ISSN:0975-9646
Sarvani Rallabandi et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (5) , 2011, 2416-2420

Spam Weeding, Detection and Filtering on P2P System

Sarvani Rallabandi , I M V Krishna
Department of Computer Science & Engineering, PVP Siddhartha Institute of Technology
Vijayawada, Andhra Pradesh, India

Abstract -- **Domain keys identified mail (DKIM) defines a mechanism for using digital signatures on E-Mail at the domain level, allowing the receiving domain to confirm that E-Mail came from the domain it claims to. Erasure codes introduce pollution attack, an attack in which the adversary injects packets to disrupt the erasure decoding procedure and consequently denies the authentication service to the receiver. This paper uses the agent for checking the spam E-Mail by using DKIM and proposes a new lightweight, pollution-attack resistant multicast authentication scheme (PARM), which generates evidence that receiver agent can validate on a fast, per-packet basis. Using agent, the authenticity of the system will increase and also our system will be more secure. Because of using the temporal key, time will be saved for generating evidence to the same message.**

Keywords -- **Peer To Peer, Spam Weeding, DKIM, Encryption**

I.INTRODUCTION:

E-Mail systems are vulnerable to a number of security risks. In an E-Mail exchange process, receiving party cannot be sure that the E-Mail is from the actual sender. Similarly, sending party is not able to make sure that intended recipient has received E-Mail. Since E-Mail is delivered with other Internet traffic over the same transport service, it is vulnerable to eavesdropping. Also, malicious content in E-Mail content might enter user host through E-Mail client.

The Peer-to-Peer (P2P) systems have emerged to be a focused architecture in both significant social and technical points of views. The P2P architecture refers to a class of systems and applications that employ distributed resources to perform a critical function in a decentralized manner. P2P systems usually provide infrastructure for interpersonal collaborative communities that share computing power and storage space.

P2P is used primarily to exchange pirated or inappropriate audio, video, and software files, user must take appropriate measures to guard against the associated legal liability. User should also be concerned about employee use of P2P applications that have no direct positive business impact. The effects of running P2P applications, downloading large files, allowing P2P users to upload files from shared desktop folders can slow down network performance. This has a negative impact on the performance of business-critical applications. All of these threats currently exist in a significant number of enterprise networks and must be managed.

A. OBJECTIVE OF THE STUDY

Two main objective of this paper are

- To propose an algorithm for detecting and filtering Spam in Peer-to-Peer system by modifying DKIM by incorporating sender evidence in the E-Mail messages thereby making Spam Weeding possible.
- To detect and provide countermeasures to two kinds of attacks like Sybil attack and pollution attack in a P2P system.

II.DESIGN OF SPAM WEEDING SENDER SYSTEM

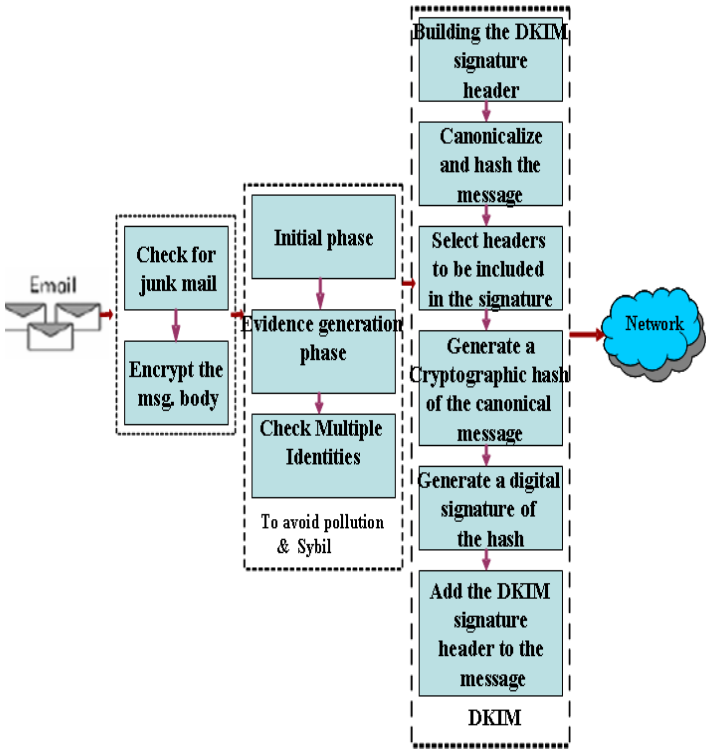
Email
Check for junk mail
Encrypt the msg. body
Initial phase
Evidence generation phase
Check Multiple Identities
To avoid pollution & Sybil
Building the DKIM signature header
Canonicalize and hash the message
Select headers to be included in the signature
Generate a Cryptographic hash of the canonical message
Generate a digital signature of the hash
Add the DKIM signature header to the message
DKIM
Network

