ISSN:2231-0711
www.ijcset.net

Proactive Approach: Proactive security is based on threshold and key refreshment is performed By different cryptography function

Rashmi Singh[1],Minu Choudahry[2]
[1]*MTech (SE), RCET, Bhilai*
Rashmi.28nov@rediffmail.com

[2]*Prof. in Dept of IT, RCET, Bhilai*
minukum@gmail.com

***Abstract*- The security of public key cryptosystems relies heavily on the secrecy of the private key. Thus such cryptosystems should be augmented with methods for protecting the secret key while providing continuous availability of the system. A naive solution may be to share the private key using a proactive secret sharing scheme. This solution provides the necessary protection as long as the key is not used. However, in order to generate a signature the private key would need to be reconstructed in a single site, thus losing the advantage of distribution: A single break-in to this site will compromise the security. Instead, a proactive threshold signature scheme allows the servers to individually generate valid signatures in a special way that prevents an attacker from generating fake signatures. In particular, the scheme makes sure that the key is never reconstructed at a single site.**

Refreshment of key is based on threshold value. Each node which is participated in the network, generates random number if the random number is match on threshold then key is not refreshed otherwise it will be refreshed.

In this paper, we have developed different function for key refreshment but key refreshment is depending on threshold.

Keywords— **Proactive Security, Threshold, Key Refreshment, Cryptography Function.**

1. INTRODUCTION:

A proactive signature scheme involves three phases: the key generation phase (preferably done without a trusted dealer), the joint signature-generation phase and finally a special proactive key refreshment phase of the servers' key shares which is done periodically. The signature is generated in a distributed fashion from the shares of the key. Moreover, it has to hold that despite proactivization of the signing key, the signature on a message m, computed under any of the representations of the key is the same. The scheme withstands attackers that eventually break into all servers, as long as only a limited number of the servers are broken into between two consecutive invocations of the refreshment protocol. Proactive solutions for few signature schemes have been devised; among them is a solution to RSA signatures and to DSS (Digital Signature Standard) signatures.

2. CRYPTOGRAPHY

Cryptography offers a set of sophisticated security tools for a variety of problems, from protecting data secrecy, through authenticating information and parties, to more complex multi-party security goals. Yet, the most common attacks on cryptographic security mechanisms are `system attacks' where the cryptographic keys are directly exposed, rather than crypto analytical attacks (e.g., by analyzing cipher texts). Such `system attacks' are done by intruders (hackers, or through software trapdoors using viruses or Trojan horses), or by corrupted insiders. Unfortunately, such attacks are very common and frequently quite easy to perform, especially since many existing environments and operating systems are insecure (in particular Windows). As a result, computer and network security involve set tools to prevent and detect intrusions, and to regain control over a computer from the attacker. Detection is particularly important, since once an attack is detected on any one computer; system administrators are alarmed and are likely to regain control from the attacker - on most or all computers. Furthermore security measures are likely to be tightened, and at least some security exposures found and fixed. Therefore, attackers often do their best to avoid detection, and indeed often give up control over a computer rather than risk being detected.

2.1 DISTRIBUTED CRYPTOGRAPHY

Distributed cryptography is currently a very active research field that is related to many different areas in cryptology. There are several open problems whose solution would lead to the construction of more efficient and versatile distributed cryptosystems. Many of these problems are related to the different cryptographic protocols that are used as pieces of a distributed cryptosystem.

In general, it is not convenient that the security of a system relies on the behavior of a single agent. Let us consider, for instance, the case of a certification authority, a trusted entity that certifies that a given public key corresponds to a given user. Clearly, a certification authority that is composed by several independent servers is more reliable than one that is formed by a single server.
In this way, the security of the system is increased, because the loss or theft of several shares of the secret key does not necessarily break the system's security. Several distributed cryptosystems have been proposed until now. Most of them have a threshold structure, that is, the sets of users that are able to execute the protocol are those having a certain number of elements. Due to this fact, distributed cryptography is called also in general threshold cryptography.

3. PROBLEM WITH KEY REFRESHMENT

In the existing paper key refreshment is periodically performed by Shamir's polynomial which is available on every node. Each node generates new share and distributed to other nodes, now each node has own share and new shares which is received by other nodes, all the shares are combined and generate new share but problem is that if attacker is find out that polynomial then he can easily compute new shares or during the share distribution, attacker attack on that particular share then he can generate new shares.
We consider the example of key refreshment for three nodes. Every node generates three shares, one share is hold by node and other two are distributed. This procedure is followed by other nodes. Now each node has got three new shares, combine them and generate new sub shares.

