|  |
| --- |
|  |
| CSE 565: COMPUTER SECURITY |
| PROJECT 3: EMAIL FORENSICS WITH DKIM |
|  |
|  |
| **PRASANNA CHANDRASHEKHAR PAI UB#:50132731**  **GAURAV GOSAVI UB#:50096832**  **KRISHNAKANT CHAVALI UB#:50132680**  **AVANI CHANDRASHEKHAR ATRE UB#:50134720** |
| **DATE OF SUBMISSION: NOVEMBER 25, 2014** |

|  |
| --- |
|  |

**INDEX**

**Section 1.1……………………………………………………………………………………… 3**

**Section 1.2……………………………………………………………………………………… 5**

**Section 1.3……………………………………………………………………………………… 6**

**Section 1.4……………………………………………………………………………………… 7**

**Section 1.5……………………………………………………………………………………… 8**

**Section 2.1……………………………………………………………………………………… 9**

**Section 2.2……………………………………………………………………………………… 10**

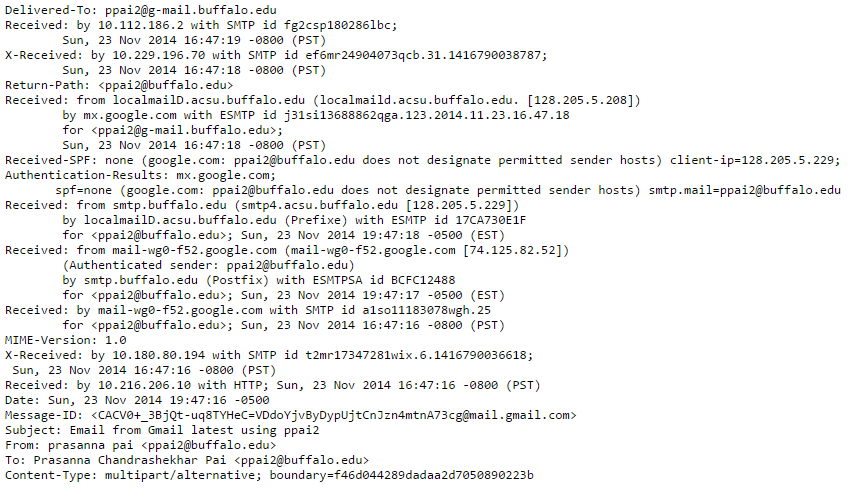
**Section 2.3……………………………………………………………………………………… 11**

**Section 2.4……………………………………………………………………………………… 12**

Phase 1: Investigate E-mail Spoofing

Section 1.1

Email header basically contains all the details from sender to receiver including the intermediate path taken from the servers by the email to reach the recipient with the timestamps involved. This has been explained in the below mentioned snapshot.



This mail has been sent from Gmail to UB Mail ID. Please find below the details for each field type:

**Delivered To**: Mail ID of the recipient where the email has been delivered

**Received**: This field type basically contains all the routing details in the servers including their IP addresses from the sender’s and receiver’s perspective. Here, the ‘Received: by’ tag is used to demonstrate how the email has been routed by the servers before reaching the receiver. The email was received by server with IP address 10.112.186.2 at local time 16:47:19 on 23rdNov, 2014 using SMTP protocol used by the local server before reaching the receiver. The same context can be applied to ‘Received: by’ tag which received the email from the sender. Here, the email was being received from the sender by server with IP address 10.216.206.10 at local time 16:47:16 on 23rd Nov, 2014. Each server that handles the message stamps it with the local timestamp (and not the receiver/sender’s timestamp) due to which we see different timestamps in a single header.

**Return-Path**: This is same as that of Reply-To functionality which specifies the address of the sender which is sender’s email ID i.e. ppai2@buffalo.edu.

**Received-SPF**: This field checks for the authenticity of the sender's mail server with the Sender Policy Framework (SPF) and checks if it is allowed to send the mail to the designated recipient's domain. SPF helps in preventing spammers and phishers in forging emails and catches spam mails in spam filters. The Field usually associates values such as pass/fail/neutral/none which determines if the sender domain is valid. Here, it means that the sender‘s domain could not be authorized since the receiver i.e. university mail server does not have SPF framework configured at their end.

**Authentication Results**: This details the above mentioned functionality which is verified with DKIM. DKIM states that if the value of Received-SPF is ‘fail’ then the message should be treated as an unsigned message. In this case, it is ‘none’ which states that the SPF framework has not been configured at the receiver’s domain and the checking software cannot confirm if the sender’s domain is valid. The email will be successfully delivered to the receiver but can be a likely case of email spoofing if used by spammers or phishers who might take the advantage of the receiver’s domain to send spams.

