sdnog

# build your own email server

**Sdnog activities - 15th Feb 2020**

**By Manhal Mohammed**

# topics:

All what you need to know to build your own email server ;)

# What is email?

- The term “email” stands for “electronic mail”. The electronic mail is introduced first in the 1960s, however it became available in the current structure in the 1970s.
- *Simple Mail Transport Protocol* (SMTP) is a networking protocol that is responsible for transporting email messages from one email server to another.
- It establishes a reliable connection between the source and the destination email server for message handover.
- This protocol runs on top of the IP protocol and uses well-known TCP port 25 “465 ??! And 587 also 2525”  for its operation.

# Common Terms

Mail User Agent (MUA)

Mail Submission Agent (MSA)

Mail Transport Agent (MTA)

Mail Delivery Agent (MDA).

- **Mail User Agent (MUA):**

email client program used to compose messages, and to submit them to an outgoing MTA.

* On the receiving side, an MUA pulls the message into the inbox of the user and allows them to read. An MUA uses either the POP or IMAP protocol for mail retrieval. --→ *thunderbird*, and *Outlook*

- **Mail Submission Agent (MSA):**

A *Mail Submission Agent* is responsible for accepting new mail messages from an MUA.

- **Mail Transport Agent (MTA):**

A *Mail Transport Agent* is responsible for transporting a message from a sending mail server, and another MTA is responsible for accepting the message at a receiving mail server, and they both use SMTP -→ send mail , postfix

- **Mail Delivery Agent (MDA):**

A *Mail Delivery Agent* is responsible for delivering an incoming message to a local mail spool location for storage.

# Protocols:

- **Post Office Protocol (POP)** used by an MUA and it is responsible for downloading user mail messages from the mail server to their local inboxes and optionally, delete them on the server to free up space port 110 , 995
- **Internet Message Access Protocol (IMAP)**

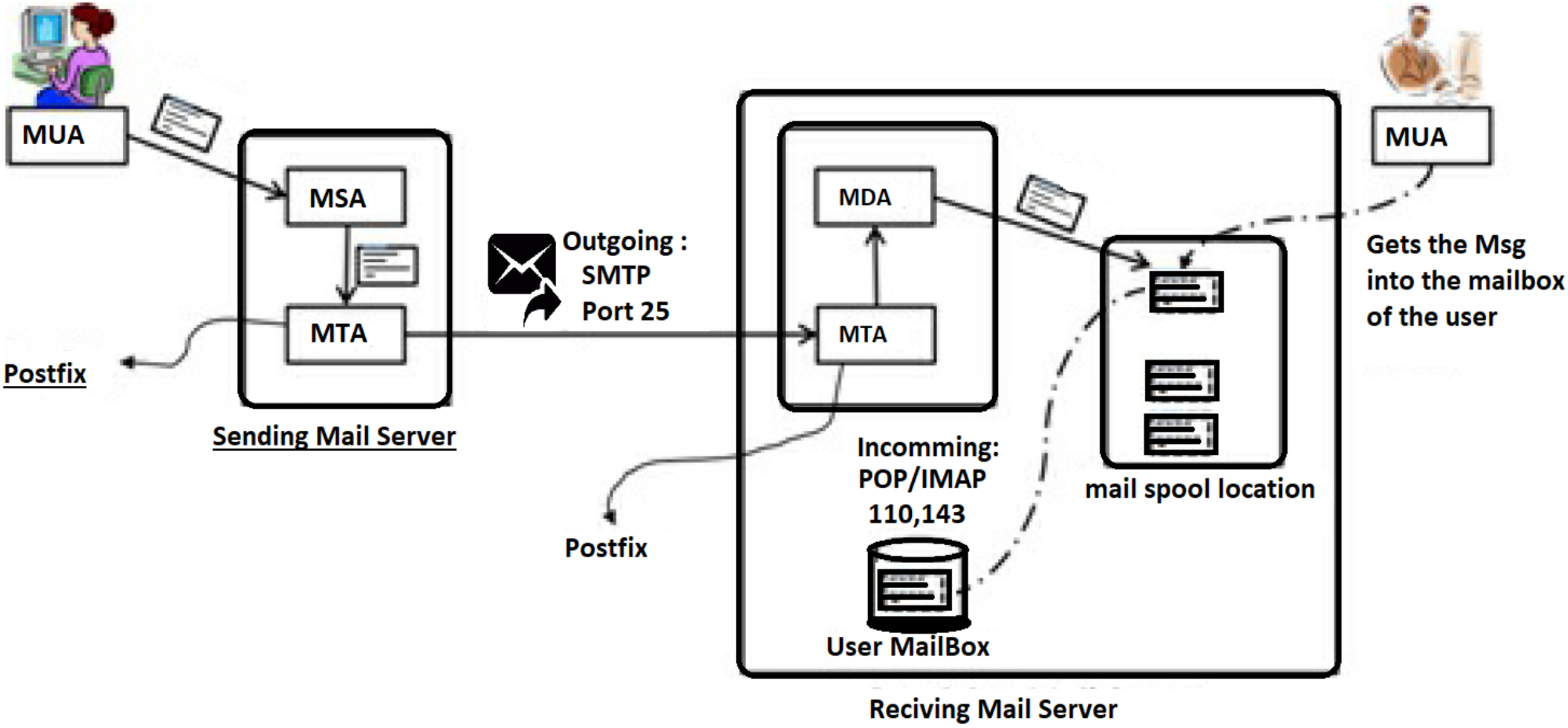
is used by an MUA and it is responsible for downloading user mail messages from the mail server to their local inboxes. Port 143 , 993