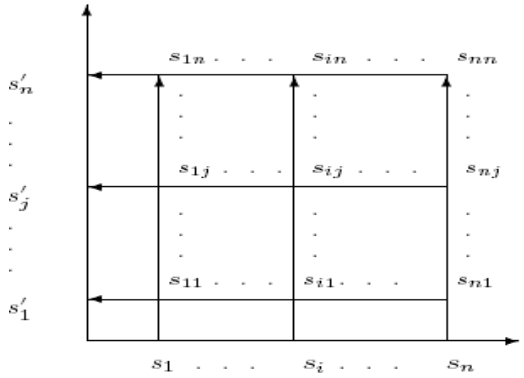

Figure 1. Share Refreshment process

Share refreshing: given an (n, t+1) sharing (s1, . . . , sn) of a private key k, with share si assigned to server i. To generate a new (n, t+1) sharing (s01, . . . , s0n) of k, each server i generates subshares si1, si2, . . . , sin, which constitute the ith column in the figure. Each subshare sij is then sent securely to server j. When server j gets all the subshares s1j , s2j , . . . , snj, which constitute the jth row, it can generate its new share s'j from these subshares and its old share sj .

4. SOLUTION APPROACH

PROACTIVE APPROACH

Proactive security provides a method for maintaining the overall security of a system, even when individual components are repeatedly broken into and controlled by an attacker. In particular it provides for automated recovery of the security of individual components, avoiding the use of expensive and inconvenient manual processes (unless perhaps when an ongoing attack is detected). The technique calls for the distribution of trust among several components (servers), together with periodic refreshments of the sensitive data held by the servers. This way, the proactive approach guarantees uninterrupted security as long as not too many servers are broken into at the same time. We describe the proactive approach and review some algorithms, implementations, and applications. We elaborate on two of the most important results: proactive signatures and proactive secure communication. Proactive signatures provide a solution for long-lived secret keys, such as the key of a certification authority. Proactive secure communication ensures secrecy and authenticity of communication, with automated refresh of the secret keys.
A common approach to enhancing the security is periodic refreshments of secrets. Examples include refreshments of passwords, and of session-key refreshment in secure communication protocols. The idea is to make `old secrets' useless for the attacker. Thus the attacker is forced to either lose control or to be constantly active, thus risking detection. Another approach to enhancing the security is the distribution of cryptographic trust among several or servers. This approach is exemplified in secret sharing algorithms and taken to a much greater extent in the notion of threshold cryptography .Here a secret key is split into shares, and each share is given to one of a group of servers. The server's engage in a protocol that `emulates' the behavior of the centralized solution. The protocol ensures security as long as at most some predefined number of servers is broken into. Threshold cryptography can indeed enhance the security against break-in attacks in many scenarios. However, it is also limited: Given sufficient amount of time, an attacker can break into the servers one by one, thus eventually compromise the security of the system. This danger is particularly eminent in systems that must remain secure for long periods of time or where secure recovery may be difficult. Proactive security is a mechanism for protecting against such long-term attacks. Proactively secure system does not wait until a break-in is detected. Instead, it invokes the refreshment protocol periodically (and proactively) in order to maintain uninterrupted security but key refreshment is performed by different function which is held by nodes.

5. WORKS ON PROACTIVE SECURITY

Ostrovsky and Yung showed how a large class of multiparty protocol problems can be solved in a proactive way, in a setting where secure communication channels are available. Their solution, based on the general paradigm of multiparty computation is of significant theoretical interest but leaves the door open to efficient, practical solutions to specific problems. In the proactive approach as a security enhancement to centralized systems is considered, and a practical proactive pseudo-random generator with applications to secure sign-on is presented.
Another basic task that has been `proactivized' is secret sharing, and in particular verifiable secret sharing (i.e., secret sharing resilient against malicious faults). This algorithm played a key role in proactive solutions for public-key cryptosystems and in particular in proactive signature systems (extending the threshold signature). Proactive solutions were found for the DSS signature algorithm and for RSA. Proactive signatures are a powerful tool. They were used in to provide a proactive, automated solution to key refresh. Namely, show how to use cryptography to ensure authenticated and secret communication among servers, with recovery from penetrations and key exposures. This provides an alternative to manual key refresh.

