SMART INDIA HACKATHON '17

Email Spoofing Detection Prevention & Alerting

"Winning Team" Under Ministry of External Affairs At

Smart India Hackathon 2017 Grand Finale

Techno Freakz

Team Members:

P.SEENIVASAN, R.SUJITHA, N.SATHISH KUMAR, M.R.REVATHY,C.VIKRAM KUMAR, M.NITHYA

Team Mentors:

Dr. A. JAMEER BASHA, ARUN THUNDYILL SASEENDRAN

Aim:

To develop a product that can detect the spoofed mails and block the mails receiving server end before entering into the receiver's mailbox.

Summary:

Email Spoofing is a well-known threat to the information security. In the recent past several email spoofing attacks have been executed resulting huge losses in terms of information security and monetary loss. This proposal provides a comprehensive solution to provide immunity against email spoofing attacks at the receiving mail server side. The solution provided in this proposal, primarily makes use of the MIME headers available in the email along with industry standard algorithms to detect spoofed emails, stop them from reaching the inbox of the recipient and to send alerts to authorized persons as and when spoofed emails are received. The proposal provides three custom algorithms making use of MIME data and also makes use of Sender Policy Framework (SPF) and Domain Keys Identified Mail (DKIM) to provide a comprehensive solution in a fail fast manner. The results are demonstrated in a live system using Apache James Mail Server. The solution provided can easily be adopted to any messaging and communications server containing similar architecture with minimal changes.

Introduction:

E-mail spoofing is a term used to describe (usually fraudulent) email activity in which the sender address and other parts of the email header are altered to appear as though the e-mail originated from a different source. Email spoofing finds a wide variety of victims who may be attacked for a multiple reason. Initially when a mail is sent from a server, it is routed by the domain name server to the receiver's mail server environment. The proposed algorithm is placed as a pluggable product inside the receiver mail server environment, which detects the spoofed mails using the raw MIME format associated with every mail. The MIME format consists of variety of details that are related to the specific mail. The proposed algorithm retrieves specific fields from the MIME format and uses them in identifying the spoofed mails that differs from the legitimate mail. Once the spoofed mails are detected using the set of algorithms they are prevented from entering into the receiver's mail box. This proposed solution not only prevents the mails from entering into the mail box but also adds an additional functionality of alerting to configurable authorized person regarding the spoofed mails.

Objective:

- The main objective of this proposed system is to detect the spoofed mails at the receiving server side itself.
- To stop the spoofed mails from entering into the receiver's mailbox.
- To alert the authorized persons regarding the spoofed attack with a detailed report.
- To prevent data loss due to fraudulent emails.
- To enhance the security in the receiver's mail server.

Status:

The current status of our project is that our team has developed a "Mailet (Apache Mailet Compatible)" using JAVA 1.8 programming language. Our team developed the prototype as an "Mailet" since it is pluggable module accepted by many standard Mail servers and is also supported by Apache James Mail server which was used for the demo. This developed mailet is plugged into Apache James 3.0 mail server which has been used as test server for testing our test cases regarding the detection of spoofed mails. The prototype is fully functional and detect and prevent spoofed emails from reaching the mail box of the receiver. In addition, it can send detailed alert reports to the configured administrator.

To suit the needs of the ministry, this prototype can be developed as a "Milter (Oracle Communications and Messaging Server compatible filter plugin)" which can be configured with Oracle Communications and Messaging Server. The product can easily be adapted as a plugin in the client side (Thunderbird, Outlook etc.,) since the algorithms are implemented in a platform independent manner. The milter implementation can be done with minimal efforts as the mailet implemented in the prototype is architecturally similar to a milter. A milter satisfying all the objective outlined above is no available as an indigenous open-source/proprietary product. It was exclusively researched and developed by the "Techno freakz" team to satisfy all the requirements put forward by the ministry.

Novelty:

- Smart India Mailet is an indigenous product developed exclusively to satisfy all the requirements put forwards by the Ministry of External Affairs in terms of detection and prevention of spoofed emails.
- There is no similar product existing in the market.
- This is the first indigenous spoofed email detection product available in the market and whose source code is neither open-source nor proprietary. Hence utmost security is guaranteed.
- The Smart India Mailet is designed in such a way that it can easily accommodate new and diverse requirements. The architecture is suitable for majority of the standard enterprise mail servers.
- These algorithms used are independent of the implementation platform and hence the same product can easily be adapted to work along with Oracle Messaging and Communications server or as a plugin in the client side (Thunderbird, Outlook etc.,)

Work plan:

The plan is to develop Smart India Mailer as a plugin/add-on for 3rd party cross platform applications like Mozilla Thunderbird, MS Outlook so that it can be used without making changes to the server. The plugin can be adapted to be used by any of the cross platform email applications.