# Smart Host (Relay)

A *smart host* is an MTA that is configured with the intent to deliver email messages on behalf of other systems. The other systems may have non-persistent connection to the Internet or lack the capacity to perform this function efficiently.

# So How the Email System really Works ??

# MUA → MSA → MTA → Network / Internet → MTA → MDA → MUA



# How the Email System Works

- an email client (MUA) to compose an email message with all the relevant information
- a program (MSA) that takes this email message and submits it to a temporary location (mail queue) for further processing
- another program (MTA) for transporting that email message over the network or the Internet to the destination system (another MTA) using the SMTP protocol
- On the receiving side, the MTA receives that message and invokes an MDA
- MDA forwards the message to a temporary mail spool location and holds it there until an MUA picks it up and saves it in the mailbox of the user via either POP3 or IMAP protocol.

# Lets dig more ☺

# What is POSTFIX

- **Postfix** is a free and open-source mail transfer agent (MTA) that routes and delivers electronic mail, intended as an alternative to the widely used Sendmail MTA.
- Postfix is released under the IBM Public License 1.0 which is a free software licence.
- Originally written in 1997 by Wietse Venema at the IBM Thomas J. Watson Research Centre and first released in December 1998, Postfix continues as of 2014 to be actively developed by its creator and other contributors. The software is also known by its former names **VMailer** and **IBM Secure Mailer**.
- In January 2013 in a study performed by E-Soft, Inc. found that approximately 25% of the publicly reachable mail-servers on the Internet ran Postfix.

# Postfix

- Works on UNIX-like systems including AIX, BSD, HP-UX, Linux, MacOS X, Solaris, and more.
- It is the default MTA for the OS X, NetBSD[3] and Ubuntu operating systems
- Used by: AOL, Apple Server, Stanford University, United States Navy, NASA, Rackspace, many ISPs
- Able to process thousands …

# Some Key Features

- SASL authentication Simple Auth Security Layer
- Mail forwarding or delivery
- "Virtual" domains with distinct address-namespaces
- A large number of database lookup mechanisms including Berkeley DB, CDB, OpenLDAP LMDB, Memcached, LDAP and multiple SQL database implementations Extended
- Deep content inspection before or after a message is accepted into the mail queue;
- Mail authentication with DKIM, SPF, or other protocols;
- SMTP-level access policies such as greylisting or rate control.