**MIME Version**: Multipurpose Internet Mail Extension is an Internet standard which extends the format of email which is 1.0 in this header

**Date**: Timestamp when the email was sent by the sender i.e. Sun, 23 Nov 2014 19:47:16

**Message-ID**: This is a unique identifier used to identify an email.

**Subject**: Subject title of the email as listed by the sender.

**From**: Sender’s email address

**To**: Receiver’s email address

The ‘From’ and ‘To’ fields can easily be forged since it is the sender who determines the fields.

**Content-Type**: multipart/alternative

The type of content present in the message i.e. text/plain or text/html

Reference: http://www.ietf.org/rfc/rfc4408.txt

Header from UB Mail account to UB Mail account using UB Mail ID:



**MIME Version**: Multipurpose Internet Mail Extension is an Internet standard which extends the format of email which is 1.0 in this header

**Received**: This message was received by host server with IP 10.112.186.2 using HTTP protocol at local time 21:53:12 on 18th Nov, 2014

**Date**: The time when the message was sent by the sender i.e. 19 Nov 2014 00:53:12 which is the local time of the sender

**Delivered-To**: Recipient’s email address where the message has been sent

**Message-ID**: Unique identifier assigned to the message

**Subject**: Subject title of the message

**From**: Sender’s email address

**To**: Recipient’s email address

**Content-Type**: type of the content present in the message i.e. multipart/alternative which involves text/plain and text/html in this message

Section 1.2

Message ID in the email header is a unique global attribute in the message. It can contain format including local host domain’s name along with date and timestamp or some other encoded format which can be a subset of the sender’s email address. Message ID is unique for each message and every message contains exactly one unique instance message ID.

Message ID can be easily spoofed. It can be forged by using the same message ID from the same website of some previous mail and used by the attacker in some other mail which is sent to the target. The only way to detect forging in this case is to check if the message ID is unique in the header and is not repetitive from previous mails.

It can also be forged from some email client which has its own way of formatting the message ID. For instance, in case of other mail servers such as Yahoo or Hotmail, they have message ID which contain some unique string followed by the sender domain such as yahoo.com. The forger can replace the string with some reference which can be easily accessed by him and append it with legal sender domain such as yahoo.com which might easily pass as legitimate message ID.

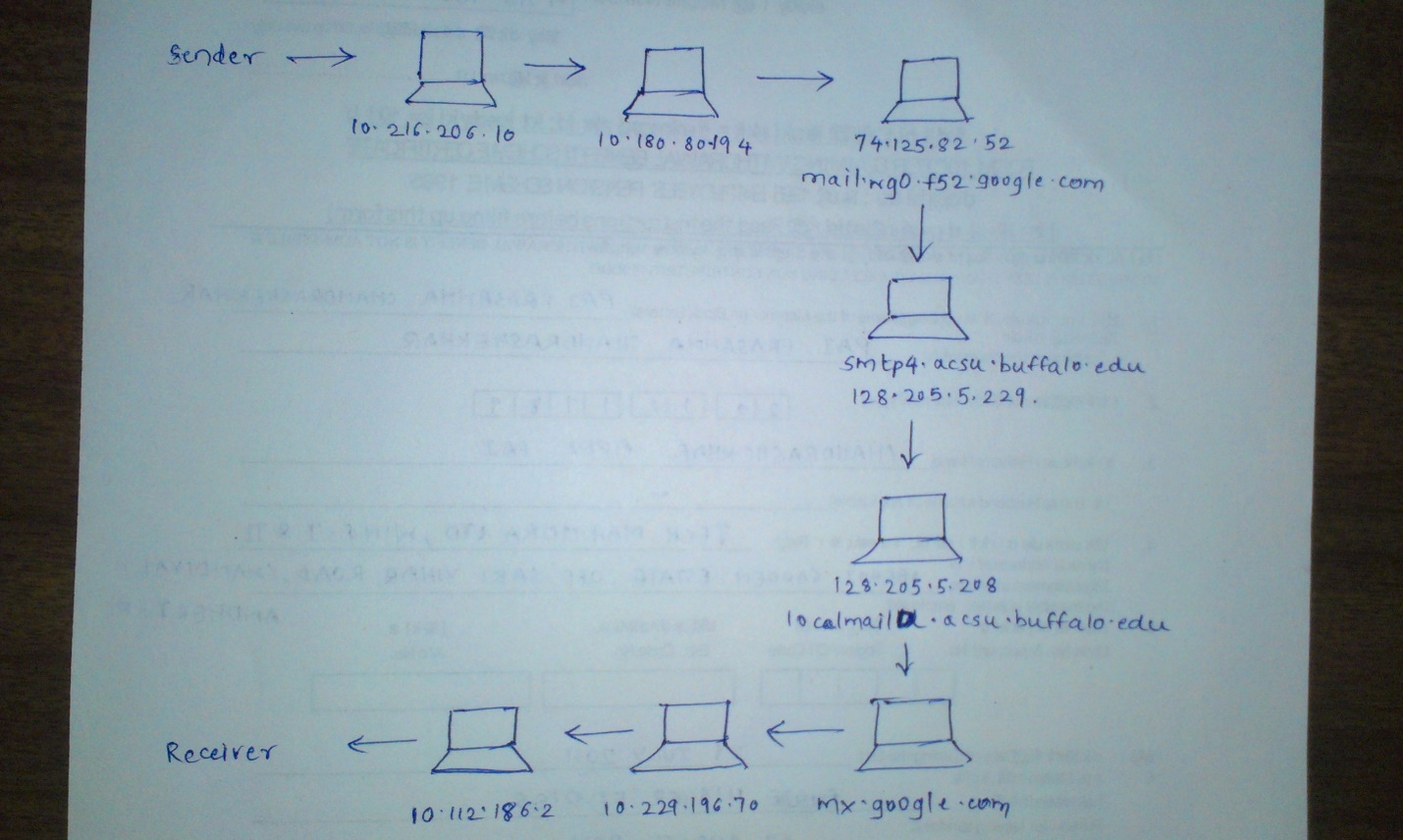
Reference:

<http://en.wikipedia.org/wiki/Message-ID>

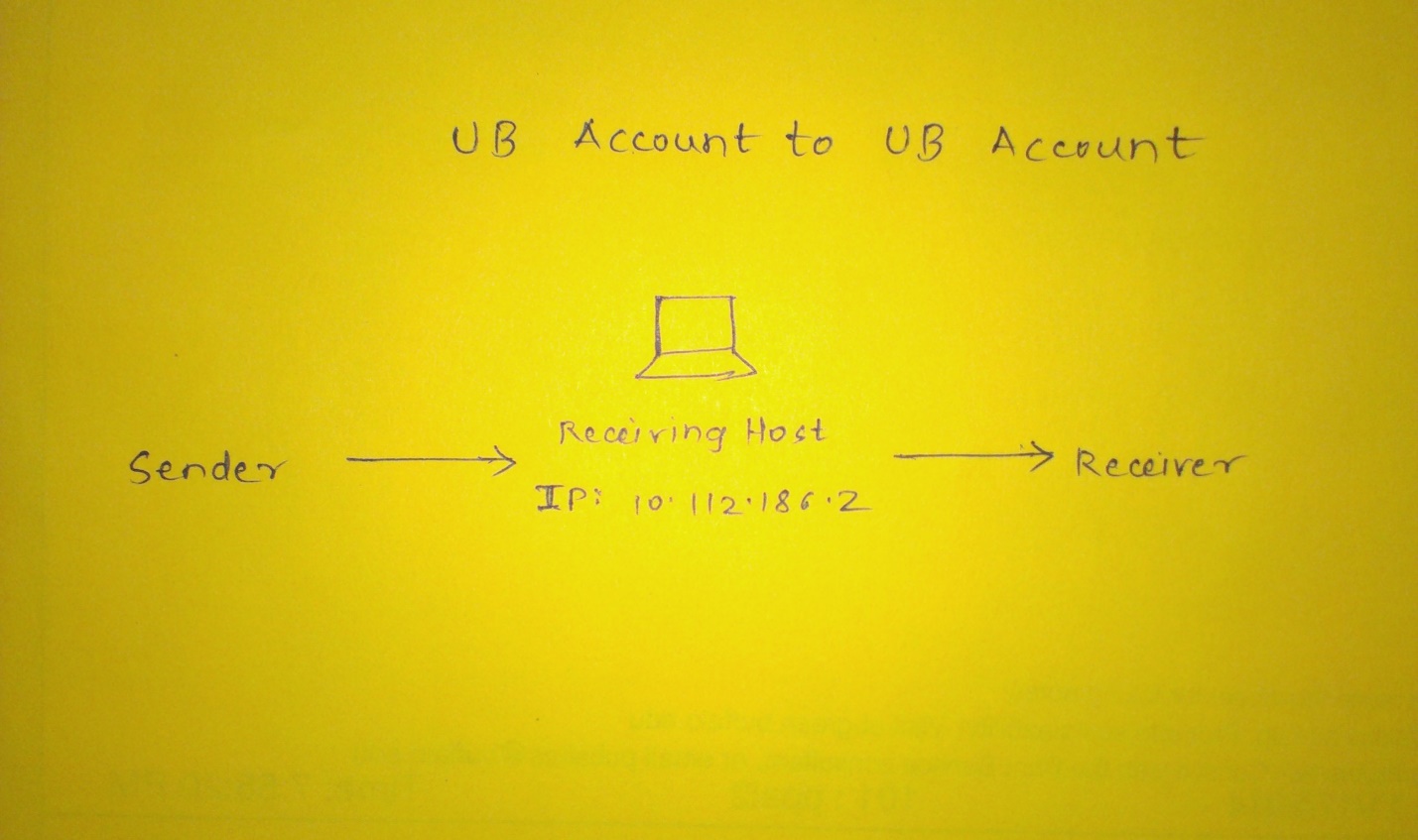
<http://www.forensicswiki.org/wiki/Using_message_id_headers_to_determine_if_an_email_has_been_forged>