As enhancements to the existing product, the proposal is to identify the geographical location from where the mail has been generated and also in identifying the server from which the mail is composed when a spoofed email is detected so that it will greatly aid the cyber forensics department. By identifying the server and geometric location of the fraudster, the concerned department can take necessary actions to prevent future spoofing attacks.

Open Source Technology Stack to be used:

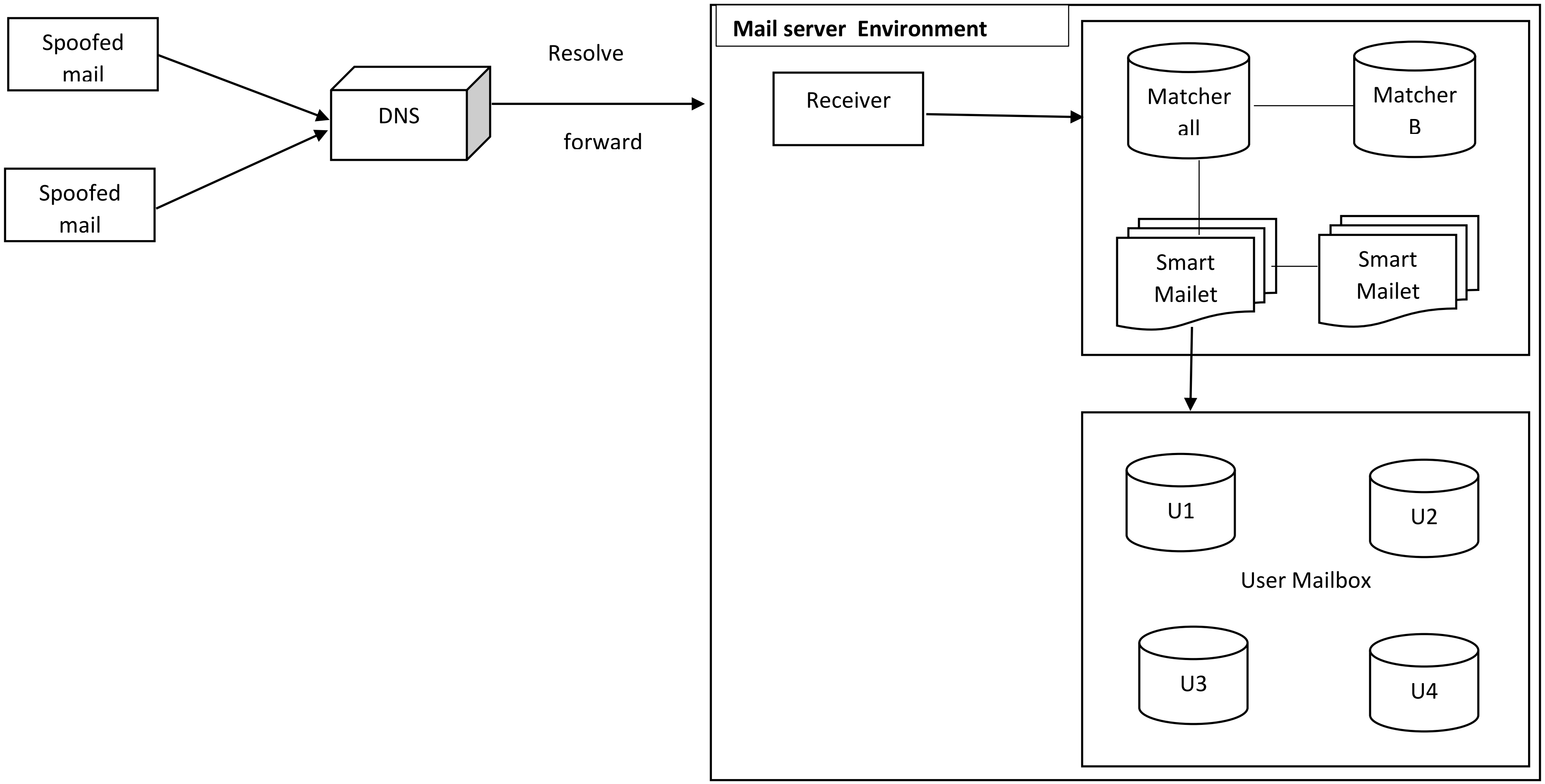
- Apache James 3.0 Server for testing.
- Programming Language – JAVA 1.8
- Apache Mailet 2.0
- Mozilla Thunderbird SDK (For Dev and Test of the Thunderbird Plugin)

- Visual Studio 15 (For Dev and Test of the Outlook Plugin)

Any technologies that you used during SIH2017 that you wish to change/ alter? Why?