# Postfix on RedHat & Debian , Frebsd

- **Redhat**
  - Installed via: **$sudo yum install postfix**
  - Directories: **/etc/postfix**
- **Debain**
  - Installed via: **$sudo apt-get install postfix**
  - Directories: **/etc/postfix**
- **FreeBSD**
  - Installed via: **$sudo pkg install postfix**
  - **Directories: /etc/postfix** or **/usr/local/etc/postfix**
- **Configuration files**
  - main.cf - stores site specific Postfix configuration parameters while
  - master.cf – defines daemon processes

# master.cf

- defines how a client program connects to a service, and what daemon program runs when a service is requested.
- The Postfix master daemon launches all of the other Postfix services as they are needed. The various services, and how they are run, are specified in the master.cf file.
- The SMTP service is defined in this file as well as third party apps like an SPF program or a DKIM Program

# main.cf

- specifies a very small subset of all the parameters that control the operation of the Postfix mail system you will have to set up a minimal number of configuration parameters.
- Postfix configuration parameters resemble shell variables
  - parameter = value
  - other_parameter = $parameter
- Postfix uses database files for access control, address rewriting and other purposes

# main.cf Key Settings

- myorigin = $myhostname

specifies the domain that appears in mail that is posted on this machine. Defaults to the value of the machine's hostname

- mydestination = $myhostname, localhost

specifies what domains this machine will deliver locally

- if your machine is a mail server for its entire domain, you must list $mydomain as well in this setting
- The mydomain parameter specifies the parent domain of $myhostname. By default, it is derived from $myhostname by stripping off the first part (unless if the result would be a top-level domain)

# Main.conf “/etc/postfix”

| Directive | | Description |
|---|---|---|
| queue_directory | /var/spool/postfix | Location of Postfix queue. |
| command_directory | /usr/sbin | Location of the Postfix commands. |
| daemon_directory | /usr/libexec/postfix | Location of the Postfix daemons. |
| mail_owner | postfix | Owner name for Postfix queue and daemons. |
| myhostname | host2.example.com | FQDN of the Postfix server. |
| mydomain | example.com | Domain name of the Postfix server. |
| myorigin | $my hostname | Host or domain name the outgoing mail appears to have originated from. |
| inet_interfaces | localhost | Network interfaces to be used for incoming mail. |
| mydestination | $my hostname, localhost.$my domain, localhost | Domains the Postfix server accepts mail from. |
| Mynetworks | 192.168.2.0/24, 127.0.0.0/8 | IP addresses of trusted networks. |
| relayhost | $my domain | Hostname of another mail server (smart host) to forward mail to. This mail server will act as an outgoing mail gateway. |
| alias_maps | hash:/etc/aliases | Aliases database used by local delivery agent. |
| aliases_database | hash:/etc/aliases | Aliases database generated with the newaliases command. |
| mail_spool_directory | /var/spool/mail | Location for storing user mailboxes. |

# Relaying Mail – From

- Postfix will forward mail from clients in authorized network blocks to any destination
- Authorized networks are defined with the mynetworks configuration parameter
- The default is to authorize all clients in the IP subnetworks that the local machine is attached to.
- By default, Postfix will NOT be an open relay ie it will not forward from IPs outside your network to the Internet
- mynetworks_style = subnet
- mynetworks = 127.0.0.0/8 168.100.189.2/32

# Relaying mail - to

- By default, Postfix will forward mail from strangers (clients outside authorized networks) to authorized remote destinations only.
- Authorized remote destinations are defined with the **relay_domains** configuration parameter.
- The default is to authorize all domains (and subdomains) of the domains listed with the **mydestination** parameter.
- This means that by default, your Postfix mail server will accept mail from anyone to recipients to the local Postfix server

# Outbound emails

- By default, Postfix tries to deliver mail directly to the Internet.
- Depending on your local conditions this may not be possible or desirable
- For example, your system may be behind a firewall, or it may be connected via a provider who does not allow direct mail to the Internet.
- In those cases you need to configure Postfix to deliver mail indirectly via a relay host.
- relayhost = [mail.isp.tld]
- Note that the [] disables MX lookups so is necessary