Figure 2.1 : Design of the Sender System

A. CHECK FOR JUNK E-MAIL

Using common spam characteristics user can identify the spam and also control the spam

1) Common Spam Characteristics

Spam characteristics appear in two parts of a message

- E-Mail headers
- Message content

2) E-Mail Header

E-Mail headers show the route an E-Mail has taken in order to arrive at its destination. They also contain other information about the E-Mail, such as the sender and recipient, the message ID, date and time of transmission, subject and several other E-Mail characteristics. Most spammers try to hide their identity by forging E-Mail headers or by relaying mail to hide the real source of the message.

3) MESSAGE CONTENTS

Apart from headers, spammers tend to use certain language in their E-Mails that companies can use to distinguish spam messages from others. Typical words are free, limited offer, click here, act now, risk free, lose weight, earn money, get rich, and (over) use of exclamation marks and capitals in the text. Spam can be blocked by checking for words in the E-Mail body and subject, but it is important that filter words accurately since otherwise user might be blocking legitimate mails as well.

B. ENCRYPT THE MESSAGE BODY

After the spam analysis the body of the E-Mail is encrypted using Digital Signature Algorithm (DSA).

C. INITIALIZATION PHASE

In this phase, we define how to generate a temporal key pair, which contains a Temporal Secret Key (TSK) chain and a Temporal Public Key (TPK), using a one-way hash function. The sender creates the evidence of a packet from a TSK chain, and the receiver validates the evidence of a received packet with the TPK. Before communicating with receivers, the sender must generate the TSK chain and TPK in advance.

First, the sender generates k n-bit random numbers (R_0, R_1,…, R_{k-1}) and denotes this set of numbers as TSK_0 of the TSK chain. Then, the sender uses the oneway hash function h to recursively generate the remaining TSK_5 of the TSK chain. By applying the hash function to each member of the previous TSK, the sender can produce the next TSK. For example, TSK_1 is generated by hashing each element in TSK0 i.e. TSK1=(h(R0), h(R_1),…, h(R_{k-1})). The TSK chain has a length of L and is represented as (TSK_0, TSK_1,…, TSK_{L-1}). The temporal public key (TPK) is created by hashing every element of TSK_{L-1}.

R_0 denotes the randomly generated number, and the arrows specify the direction of the one-way hash function h. Thus, h(R_0) is the hash result of R_0, and $h^2(R_0)$ is the hash result of h(R_0). The set of the elements in the same row comprises a TSK elements array, e.g. TSK_0=(R_0, R_1,…, R_{k-1}) and TSK_1=(h(R_0), h(R_1),…, h(R_{k-1})). The elements of the last row form the TPK.

