Alice with many open conversations, possibly leading to a denial of service attack on Alice if she has large state associated with each open conversation. In practice, it is unlikely this could occur. It is assumed Alice is an instant messenger client, and would personally approve all OTR conversations. This attack may be nothing more than an anomaly in the protocol.

The fix for this flaw is to add information about who you believe you are communicating with, both encrypted and MAC'd in the final two AKE messages. This way, Alice could see that Bob believes he is speaking with Mallory, and know not to accept the results of the AKE.

3.4 Message Integrity

The message integrity property can be broken through the attack presented in Figure 5. After two normal messages,

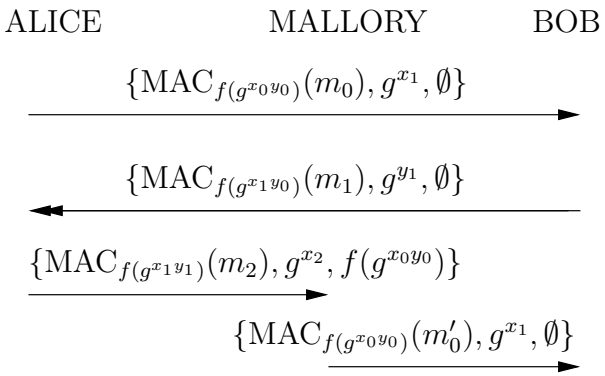

Figure 5: Attack on message integrity

Alice will publish the expired MAC keys associated with (x_0, y_0). Because Alice has already seen a message from Bob encrypted using x_1, y_0, she correctly assumes no more messages will be sent by Bob with these keys, since messages are assumed to arrive in order. However, Mallory, who is assumed to have total network control, can block this outgoing message from Alice. She can then re-send Alice's original message m_0. Since Mallory has learned the MAC keys used for this message, however, she can modify the message contents to m_0', since a malleable encryption scheme is used. When Bob receives this message, it appears it was sent legitimately by Alice before receiving his message m_1, and was simply delayed in the network, so Bob accepts. Mallory could then send Alice's message m_2, and the conversation between Alice and Bob will proceed, although a spurious message m_0' has been inserted.

The attack could also take the form of Figure 6, where Alice sends two messages out before getting Bob's response. Mallory could then block the second message, wait until Alice publishes the MAC keys associated with it, and then send it.

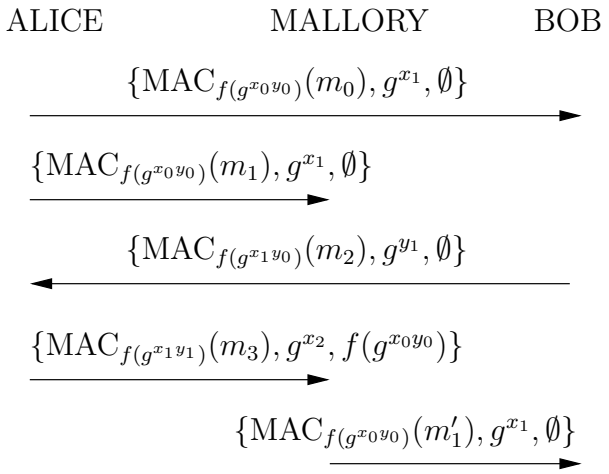

Figure 6: Attack variant on message integrity

The fix for this attack is not entirely clear. According to the protocol authors, the currently deployed implementation is not susceptible to this attack, because each party only publishes the MAC keys they used to receive messages. By publishing only MAC keys used to receive, previously sent messages cannot be modified.

However, this fix weakens the strong deniability property, since it means that each principal only publishes half of the MAC keys used, instead of each principal redundantly publishing the full set, as the protocol specification calls for. This means that if either party is dishonest they can publish their MAC keys incorrectly, and disable the strong deniability property.

There is a negative interaction occuring between the desire for deniability and message integrity. We believe a better fix might be to publish MAC keys which are two generations removed from use, thus ensuring the other principal has already seen a message at least one generation newer

than the published keys. The downside to this approach is that it requires remembering keys for longer, increasing the window of vulnerability for deniability and perfect forward secrecy.