# Reporting problems

- You should set up a postmaster alias in the aliases table that directs mail to a real person
- The postmaster address is required to exist, so that people can report mail delivery problems.
- While you're updating the aliases(5) table, be sure to direct mail for the super-user to a human person too.
- /etc/aliases:

  postmaster:

  sdnog root: sdnog

- After editing the aliases file, run the command $sudo newaliases

# Logging

- Postfix will log all messages to /var/log/maillog
- Done using the syslogd daemon
- All transactions of messages coming in being sent out of the server will be logged
- Logs will contain details like hostnames, recipients, time and date, and whether the email was queued or dropped

Log , check log , read logs all the time
logs are our friends in need indeed
☺
A screen you should love :
# tail -f /var/log/maillog

# Dovecote

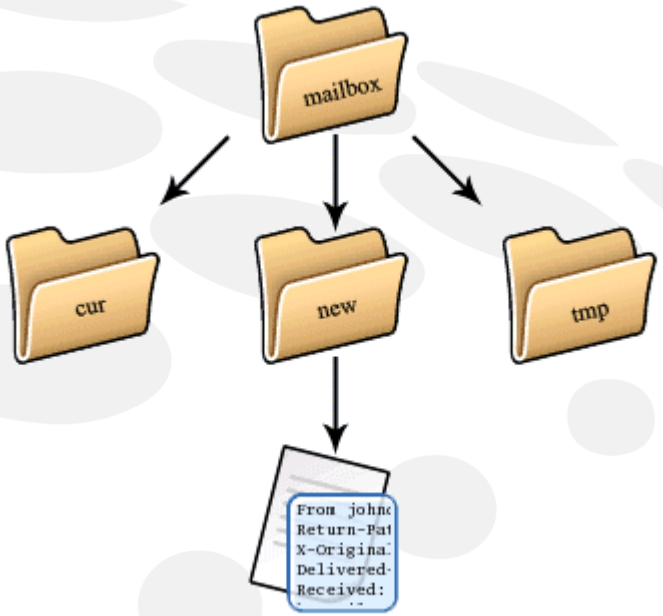
- High-performance POP and IMAP server Developed by Timo Sirainen
- Supports multiple user databases (MySQL, Postgresql)
- Supports both mbox and maildir formats Works on most Linux and UNIX distros
- Graceful around failures (index repair for example)
- Designed with Security in mind – support for Authentication Mechanism and SSL/TLS
- Install:

  #yum install dovecot
- Note that mail servers like EXIM and Postfix are simply mail engines. You will need an IMAP server to allow users to connect and check email

# Mail Storage Formats

- Mailbox Format (Mbox)
- Defined in RFC 4155
- All messages in an Mbox mailbox are concatenated and stored as a plain text in a single file.
- Mails are stored in RFC822 format with a blank space separating each message (2 spaces as each message has one space) and “From” determining start of next message.
- Mbox has a distinct disadvantage in cases of large mailbox (a single large file) requires more resources to read/open and can be slow depending on the server’s load.

# Maildir Storage Format




- Mail Directory Format (Maildir)
- Each message is stored in a separate file with a unique name and each folder in a directory
- Maildir++ provides extension to the Maildir
- specification providing support for subfolders and quotas.
- Maildir directory has 3folders temp, new and
- current

# How Maildir Works

- The mail delivery agent stores all new emails to the mailbox in the tmp directory with a unique
- filename. (unique = time + hostname+ random generated number)
- The MDA creates a hard link to the file in tmp/ unique to new/unique
- The Mail User Agent will check for new emails in new folder and move them to current folder
- The MUA modifies the filename to add a colon (:), a ‘2’ and various flags to represent message status i.e read, replied, forwarded, deleted, etc

# Key settings in Dovecot Config

- Edit the file /usr/local/etc/dovecot.conf

  protocols = imap imaps pop3

  log_path = /var/log/dovecot

  ssl_cert_file = /usr/local/etc/apache22/server.crt

  ssl_key_file = /usr/local/etc/apache22/server.key
- Disable plaintext authentication by finding the line below

  #disable _plaintext_auth = no
- Uncomment the line and Set the value to yes as below

  disable_plaintext_auth = yes
- Note: unencrypted connections can still be made from localhost!