In SIH 2017, our team developed the prototype as an mailet which can be used on servers that support Apache Mailet like the Apache James 3.0 server. On discussion with the Ministry of External Affairs, it was understood that a server side solution is not easily feasible as the servers are not maintained by the Ministry. Hence a client side solution with the same working was required. So, we plan to implement the product as a plugin/add-on, which can be used as per the user requirements in cross-platform applications like Mozilla Thunderbird or, MS Outlook.

Architecture Block Schematic



Time line and responsibility of participating team

Particulars			Time (Months)																							
#	Tasks	Team Member's Name	Month 1				Month 2				Month 3				Month 4				Month 5				Month 6			
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	Existing System Study	MEN1, MEN2, TL, M1	🟦	🟦																						
2	Infrastructure Requirements and Inventory	M2,M3,M4,M5	🟦	🟦																						
3	System Study and Client Interaction	MEN1, MEN2, TL, M1			🟦																					
4	Development Bed Setup	M2,M3,M4,M5			🟦	🟦																				
5	System Architecture Design and Documentation	TL, M1, M2				🟩	🟩	🟩	🟩																	
6	Test Bed Setup, Build and Dev Process	M3,M4,M5					🟦	🟦	🟦																	
7	Architecture Review and Sign Off	MEN1,MEN2								🟦																
8	Thunderbird Client Design-High Level (Class and Components)	TL, M1, M2									🟧	🟧														
9	Thunderbird Client Test Scenarios Design	M3,M4,M5									🟨	🟨														
10	Thunderbird Client Design-Low Level(Sequence Diagrams and Use Cases)	TL, M1, M2											🟧	🟧												
11	Thunderbird Client Test Scenarios Cases	M3,M4,M5											🟨	🟨												
12	Design Review and Sign Off	MEN1, MEN2													🟦											
13	Thunderbird Client Implementation	TL,M1,M2,M3													⬛	⬛	⬛	⬛	⬛							
14	Thunderbird Client Testing and Validation	M4,M5														🟫	🟫	🟫	🟫	🟫						
15	Review and Demo to Client	MEN1, MEN2, TL																		🟦						

16	Bug Fixing and Additional Feasible Requirements Implementation	TL,M1,M2,M3																			⬛	⬛	⬛	⬛		
17	Bug Testing and New Feature Testing	M4,M5																			🟫	🟫	🟫	🟫		
18	Final Review and Demo	MEN1, MEN2, TL																							🟦	
19	Smoke Testing, Patches and Product Documentation	M1,M2,M3,M4,M5																							🟫	
20	Sign-off	ALL																								🟩

Legends

Definition	Symbol/Color
P.SEENIVASAN	TL
R.SUJITHA	M1
N.SATHISH KUMAR	M2
M.R.REVATHY	M3
C.VIKRAM KUMAR	M4
M.NITHYA	M5
Dr. A. JAMEER BASHA	MEN1
ARUN THUNDYILL SASEENDRAN	MEN2
🟦	Analysis
🟦	Infra Setup
🟩	Architecture
🟦	Review
🟧	Design
🟨	Test Preparation
⬛	Development
🟫	Testing
🟩	Sign-off

Comprehensive budget

Budget requirements (total as well as individual institutions/laboratory along with monthly break-up covering manpower, travel, contingencies, overheads, others (if any) and equipment for the 6 months' project duration)

(***Rs. in lacs***)

Head of Expenditure	1st Month	2nd Month	3rd Month	4th Month	5th Month	6th Month	**Total**
	Recurring						
Travel (Client Meetings & Review)	1	0.1	0.1	0.1	0.1	0.8	2.2
Contingencies	0.15	0.15	0.15	0.15	0.15	0.15	0.9
Compute Resources (http://s3.amazonaws.com/calculator/index.html#r=IAD&s=EC2&key=calc-69E91777-4BE6-4539-BCD5-C2A50A70E8F7)	0.3	0.3	0.3	0.3	0.3	0.3	1.8
Other research expenditure - Stipend	0.72	0.72	0.72	0.72	0.72	0.72	4.32
	Non-Recurring						
Equipment and accessories (Network, Peripherals)	2						2
Licensing cost (Domains Registrations)	0.2						0.2
Total	4.37	1.27	1.27	1.27	1.27	1.97	11.42