Lecture 13: Certificates, Digital Signatures, and the Diffie-Hellman Key Exchange Algorithm

Lecture Notes on "Computer and Network Security"

by Avi Kak (kak@purdue.edu)

March 1, 2017

12:01 Noon

Goals:

- Authenticating users and their public keys with certificates signed by Certificate Authorities (CA)
- Exchanging session keys with public-key cryptography
- X.509 certificates
- **Perl and Python code for harvesting RSA moduli from X.509 certificates**
- The Diffie-Hellman algorithm for exchanging session keys
- The ElGamal digital signature algorithm
- **Can the certificates issued by CAs be forged?**

CONTENTS

13.1: USING PUBLIC KEYS TO EXCHANGE SECRET SESSION KEYS

- From the presentation on RSA cryptography in Lecture 12, you saw that public key cryptography, at least when using the RSA algorithm, is not suitable for the encryption of the actual message content.

- However, public key cryptography fulfills an extremely important role in the overall design and operation of secure computer networks because it leads to superior protocols for managing and distributing secret session keys that can subsequently be used for the encryption of actual message content using symmetric-key algorithms such as AES, 3DES, RC4, etc. [although, not RC4 as much any longer].

- How exactly public key cryptography should be used for exchanging the secret session keys depends on the application context for secure communications and the risk factors associated with the breakdown of security.

- If a party A simply wants to receive all communications confidentially (meaning that A does not want anyone to snoop on the incoming message traffic) and that A is not worried about the authenticity of the messages received, all that A has to do is to publish his/her public key in some publicly accessible place (such as on a web page). Subsequently, anyone wanting to send a confidential message to A would encrypt that message with A's public key. Only A would be able to decrypt such messages.

- If two parties A and B are sure about each other's identity, can be certain that a third party will not masquerade as either A or B vis-a-vis the other, they can use a simple and direct key exchange protocol for exchanging a secret session key. In general, such protocols will not require support from any coordinating or certificating agencies. A direct key exchange protocol is presented in Section 13.2.

- The key exchange protocols are more complex for security that provides a higher level of either one-sided or mutual authentication between two communicating parties. **These protocols usually involve Certificate Authorities**, as discussed in Section 13.3.

13.2: A DIRECT KEY EXCHANGE PROTOCOL

- If each of the two parties A and B has full confidence that a message received from the other party is indeed authentic (in the sense that the sending party is who he/she/it claims to be), the exchange of the secret session key for a symmetric-key based secure communication link can be carried out with a simple protocol such as the one described below:

  – Wishing to communicate with B, A generates a public/private key pair $\{PU_A,\ PR_A\}$ and transmits **an unencrypted message** to B consisting of PU_A and A's identifier, ID_A (which can be A's IP address). Note that PU_A is party A's public key and PR_A the private key.

  – Upon receiving the message from A, B generates and stores a secret session key K_S. Next, B responds to A with the secret session key K_S. This response to A is **encrypted** with A's public key PU_A. We can express this message from B to A as $E(PU_A,\ K_S)$. Obviously, since only A has access to the private key PR_A, only A can decrypt the message containing the session key.