# Webmail

# What is Webmail?

- Webmail provides a web-based (HTTP) Mail User Agent (Front end) to access emails
- This makes webmail available anywhere which is practical for most users.
- Use the default http port 80 but can ran on other user defined ports.
- Webmail systems will access the mail server using IMAP4(s), POP3(s). Some read the files directly from the mailbox stored in Maildir format e.g Sqwebmail.


- Started in 1998 by two brothers Luke and Nathan Ehresman
- Named after squirrels for their agility
- www.squirrelmail.org
- It supports IMAP and SMTP protocols and can be setup to support a wide range of MTA and implementations
- Its written in PHP

# Why Squirrelmail

- Squirrelmail is stable and scales well in most environments
- Continues to have features support in plugins including password change and server-side filters
- Its fast with options header caching and supports server side indexing – works well with Dovecot and Cyrus IMAP
- It has a light interface due to php
- Additional resources would be required to make it scale for large scale implementations

This is going to be ONLY for one hour

# Lab setup

# Lab preparation

- Please install VBox in your laptop with its extension pack ☺ if you are familiar with VMware feel free to use it
- Download ssh client “putty/Mobaexterm”
- Please create your centos7 minimal VM with two interfaces
  - **Enp0s3: NAT**
  - **Enp0s8: hostonly-adapter**
  - **Enp0s9: bridge connected to your wifi adapter**
- All sources can be found at the FTP server
- If you are done , help others ☺

# VM setup

- Connect your pc to sdnog SSID
- Configure your interface to be bridge to access our LAB network
- Root password must be **sdnog@123**
- Create user **sdnog** with password **sdnog@123**
- Get ip from the DHCP by using the command :

```
# dhclient
```

- Make sure you can ping your neighbors

**nmcli con add type ethernet con-name sdnog-if ifname enp0s9 ip4 192.168.70.X/24**

**Or use**

**nmtui tool ☺**

# Email Security and Best Practices

Referenced to Kevin Chege , **Afnog 2019**

# Why your email setup is critical

- Billions of SPAM emails are generated every day
- The tips here can help you to reduced the chances of you receiving SPAM email or inadvertently being the source of SPAM emails
- Because email is so efficient, its now used to send malware, ransomware, worms etc.

For example: WannaCrypt!

# Security

- Run secure pages from the mail server and secure SMTP to clients
  - Secure Webmail – port 443
  - Secure SMTP – port 465/587
- Force clients to use secure IMAP or Secure POP
  - Secure POP – port 995
  - Secure IMAP – port 993
- Require authentication on your mail server before a mail enters the queue from a sending client aka SMTP AUTH
- Lock down your box and block all unnecessary ports

# User Training is important

- Innocent actions by your users may trigger anti-spam rules
- Adding tens of email addresses in the “TO” field when composing email
- Adding Subject with ALL CAPS IN THE SUBJECT
- Attaching files with different extensions
- “ImportantContract.PDF.Docx”
- Opening Phishing emails that contain trick subject lines like “Your inbox is full” or “Attention your email is compromised”

# SPF – Sender Policy Framework

- SPF – Sender Policy Framework
- SPF allows administrators to specify whichhosts are allowed to send mail from a given domain by creating a specific SPF record (or TXT record) in the Domain Name System (DNS).
- @ IN TXT “v=spf1 include:gmail.com ip4:1.2.3.4 mx -all”
- The above will only allow mail from IP 1.2.3.4 and any server in the domain with an MX record
- If not sure use a generation tool online

http://www.mtgsy.net/dns/spfwizard.php

# Domain Keys Identified Mail (DKIM)

- DKIM (DomainKeys Identified Mail) is an authentication mechanism to help protect both email receivers and email senders from forged and phishing email.
- It is intended to prevent forged sender addresses in emails, a technique often used in phishing and email spam.
- DKIM allows the receiver to check that an email claimed to come from a specific domain was indeed authorized by the owner of that domain which is done using cryptographic authentication.
- Verification is carried out using the signer's public key published in the DNS. A valid signature guarantees that some parts of the email (possibly including attachments) have not been modified since the signature was affixed

