Brief yet Scintillating Article about WEP and WPA2

Ashley Oudenne
The University of Texas at Austin
aoudenne@cs.utexas.edu

Jerremy Adams
The University of Texas at Austin
ja7872@cs.utexas.edu

December 12, 2012

Abstract

Interesting yet vague things about our paper.

1 Introduction

2 Related Work

3 Modeling of Protocols

3.1 Protocol Descriptions

3.1.1 WEP

WEP, described in [4], uses a one-message protocol to transmit data between two parties that relies on a secret key k that has been shared between them. Before a message is sent, an integrity checksum *C(m)* is computed on the message so that the recipient can verify that the message has not been altered in transit. The message is concatenated to this checksum to form the plaintext P.

The plaintext is now encrypted using the RC4 cipher. The sender chooses an initialization vector IV and uses this vector along with the secret key k to generate a keystream, which is a long sequence of pseudorandom bits. The sender then uses exclusive-or (or XOR, denoted by $\oplus$) to XOR P and the keystream to generate ciphertext C. The complete encryption process can be represented by:

$$C = P \oplus RC4(IV, k)$$

The message is then ready to be transmitted from the sender to the receiver. We will represent this transmission symbolically as:

$$A \rightarrow B : IV, (P \oplus RC4(IV, k)), where P = \langle m, C(m) \rangle$$

Notice that the initialization vector IV is sent in the clear, meaning that anyone can read the value of the initialization vector. This should not matter because an attacker would need boh the IV and the secret key k to recover the keystream used to decrypt the message, but we will show later in this paper that this is enough to break WEP.

Decryption works by reversing the encryption process described above. The recipient first regenerates the keystream used to encode the message with the secret key k and the IV sent along with the encrypted message. He can XOR the ciphertext with the keystream to recover the plain-

text P:

$$\begin{aligned} P &= C \oplus RC4(IV, k) \\ &= (P \oplus RC4(IV, k)) \oplus RC4(IV, k) \\ &= P \end{aligned}$$

The recipient can then verify the integrity of the message by splitting P into $\langle m, C(m) \rangle$ and recomputing the checksum of m. If the computed checksum matches the sent checksum, then the message has not been tampered with.

3.1.2 WPA/WPA2 Four-Way Handshake

The Four-Way Handshake protocol is used in both WPA and WPA2 to authenticate a station to an access point, to compute a pairwise transient key (PTK) to be used in future communication between these parties, and to distribute a group transient key (GTK) to be used by the station to communicate with other devices connected to the access point [6].

TO-DO:Finsih this part

3.2 ProVerif

To formally prove the correctness of WEP and the Four-Way Handshake, we use the ProVerif Cryptographic Verifier created by Bruno Blanchet. ProVerif uses prolog rules to encode the protocol and abstracts away fresh values and the number of steps in the protocol [2]. Instead, Proverif treats each fresh value as a function of other messages in the protocol, meaning that different values are used for each pair of protocol participants. Each step in the protocol can be completed any number of times, and past steps can be re-executed arbitrarily. This permits ProVerif to execute an unlimited number of runs of a protocol. A protocol is proved to be secure by querying ProVerif whether an attacker knows a key or the content of some encrypted message. Since we assume that messages are placed on a channel c, and that the attacker has access to all messages on c, security is proved if the attacker cannot learn the security property despite access to all the messages. This verifier has been used to successfully prove the insecurity of protocols such as the Diffe-Hellman key exchange protocol, Initial Key Agreement, and the Needham-Schroeder symmetric-key protocol [5, 1].

3.2.1 Horn Clauses

ProVerif can take protocols in either horn clauses or pi-calculus. In our protocol implementations, we chose to use typed horn clauses. An untyped horn clause is a disjunction of literals with at most one positive literal (ex. $\neg p \vee \neg q \vee t$) [3]. They can be written as implications (ex. $(p \wedge q) \rightarrow t$), as they are in Prolog, on which ProVerif is based. Typed horn clauses merely allow the user to add a type system to the program for clarity, but the underlying logic is the same as that of untyped horn clauses.

3.2.2 Attacker Model

In ProVerif, it is not necessary to explicitly model the attacker. Rather, the user constructs a list of clauses detailing what anyone, including the attacker, can do with messages that are put on the channel c. Such abilities include separating a message $c(a, b)$ into its separate parts and placing those parts on the channel ($(c(a), c(b))$, decrypting messages if the encryption key of the message is known, encrypting messages using a known key, XOR-ing messages, and computing checksums of messages. We use the Dolev-

Yao attacker model, in which the attacker can intercept, overhear, and create new messages, since this corresponds closely to the capabilities a WEP or Four-Way Handshake protocol attacker would have.

3.3 Modeling Security Properties

4 Analysis of Protocol Models

4.1 WEP

4.2 WPA/WPA2 Four-Way Handshake

5 Conclusions and Future Work

References

[1] Martín Abadi. Security protocols: Principles and calculi tutorial notes.

[2] Bruno Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *14th IEEE Computer Security Foundations Workshop (CSFW-14)*, pages 82–96. IEEE Computer Society, 2001.

[3] Bruno Blanchet. Using Horn clauses for analyzing security protocols. In Véronique Cortier and Steve Kremer, editors, *Formal Models and Techniques for Analyzing Security Protocols*, volume 5 of *Cryptology and Information Security Series*, pages 86–111. IOS Press, 2011.

[4] Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting mobile communications: The insecurity of 802.11. In *MobiCom '01: Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pages 180–189, 2001.

[5] Pascal Lafourcade, Vanessa Terrade, and Sylvain Vigier. Comparison of cryptographic verification tools dealing with algebraic properties. In *FAST '09: Proceedings of the 6th International Conference on Formal Aspects in Security and Trust*, 2010.

[6] Jong Liu, Jun Zhang, and Jun Li. Security Verification of 802.11i 4-Way Handshake Protocol. In *ICC '08: IEEE International Conference on Communications, 2008*, pages 1642–1647, 2008.