Chapter 4

Performance Analysis

There are two parts of performance analysis. First, we evaluate our Round-Robin Key Management (RRKM) performance with IEEE 802.11 standard. This case will show you our security enhancement with approximate zero penalty and seamless data transfer. Second, we compare our key management performance with the most popular authentication type, EAP-TLS, of IEEE 802.1x standard. Although both RRKM and EAP-TLS is running on different layer that MAC and Application respectively, they supply the same functionality. Our evaluation indicates the RRKM is superior to EAP-TLS on the performance and total deployment cost.

In this chapter, we will first count the run-time of used cryptographic algorithms. We then compare the IEEE 802.11 and EAP-TLS with our proposed RRKM.

4.1 The Cost Of Cryptographic Algorithm

In order to compare each key management performance, we first evaluate the amount of running time of cryptographic algorithm. The evaluation environment uses a laptop machine, which runs on Mandrake Linux 9.2 with Pentium 4 Mobile 2GHz processor and 256MByte SDRAM. The evaluating program is based on the OpenSSL-0.9.7b [24] package, an open-source project dedicates to data security, with our purposeful modification. Our encrypted data is a visible string with 114-byte long. In the RSA algorithm, it uses a 1024-bit private-public key pair. The plain text is encrypted using public key and its final output, cipher text, is decrypted using associated private key respectively. In the RC4, it uses a 64-bit symmetric key, which encrypts a 114-byte plain text and decrypts the cipher text in reverse. However, the SHA-1 is not an encryption algorithm in which we give a 114-byte input and get a 20-byte output.

Table 4-1. The run-time comparison between RSA, RC4, and SHA-1

	Action	10,000 times	Cost Ratio
RSA	Encrypt (Public)	8.46 sec	423
	Decrypt (Private)	159.95 sec	7997.5
RC4	Encrypt	0.02 sec	1
	Decrypt	0.02 sec	1
SHA-1	Digest	0.025 sec	1.25

4.2 IEEE 802.11 and RRKM

4.2.1 Mutual Authentication

The mutual authentication is that the peer side has to trust on each other, AP authenticates station and station authenticates AP. This is especially important in the wireless network in order to make sure you are connecting to the right network. Unfortunately, the IEEE 802.11 standard provides one-way authentication, the station is only authenticated to the AP. In our authentication, the mobile station can validate the AP's encrypted challenge text with its individual pre-shared key.

From table 4-2, we can find that the RRKM takes very limited computation, because the AP and stations require an additional WEP and SHA-1 algorithm. In table 4-1, we have known that it takes 0.02 seconds for 10,000 times RC4 (WEP) encryption. Therefore, we can almost ignore the extra effort. Furthermore, the authentication only happens whenever the station is trying to connect to the wireless network.

Table 4-2. IEEE 802.11, RRKM, and EAP-TLS running cost

	IEEE 802.11		**RRKM**		**EAP-TLS**	
	AP	**STA**	**AP**	**STA**	**AP**	**STA**
Authent i-cation	**Verify Chal:** WEP *x 1*	**Encrypt Chal:** WEP *x 1*	**Verify Chal:** WEP *x 1* **Chal Text:** WEP *x 1* **SIG:** SHA1 *x1*	**Encrypt Chal:** WEP *x 1* **Decrypt Chal:** WEP *x 1* **SIG:** SHA1 *x1*	**Verify CA:** RSA *x 1* HMAC *x1* **Decrypt:** RSA *x 1*	**Verify CA:** RSA *x 1* HMAC *x1* **Encrypt:** RSA *x 1*
Re-key	No support	No support	Hash *x 1*	Hash *x 1*	HMAC *x 1* RC4 *x 1*	HMAC *x 1* RC4 *x 1*

4.2.2 Key Management

As mentioned before, the IEEE 802.11 is lacking of key management. Our proposal, RRKM, offers the key distribution mechanism and key periodically updated. This gives a great improvement in IEEE 802.11 wireless security, because it patches the weakness of long-term shared key - WEP. User can set the re-key period to prevent the WEP key broken.

The only extra effort in our key management is a hash function routine in the both of sides, AP and stations. In fact, this function usually only requires very light-weigh resources, CPU computation, storage, and memory.

We specially simulate the running time of simple hash function in an embedded device, which has MIPS CPU 125MHz and 16MByte SDRAM with Linux-2.4 kernel. The result shows that a simple hash function just takes 0.055 seconds for 10,000 times key generations.

Assume we have 100 stations and the re-key period is 60 seconds, then there will be 144,000 times re-key per day. According to the simulation result, the extra running time of RRKM even doesn't take 1 second within all day long. Therefore, we consider the re-key management takes no extra work.

4.3 EAP-TLS and RRKM

The EAP-TLS of IEEE 802.1x standard and RRKM performs the same roles, authentication and re-key mechanisms. The EAP-TLS is high-level, Application layer, authentication that requires authentication (RADIUS) server to validate the mobile station, but our key management is on the MAC layer and doesn't need any additional server. In order to evaluate both key managements performance under the same equipment, we suppose the authentication server is inside the Access-Point for EAP-TLS type.

The EAP-TLS uses EAPOL-Key frame to distribute the WEP group key for authenticated stations. It only takes one HMAC for signature and one WEP for encryption, thus it could be thought as very light-weigh. But, it requires RSA cryptographic algorithm, which costs high running time, in the authentication phrase. In opposition, the RRKM only uses WEP protocol and SHA-1 in authentication stage. The cost analysis is showed in table 4-2.

To analyze the running time of RSA operations, we refer [25]. That is, assume the public key (e, n) and secret key (d, n) satisfy $|e| = O(1)$ and $|d| = |n| = \beta$ then apply a public key requires $O(\beta)$ bit operations, and a secret key requires $O(\beta)$ bit operation. About the WEP encryption, it is based on RC4 and will take one iteration loop per secret key bit. Therefore, it requires $O(\alpha)$ running time, assume α is the key length. It seems still not significant to tell people how difficult it is, but the result of table 4-1 shows the heavy load in RSA cryptographic algorithm. The cost ratio could be thousands times over RC4. Our conclusion is that the bottom neck of EAP-TLS is on the RSA. The EAP-TLS could not be able to run without authentication server unless the AP and stations have powerful computation capability.