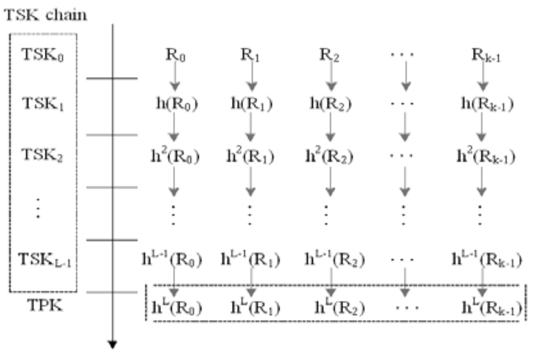
TSK chain
TSK₀
TSK₁
TSK₂
TSK_{L-1}
TPK
R₀ R₁ R₂ R_{k-1}
h(R₀) h(R₁) h(R₂) h(R_{k-1})
h²(R₀) h²(R₁) h²(R₂) h²(R_{k-1})
h^{L-1}(R₀) h^{L-1}(R₁) h^{L-1}(R₂) h^{L-1}(R_{k-1})
h^L(R₀) h^L(R₁) h^L(R₂) h^L(R_{k-1})

Figure 2.2 : Temporal Key Pair Generation

After successful generation of the TSK chain and TPK, the sender provides receivers with the TPK. Since receivers will use the TPK to determine the validity of received packets, it is vital that the sender sign the TPK with a digital signature to protect it during distribution. Otherwise, an attacker can convince receivers to accept a forged TPK. Consequently, all valid packets will fail to pass evidence validation. The receiver stores the TPK if it verifies the signature.

D. EVIDENCE GENERATION PHASE

Prior to broadcasting a message, the sender must generate for each packet the evidence, or verification information, which allows receivers to determine the validity of a packet. Since each packet is augmented with evidence, the evidence generation phase should be lightweight and fast. For a given temporal key pair, the sender needs to maintain a usage table, that tracks the number of times each column index of the TSK elements array is used. The row index denotes the column index of the TSK elements array, while the row usage tracks the number of uses of the corresponding index.

Table 2.1 : Usage Table

Usage Table					
Chain Index	*TSK_0*	*TSK_1*	*TSK_2*	…	*TSK_{k-1}*
Usage Amount	*l-3*	*14*	*0*	…	*l-1*

To generate evidence EM for a packet M, the sender first hashes the packet with a one-way hash function h. The hash value is divided into a set of p segments, denoted s=(i_0, i_1,…, i_{p-1}), where each segment size is b-bits. Interpreted as an integer between 0 and 2_{b-1}, each segment in the set S represents a column index of the TSK elements array. For each index i, the sender determines the TSK based upon the usage of i by selecting TSK($_{L-1}$)-ai, where ai denotes the usage of i.

The sender chooses the last TSK of the chain, TSK_{L-1}, if i has never been used. Once the sender determines the TSK, it chooses the i-th element of the selected TSK. For example, if i0 used L-1 TSK elements, then the sender chooses the i0-th element of TSK0, which is R0. Since each segment of s corresponds to an index of the TSK elements array, the sender produces p elements, which constitutes the evidence of the packet. After appending the evidence to the packet, the sender can finally broadcast the packet to the receiver.

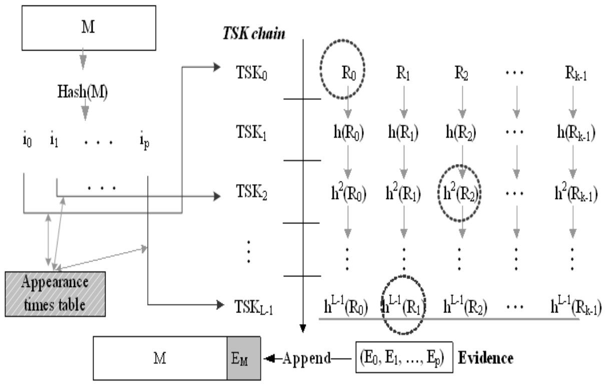

Figure 2.3 : Evidence Generation Phase

E. CHECK FOR MULTIPLE IDENTITIES

This module avoid the sybil attack by checking the user's address and time of sending the message.