In any case, as specified the protocol has a serious flaw which leads to a strong attack as Mallory can insert a modified message into the conversation without detection. The protocol must be updated to eliminate this risk.

4. CONCLUSION

Via finite-state analysis of the OTR protocol using the Murφ model checker, several attacks have been found and improvements to the protocols have been offered where possible. A refactoring of the AKE protocol is given in Figure 7. In

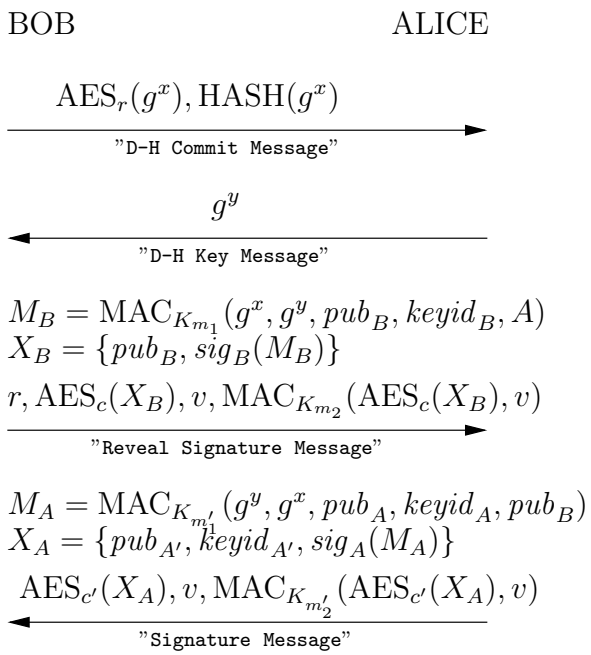

Figure 7: Improved OTR AKE protocol

Figure 7 we add a reference A to the `Reveal Signature Message` specifying that Bob believes he is talking to Alice, and a reference pub_B to the `Signature Message` signifying that Alice believes she is talking to Bob. We add a new element v to the `Reveal Signature Message` and `Signature Message` which is meant to specify the version each principal initially desired such that a version rollback attack may be more easily detected.

We also present a re-factoring of the data-exchange protocol as provided in Figure 8, with the requirement that expired MAC keys are published only when two generations old, eliminating the message integrity attack. Furthermore, an improved protocol description should make explicit that published MAC keys shall be checked for validity on both ends, and either re-published or terminating the conversation upon failure.

Overall, we conclude from our analysis that OTR is generally secure and does provide the properties it claims other

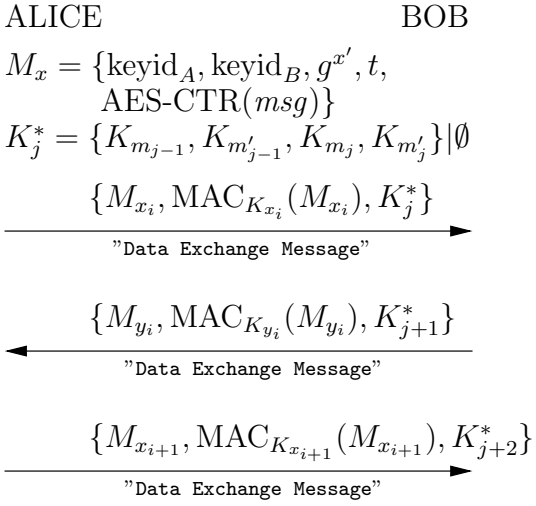

Figure 8: Improved OTR data protocol

than integrity. However, our finite-state analysis is designed only to discover attacks, not to prove the protocol correct. We recommend the OTR protocol incorporate our suggested fixes into a new version 3 to defend against the found attacks, but cannot rule out the possibility of additional attacks on the protocol.

5. REFERENCES

[1] *Off-the-Record Messaging Protocol version 2*

[2] *Off-the-Record Messaging, or, Why Not To Use PGP*, Nikita Borisov, Eric Brewer, and Ian Goldberg, WPES 2004, http://www.cypherpunks.ca/otr/Protocol-v2-3.0.0.html

[3] "Automated analysis of cryptographic protocols using Murphi," John C. Mitchell, Mark Mitchell, and Ulrich Stern, IEEE Symposium on Security and Privacy, 1997