6. PROACTIVE SECURE COMMUNICATION

Another cryptographic task where the proactive approach seems called for is maintaining authenticated and secret communication among a set of parties. Here the parties must keep the integrity and secrecy of the relevant keys: shared keys (such as session keys), private signature and decryption keys, as well as the integrity of public keys (of other parties).
It is a standard practice to keep two levels of keys: short-lived `session' keys, and long-lived `master' keys. The `master' keys are used to periodically refresh the `session' keys. This provides recovery from exposure of the session keys - but not of the master keys. Protecting against the exposure of the master keys is considered a hard problem; when deemed necessary, it is achieved via manual master key refresh process, done periodically but infrequently.
Some mechanisms, most notably perfect forward secrecy (e.g., implemented by the IP-SEC standard) provide protection of past session keys from a future exposure to the master keys. However, this does not protect future session keys from active impersonation attacks. Proactive security provides a more complete solution, where exposing a master key does not reveal either future or past session keys even from active attackers - achieving the same effect as the manual key refresh process, at much lower costs.
A solution may seem straightforward at first: at each refreshment phase, each party will choose a new pair of public and private keys, distribute the new public key to other parties (signed using the old secret key), receive new public keys (signed) from each other party, and then use the new public keys to agree on new shared keys. However, an intruder who also controls the communication links can successfully impersonate an attacked party by sending a fake public key on its behalf. Moreover, if the attacker broke into two machines, it can select fake public keys for both of them and thereby permanently `insert' itself between the two parties. This way the attacked parties lose their ability to authenticate each other, even long after the intruder lost control of the machines.

7. PROACTIVE SYSTEM

Proactive security shows how to maintain the overall security of a system even under such conditions. In particular it provides automated recovery of the security of individual components, avoiding the use of expensive and inconvenient manual processes (except for some `aggressive` attacks, which cannot be prevented but are definitely and clearly detected). The technique combines two well-known approaches to enhance the security of the system: distributed (or threshold) cryptograph, which ensures security as long as a threshold (say half) of the servers are not corrupted; and periodic refresh (or update) of the sensitive data (e.g. keys) held by the servers.
This way, the proactive approach guarantees uninterrupted security as long as not too many servers are broken into at the same time. Furthermore, it does not require identification when a system is broken into, or after the attacker loses control; instead, the system proactively invokes recovery procedures every so often, hoping to restore security to components over which the attacker lost control. Proactive security is highly desirable in many realistic settings, in particular: When a high level of security is required, together with fault tolerance (as redundancy improves fault tolerance but opens more points for attack).
To ensure acceptable level of system security using weakly secure components such as most commercially available operating systems. (Examples of specific applications are given below.) Recent results show that much fundamental cryptographic functionality may be achieved even under the proactive security model - as long as most components are secure most of the time. In particular, proactively secure protocols have been devised for the following problems:
1. Secret sharing
2. Discrete-log-based digital signatures, and in particular DSA
3. Secure end-to-end communication
4. RSA , and in particular generation of the RSA shared key
5. Pseudo-random generation
6. Key distribution center

This substantial set of known results in proactive security did not yet produce any practical security product or solution. (In fact, there are only a few deployments of

distributed security) the most well known may be the SET credit card standard's certificate authority.

The creation of such a proactive solution is non-trivial, as the protocols are often quite complex and nontrivial to implement. Here we assume some features for the project, to allow practical deployment of proactive security.

The main new contributions (assumptions) are:

1. A secure initialization mechanism, with reasonable, practical requirements from the computer and operating system. Specifically, all we require is a secure boot process (which is a good idea anyway, against viruses - and easily done with signed code); and a per-machine secret-private key pair, with the public key protected from modification (e.g. in ROM or write-once EEROM), and the secret key in erasable memory (e.g. disk). Previous results required storage of parameters specific to the particular application (such as the group's public key) in secure storage, which is not practical.

2. A set of application program interfaces (APIs) that allow the use of the toolkit to improve security, specifically provide security in spite of break-ins into computers, of existing applications, as well as the development of new applications which are proactive secure.

The security of any proactive solution relies heavily upon its correct architecture and integration with existing, non-proactive, operating system. The design of system, which does not view the proactive model as series of protocols but, rather, as a security enhancement of the operating system which transforms it into a proactively secured system via the appropriate use of proactive protocols, has not been defined nor implemented in the past. We show that it is possible to transform general applications which are required to remain secure for long periods of time to operate in a proactive environment, namely proactivizing applications.