III.DESIGN OF SPAM WEEDING RECEIVER SYSTEM

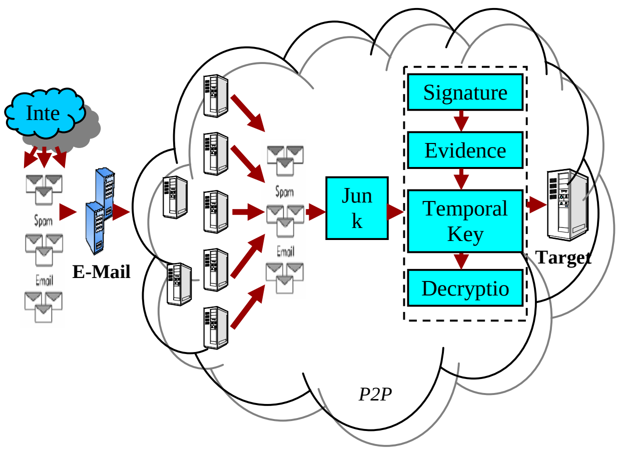

Figure 3.1 : Receiver System

A. DKIM VERIFICATION

When a DKIM-compliant MTA receives an E-Mail message, that it decides it must verify, the message may be signed, or unsigned. The message is considered to be signed if there is a valid DKIM Signature header. The verifier must carefully check the signature header for validity.

B. VERIFYING A DKIM SIGNATURE

Using the contents of the i=, d=, and s= fields in the signature header, the verifier determines the desired key identity, and then uses the q= field and retrieves the key from the specified key store. For q=dns, the key is retrieved by getting DNS TXT records for "selector._domainkey.domain". The verifier must then validate the retrieved key record, and extract the public key from it.

Any failures in this process result in the signature's being declared invalid. The verifier now uses the c=, h=, and l= (if present) fields to recreate the canonical message as originally signed. Using the a= field to determine the hash and encryption algorithms, it then computes the hash on the canonical message, decrypts the signature, and compares the two resulting hash values. If they are the same, then the signature is verified

C. CHECKING THE SIGNING PRACTICES

If there is no valid signature, or if the signing identity does not match the address in the message's From header, the verifier must check the signing practices of the domain in the From address. The verifier retrieves the policy through a DNS query. The domain for the query is obtained from the From address

D. THE VERIFIER'S DECISION

The verifier does with all this information – whether a signature was present or not, whether it verified or not, what the sender's signing practices say – is entirely up to the verifier. Verifiers may certainly treat messages with failed signatures as being more "suspicious" than those lacking signatures, but there are reasons for message signatures to fail that do not reflect on the legitimacy of the message.

E. EVIDENCE VALIDATION PHASE

Upon receiving a packet, the receiver can use the TPK to immediately check the validity of the attached evidence. To forge a packet, the attacker must generate proper evidence for a packet, which is difficult without knowledge of the TSK chain. As with the sender, the receiver must also maintain a usage table for each column index of the TSK elements array based on received packets.

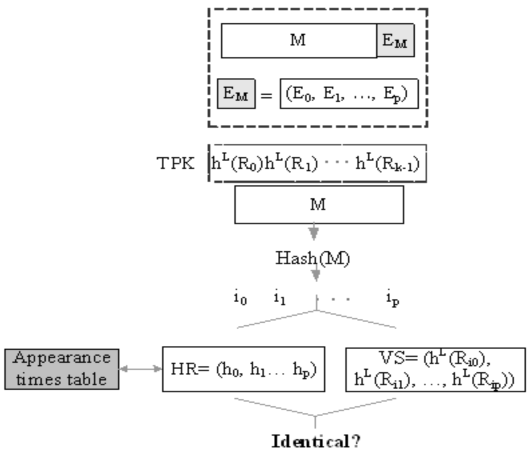
M
E_M
$E_M = (E_0, E_1, \ldots, E_p)$
TPK
$h^L(R_0)h^L(R_1) \cdots h^L(R_{k-1})$
M
Hash(M)
i_0 i_1 ... i_p
Appearance times table
$HR = (h_0, h_1 \ldots h_p)$
$VS = (h^L(R_{i0}), h^L(R_{i1}), \ldots, h^L(R_{ip}))$
Identical?