Section 1.3

Network diagram of mail sent from Gmail account to UB account’s email address



Network diagram of mail sent from UB account to UB account’s email address



Section 1.4

We cannot characterize one of these emails as spoofed mail. Email spoofing involves changing the ‘From’ field type or any field type in the email header with some illegitimate content which cannot be verified or authenticated. In this case where we are sending email from Gmail address to UB account’s email address using UB mail address, we are just sending an email from one legitimate email address to other email address (legitimate email address since we are asked for confirmation code when configuring the UB account email on Gmail account without which we cannot configure UB Mail ID on Gmail account). The email address is verified while being created which makes it difficult to forge. Also the sender’s email ID appears in the Return-Path of the email header at the recipient’s end which verifies the sender’s authenticity. This activity can be termed as Legitimate Email Spoofing since we are using legitimate email addresses to send or receive emails.

In the second case where we are sending email from UB Account to same account’s email address, we are sending email from authorized email address to same email address which can be considered as a spam. We are basically sending email where the same email address appears in the ‘From’ tag as well as ‘To’ tag. This can be considered as a version of email spoofing but does not fully classify for spoofing. This process can be termed as self - sending spam.

Section 1.5

Normally, an email header is composed from bottom to top along the timeline as the mail passes from sender to series of servers to receiver. In this way, if the mail has been forged then there is possibility of some repeated fields such as ‘Received: from’ or ‘Received: by’ or some Return-Path which ideally contains the sender’s email address. The forger is more likely to change the ‘Received: by’ or ‘Received: from’ fields and change them to some illegitimate content containing some server’s IP address which cannot be traced whatsoever. Also if the sender’s email address differs in the ‘From’ field and Return-Path field then the email is more likely to be forged. Also the receiver can check if the Reply-to functionality is directing to same address as the sender’s address in the Return-Path. If not, then the email header has been forged by the attacker. Otherwise, it is difficult to trace email spoofing just by checking the header from the receiver’s perspective.

Phase 2: Separation of HAM and SPAM

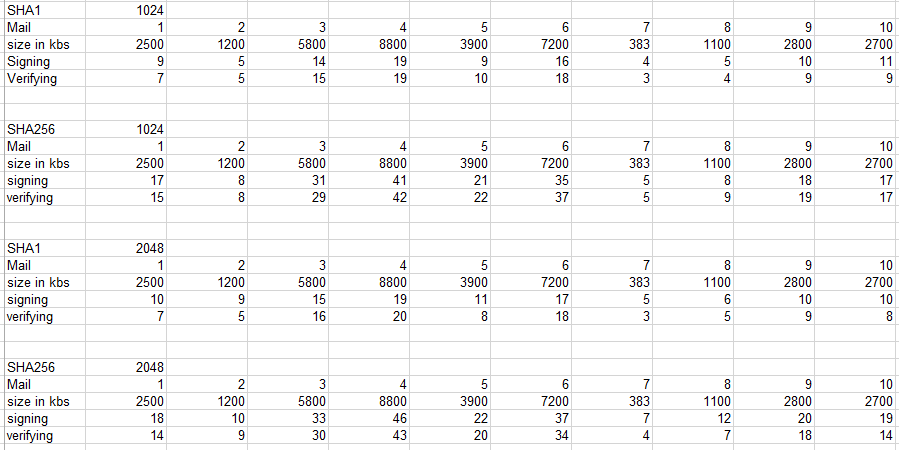
## 2.1 Brief description of how DKIM works.