To this end, we define architecture for a proactive operating environment which serves as a platform on which standard applications can be proactivized. This operating environment consists of a network of servers which are set up once, which we call the proactive network.

Each server is instantiated at boot time by the operating system and is checked periodically, also by the operating system.

Servers can recover data (both public and private data) from other servers in the proactive network, if such data is corrupted or lost. Once the proactive network is set up, any application can run on the top of the network and request proactive services by the means of API.

Assumption: we are assuming here that we have already a system where threshold cryptography is implemented.

8. CONCLUSIONS

The proactive program should first provide a toolkit consisting of communication and cryptographic primitives who are needed to implement any proactive algorithm. In addition, it should be able to support multiple instances of proactive applications running concurrently. An essential component of such a program is a module responsible for refreshing the on-going proactive tasks of the system.

Proactive schemes are proposed as a countermeasure to mobile adversaries. A proactive threshold cryptography scheme uses share refreshing based on different cryptography function, which enables servers to compute new shares in collaboration without disclosing the service private key to any server. The proposed direction for the future work could be the spanning of number of nodes in a large internet environment. Also some verification techniques could be incorporated with this application for more secure communication.

REFERENCES

[1]. Rajkumari Retoliya. "A Novel Approach Share Key Refreshing for Long Term Protection in Distribute Cryptography by Protective Security". IJCSET | July 2011,Vol 1,Issue 6,290-295.

[2]. D. Boneh and M. Franklin. "Efficient generation of shared RSA keys". In Proc. Crypto '97, pp. 425-539.

[3]. R. Canetti, R. Gennaro, A. Herzberg and D. Naor, "Proactive Security: Long-term protection against break-ins". CryptoBytes: the technical newsletter of RSA Labs, Vol. 3, number 1 - Spring, 1997.

[4]. P. Feldman. "A Practical Scheme for non-interactive verifiable secret sharing". In Proc.28t Annual Symp. on Foundations of Computer Science, pp. 427-437. IEEE, 1987.

[5]. Y. Frankel, P. Gemmell, P. Mackenzie, and M. Yung. "Optimal resilience proactive public-key cryptosystems". In Proc. 38th Annual Symp. on Foundations of Computer Science. IEEE, 1997.

[6]. Y. Frankel, P. Gemmell, P. Mackenzie, and M. Yung. Proactive RSA. In Proc. of Crypto97. 12. P. Gemmell. "An introduction to threshold cryptography". In Cryptobytes, Winter 97, pp. 7-12, 1997.

[2] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness Theorems for Noncryptographic Fault-Tolerant Distributed Computations. In Proc. 20th Annual Symp. on the Theory of Computing, pages 1{10. ACM, 1988.

[3] G. R. Blakley. Safeguarding cryptographic keys. In Proc. AFIPS 1979 National Computer Conference, pages 313{317. AFIPS, 1979.

[4] D. Boneh, Ed Felten, Bill Aiello, and Matt Franklin. http://gump.bellcore.com:7700.

[5] G. Brassard, D. Chaum and C. Crepeau, \Minimum Disclosure Proofs of Knowledge", Journal of Computing and System Sciences, Vol. 37, No. 2, 1988, pp. 156-189.

[6] D. Chaum, C. Crepeau, and I. Damgard. Multiparty Unconditionally Secure Protocols. In Proc. 20th Annual Symp. on the Theory of Computing, pages 11{19. ACM, 1988.

[7] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults. In Proceeding 26th Annual Symposium on the Foundations of Computer Science, pages 383{395. IEEE, 1985.

[8] R. Canetti and A. Herzberg. Maintaining security in the presence of transient faults. In Y. Desmedt, editor, Advances in Cryptology | Crypto '94, pages 425{438, 1994. SpringerVerlag. Lecture Notes in Computer Science No. 839.

[9] R. Canetti, S. Halevi, and A. Herzberg. Maintaining Authenticated Communication in the Presence of Break-ins. In Proc. 16thth ACM Symp. on Principles of Distributed Computation.ACM, 1997.

[10]. http://www.research.sun.com/projects/crypto.

[11]. http://www.rsa.com/rsalabs/.

[12]. L. Ertaul and N. Chavan, "Security of Ad Hoc Networks and Threshold Cryptography", in MOBIWAC 2005.