Figure 3.2 : Evidence Validation Phase

The procedure of the evidence validation phase is similar to that of the evidence generation phase. After receiving a packet containing evidence EM, the receiver separates the evidence, denoted $EM=(e_0, e_1,\ldots, e_{p-1})$, from the packet M. To validate the evidence for this packet, the receiver hashes M with the one-way hash function h, which is identical to the one-way hash function used by the sender in the evidence generation phase.

The receiver divides the hash value h(M) into p b-bit segments, denoting these segments as the set $(i_0, i_1,\ldots, i_{p-1})$. By interpreting each segment as an integer between 0 and 2_{b-1}, each segment can represent a column index of the TSK elements array. Each index i, along with its usage a_i, determines the number of times to hash the corresponding element e_i of the evidence. Given an index and its usage, the receiver should perform a_{i+1} hashes on the corresponding element of the evidence.

Thus, if index i has never been used before, the receiver need only hash e_i once. The ensuing set of hash results from every element of the evidence is denoted by $HR=(h_0, h_1,\ldots, h_{p-1})$. The receiver selects the verification subset $VS=(hL(Ri_0), hL(Ri_1),\ldots, hL(Ri_{p-1}))$ from the TPK, where $h_L(R_i)$ is the i-th element of the TPK. The receiver considers the evidence valid if the two sets, HR and VS, contain identical elements, accepting the packet with valid evidence and dropping it otherwise.

F. TEMPORAL KEY RENEWAL PHASE

Periodic renewal of used TSK elements is necessary to ensure secure communications between the sender and its receivers. User define a threshold value T in key renewal phase. $UTSK_0$ represents the number of used elements in TSK_0 (the first TSK of the TSK chain) since the last temporal key renewal, and the set (j_0, j_1,…, j_{t-1}) denotes the indexes of the used elements. When $UTSK_0$ exceeds the threshold T, new elements are required.

The sender generates $UTSK_0$ new random numbers for the used indexes of TSK_0. Using these random numbers, the sender creates the partial TSK and the partial TPK with the one-way hash function h by following the temporal key generation procedure of the initialization phase. The sender then updates its copy of the TSK chain with the partial TSK elements. Since the receiver must also update its TPK, the sender concatenates the new partial TPK with its digital signature Sign(Partial TPK), which it then encodes with erasure codes and appends to outgoing packets. Successful renewal of the TSK chain and TPK, the sender and receiver may resume evidence generation and verification of packets.

G. DECRYPTION

Final stage is to decrypt the encrypted message using the receiver's private key.

H. POLLUTION ATTACKS RESISTANT

- Attacker is unable to produce valid evidence for attack packets
- Attack packets can't pass evidence validation procedure at receiver
- Only valid packets will be accepted

IV **ALGORITHM**

This chapter explains about the algorithm used for developing spam detection system to stop spam and control pollution attack and sybil attack.

A. MODULE DESCRIPTION

The following table shows the algorithm name and the use of algorithm.

Table 4.1: Algorithm usage

Name	Uses
CHKJMAIL	Checking junk mail content
MSGENCRYPT	Encrypt the message
INITIAL	Initialization phase
EGENERATION	Evidenced Generation
DKIM	DKIM Header Generation
SIGCHECKING	Signature Checking
DECRYPTION	Decrypt the message

1) CHECK FOR JUNK E-MAIL

```
Procedure CHKJMAIL(EM)
        //EM ⟵ E-Mail
        //Assume the given E-Mail is contain 'to', 'from',
        'subject' and 'body' field. To, //from, subject are consider
        as E-Mail Header and body contains the actual //message
        SPAMCHAR(E-Mail_Header)
        SPAMCHAR(E-Mail_Body)
End CHKJMAIL
```

