Security Issues in Digital Archiving Systems

Sang Bae Park*
*KISTI, Korea
E-mail : plucky@kisti.re.kr

1. Introduction

Since digital contents are increasing very rapidly, electronic archives are increasingly being used to store digital contents that need to be available for a long time. Digital archive is very important. In order for contents in such archive to remain useful, their integrity and authenticity must be protected over their entire life span. And sensitive contents should be protected by cryptographic encryptions. There are many issues for digital archiving systems, for example data retrieval, meta-data management, long term availability, etc. The security issues are also very important [3][4][5]. The most important thing is that data should be stored longer than valid life span of cryptographic keys. And we should consider the robustness of cryptographic primitives, block cipher, hash function, PRNG (Pseudo Random Number Generator) etc. [1] As computing power and cryptanalysis evolve, the security of a cryptographic primitive may be to insecure. After archiving systems are constructed, there is almost no way to complement the security vulnerabilities except whole new construction. These security issues should be consider before system design.

In this paper, we discuss some security issues in digital archiving systems. We mainly deal these issues keeping the long-term security in mind. In chapter 2, we discuss the long term security issues including the life span of cryptographic primitives, cryptographic keys and other issues related to system programs. In chapter, we propose a new concept ‘encryption update’ for long term confidentiality.

2. Long term security issues

In this chapter, we consider the security issues related to long term preservation.

2.1. Security of cryptographic primitives.

Cryptographic primitives are basic tools for information security. For example, block cipher, stream cipher, hash function, digital signature, PRNG (Pseudo Random Number Generator) are cryptographic primitives. These algorithms might not keep safe for a long time. The block cipher DES published in 1976, DES enjoyed widespread deployment for over 20 years. But by the late 1990s computers were so cheap and powerful that a 2^56 brute force search for the key became feasible task. Although this development was anticipated by security experts and DES was replaced by Triple DES, the confidentiality protection of prior cipher-texts was lost. So we should consider the lifetime of cryptographic primitives, block cipher, stream cipher, hash function etc. PKI (Public Key Infrastructure) environment is similar. If quantum computers are fully realized then factoring the large integer and computing discrete logarithms that provide most public key algorithms will become insecure [2].

Like block ciphers, Hash functions are also mortal. After Wang et al. published collisions for famous hash functions, MD4, MD5 and RIPEMD, these algorithms are considered to be insecure [6]. This causes the security risks of digital signatures. Collisions of hash functions nullify the authenticity of digital signature.

For long term preservation, we should consider the valid period of cryptographic primitives and make a plan for algorithm updates.

2.2. Lifetime of cryptographic keys

Usually, session keys for symmetric key cryptography are used once. But encryption keys for digital archive must hold the robustness during the preservation period. Key agreement protocol and key distribution protocol are very important. Asymmetric key cryptography has same problem. Valid period of X.509 certificate is less than one year usually. Since digital contents might preserve in many decade, there should be a plan for key update.

We also consider public security services. For example, CA (Certificate Authority) is a typical example of these services. When we determine the lifetime of digital signature, we should consider the way how to sign the data and who sign on data. Time stamp is also very important. This TTP (Trusted Third Party) service is public and all entities trust this party’s signature. A single time stamp does not provide long-term security because it relies on a cryptographic hash function, a digital signature or wide-visible media, which all are subject to security deterioration over time. We carefully determine the lifespan of cryptographic key, signature, and archiving data.

2.3. Other Issues for Long term security

We can consider other issues related to digital archiving systems. At first, long term integrity should be considered. Because most mechanisms for Integrity rely on cryptographic hash function and MAC (Message Authentication Code), these do not provide long term integrity. DRM system is also very sensitive. Most DRM systems depend on system calls provided by OS. How long do we expect the backward compatibility of OS? For long term DRM system, we do not depend on system calls. We should consider DRM system based on on-line cryptographic protocols. We also consider the flexibility when we determine the file format.

For digital archiving systems, we also consider the proof of existence. Users can check whether their data is in archiving systems without retrieving whole data. Moreover, we might need the query keyword protection.

3. Encryption Update

For long term confidentiality, there should be a process for updating encrypted data. Though decrypt-encrypt-delete old file is the easiest way to encryption update, this way has some problems. Since digital archiving is usually out-sourced, there is a great amount of communication or data leakage. So we suggest some requirements for encryption update.

- Re-encrypt encrypted data without revealing plain text
- Minimize the size of key and decryption time

For example, we might use a stream cipher for data confidentiality. After valid period, we can user other secure stream cipher in that time for re-encryption. But, this example cannot reduce the key size and decryption time. Now we are going to devise a concrete model.

Including this encryption update, we can make a procedure for long term confidentiality.

Step 0. Make a plan for valid period for cryptographic primitives and choose an algorithm
Step 1. Encrypt data and store
Step 2. Before valid period is gone, do encryption updates
Step 3. Check the update using the proof of existence

Because encryption algorithm might be insecure, we should check the deletion of prior data. For assured deletion, step 3 is very important.

4. Conclusion

In this paper, we present some security issues related to digital archiving systems. Since archiving systems preserve digital contents very long term, we should consider the key change and update. After archiving systems are constructed, there is almost no way to complement the security vulnerabilities except whole new construction. These security issues should be consider before system design. And we propose a new concept “encryption update” including security requirements for long term confidentiality. Now, we are going to devise a concrete encryption update model. We expect this can be applied in real archive systems.

5. References

[1] A. Menezes, P. van Oorschot and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.
[2] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer”, SIAM J. Comput., 26: pp.1484-1509, October 1997.
[3] J. Buchmann, A. May, and U. Volmer, “Perspectives for crpyotgraphic long-term security”, Commun. ACM, 49, pp. 50-55, September, 2006.
[4] J. Hughes and J. N. Roge, “Long-term security vulnerabilities of encrypted data”, Issues in Information Systems, 8, pp. 522-528, 2007.
[5] P. Maniatis, M. Roussopolous, T. J. Giuli, D. S. H. Rosenthal, M. Baker, “The LOCKSS peer-to-peer digital preservation system”, ACM Transanctions on Computer Systems, 23(1), pp.2-50, 2005.
[6] X. Wang and H. Yu, “How to Break MD5 and Other Hash Functions”, In Advances in Cryptology EUROCRYPT 2005, Springer-Verlag, 2005