* Domain Keys Identified Mail is an email validation system used to detect email spoofing and email related malicious activities. It basically detects if an email is from a domain authorized by a trusted domain administrator.
* Here validation is achieved through the use of digital signatures appended to the message, which can be used to validate, by the receiver using signer’s public key from the Domain name Server.

### Components:

1. ADMD (Administrative management domain). Signing is done with this authority.
2. The signature. It is a validation method. It can be performed by any agent in ADMD, within the message path.
3. MUA (Mail user Agent). This is one candidate for signature validation.
4. MSA (Mail submission Agent). Enforces and polices the domain standards in the message.
5. MTA (Mail Transfer Agent). It is responsible for the routing and transfer of the message to the MDA.
6. MDA (Mail Delivery Agent). It is responsible for the successful delivery of the message.

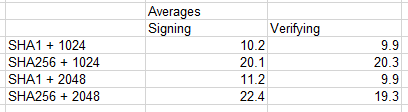
2.2. Graph plots and Source Code



The above table demonstrates the average execution time for SHA1 + 1024, SHA256 + 1024, SHA1 + 2048, and SHA256 + 2048 respectively.

2.3 Explain which of the combination of core DKIM algorithms provides the best performance using email message size as the criteria.

From the above graphs and table we can see that SHA1 has a lower execution time as compared to SHA256, and for a smaller key bit size which is 1024 here, we can see that as compared to the bigger key size 2048 the execution times are lower. Hence we can see from the averages that SHA1 + 1024 has the best performance as compared to the other 3 algorithms.



# 2.4.1 Problems with S/MIME and PGP

* It does not provide ubiquitous accessibility, as private key is accessible only to the user and not the webmail server.
* S/MIME is only suitable for end to end delivery of the message. If the message is inspected in the middle then we can face a problem.
* S/MIME requires several certificates, so this may add significant overhead in terms of performance and cost, as the CAs may be very expensive.
* PGP or S/MIME always modify the messages, they provide their packaging as MIME body parts. DKIM does not, thus being completely invisible.
* S/MIME and PGP need more costly validation and verification mechanisms as they are more of, longer term oriented.

# 2.4.2 How is DKIM different from other algorithms?

* Unlike PGP and S/MIME, DKIM does not need a lot of certificates, thus reducing the certificate authority expense and performance overhead.
* DKIM is completely invisible.
* DKIM signature validation and verification is done by domain name administrator rather than the individual users.
* No public key-private key dependency exists here. So no PKE and PKI systems are needed for DKIM.

# 2.4.3 Broad categories of Domain Validations

1. Validation using IP address.
2. Validation using digital signatures.

DKIM falls under the category of validation using digital signature as it involves signing email messages for the purpose of source repudiation prevention and validation and verification.

# 2.4.4 What does DKIM do for the signer and for the receiver?

**For Signer**: Signer provides different levels of trust in the message sent. Different parts of the message are signed in different levels of “signature-trust”. Signer uses different sub domains names for different message streams, so that receiver can make suitable judgment about the mail contents.

**For Receiver**: Provides validation of the message and the sender. Receiver can differentiate between different levels of message streams based on the signatures on them.

DKIM gives receiver an identifier for a message which he can use to differentiate between a good sender and a bad sender who may have a bad reputation of masquerading.

* + 1. Does DKIM signature signify that all the fields in the header information are not forged?

No, the DKIM signature does not help us conclude that all the fields in the header information are not forged. When a receiver receives an email message, it obtains the public key used to encrypt the DKIM signature in the header from the DNS records. The hash value (DKIM signature) obtained after decryption is then compared by the receiver to the one that the receiver itself calculates using the same algorithm. If they match, it simply means that the message contents have not been altered and that the message has been sent by the expected party without any forgery or interception.

References:

<http://www.dkim.org/info/dkim-faq.html>

<http://en.wikipedia.org/wiki/DomainKeys_Identified_Mail>

<http://en.wikipedia.org/wiki/S/MIME#Obstacles_to_deploying_S.2FMIME_in_practice>

<http://en.wikipedia.org/wiki/Pretty_Good_Privacy>

<http://www.securityweek.com/lessons-recent-dkim-mail-verification-vulnerability>