2) ENCRYPT THE MESSAGE BODY

```
Procedure MSGENCRYPT(M)
        //M ⟵ Message Body
        String CT ⟵ DSA.encrypt(M)
End MSGENCRYPT
```

3) INITIAL PHASE

```
Procedure INITIAL()
        Declare K,L,TSK[][].TPK[][]
        Ro ⟵ random(n)
        For I ⟵ 1 to l do
          For j ⟵ 1 to k do
            If i==l then
                TPK[i][j] ⟵ H(R)
                  Else
                   TSK[i][j] ⟵ H(R)
                        R ⟵ TSK[i][j]
                      End
             End
End INITIAL
```

4) EVIDENCE GENERATION

```
Procedure EGENERATION(P,Q)
  //p ⟵ packet which is going to transfer
        //Q ⟵ Sequence number of the packet
        H_Value ⟵ h(P||Q)
        HValue_Length ⟵ H_Value.length()
        Resulted_Packet ⟵ P||Q
        While(HValue_Length)
          For I ⟵ 1 to n do
                S[x] ⟵ Byte[H_Value]
                End
                Temp ⟵ s[x]
                H_Value=H_Value-temp
        HValue_Length=H_Value.length()
                X++
        End //while
        Flag=true
        For I ⟵ 1 to x do
          While(flag)
             S_Evidence ⟵ random(TSK)
                If S_Evidence    !=Content(Usage_Table)
            S[i] ⟵ s[i].append(S_Evidence)
          Usage_table ⟵ add(S_Evidence)
                Flage ⟵ false
                  Else
                        Flag ⟵ true
                End //while end
        Resulted_Packet=Resulted_Packet || S[i]
        End //for loop
End EGENERATION
```

5) DKIM MODULE

```
Procedure DKIM(E-Mail)
        Call Build_DKIM()
        Call Canonicalize()
        Call Select_Header()
        Call Cryptographic_Hash()
        Call Digital_Sig_Hash()
        Call Cancat(DKIM_Sig || Message)
End DKIM
```

6) SIGNATURE CHECKING

```
Procedure SIGCHECKING(Resulted_Packet)
        //SIGCHECKING is just reverse process of EGENERATION()
        VS ⟵ VSet()
End SIGCHECKING
```

7) MESSAGE DESCRIPTION

```
Procedure DECRYPTION(CT)
        //CT ⟵ ciphertext of the encrypted message
        M=DSA.decrypt(CT)
End DECRYPTION
```

V.CONCLUSION

Spam is considered as a serious problem since it causes huge losses to the organization due to bandwidth consumption, mail server processing load, and user's productivity. The objective of this paper is to design a SpamWeeder for a P2P system which secures E-Mail server from receiving and sending Spams. This paper gives the idea of fighting spam E-Mail, which allows users to precisely expose parties engaged in E-Mail address trafficking and block all E-Mail from particular party belonging to a given trafficking chain.

A system designed for weeding out Spams from a P2P network. DomainKeys Identified Mail (DKIM) defines a mechanism for using digital signatures on E-Mail at the domain level, allowing the receiving domain to confirm that E-Mail came from the domain it claims to.

REFERENCES

[1] Tu Ouyang and Michael Rabinovich, "*Weeding Spammers at the Root : A Precise Approach to Spam Reduction*," IEEE Transactions on EECS Department, 2008.

[2] Barry Leiba, Jim Fenton, "*DomainKeys Identified Mail (DKIM) UsingDigital Signatures for Domain Verification*," Journal of Foundations and Trends in Information Retrieval, pp. 538 - 549, January 2008

[3] Ya-Jeng Lin, Shiuhpyng Shieh, Warren W. Lin, "*Lightweight, Pollution - Attack Resistant Multicast Authentication Scheme*," ASIACCS'06, March 21 – 24, November 2006.