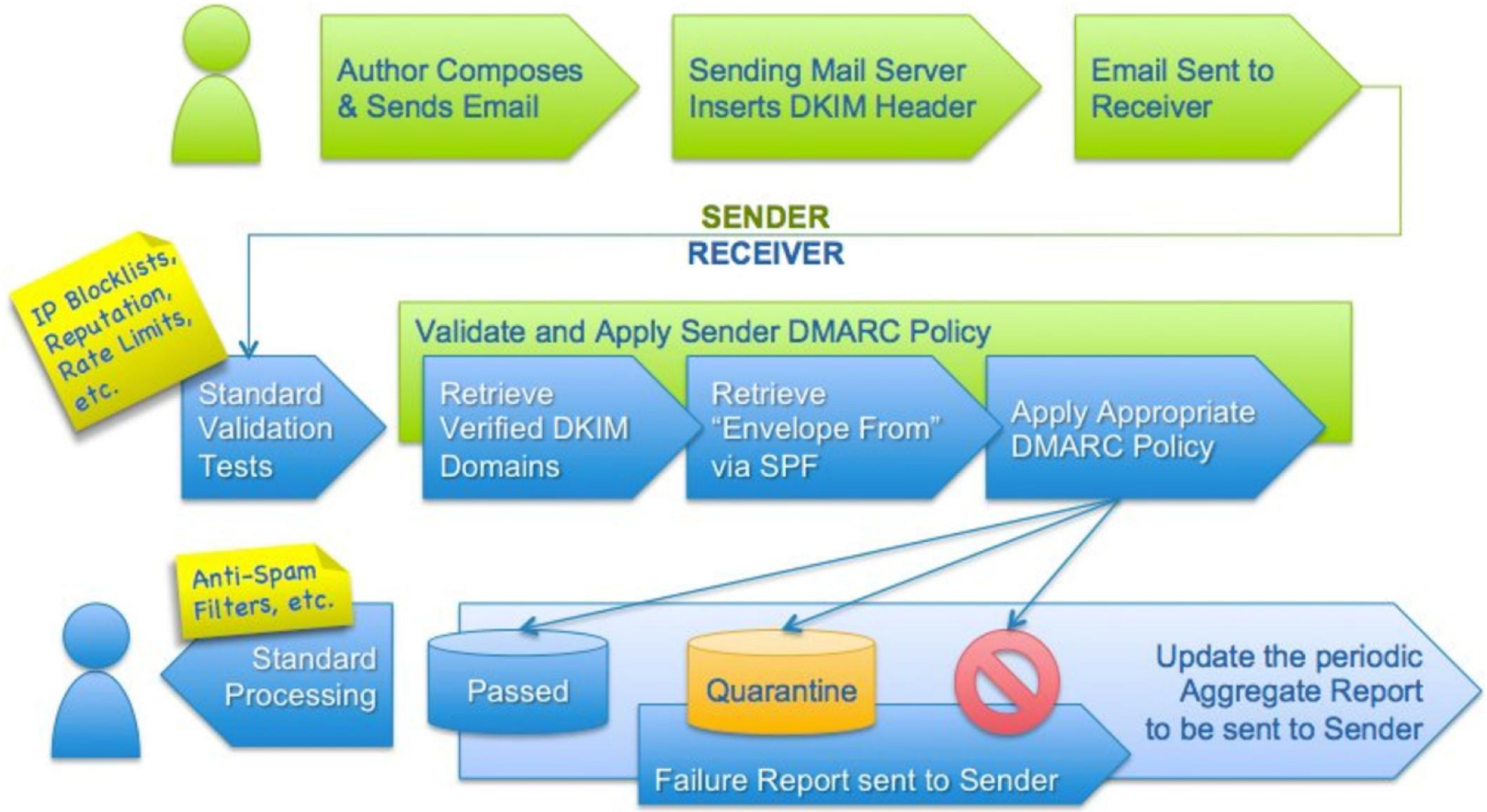
# DMARC -

- which stands for “Domain-based Message Authentication, Reporting & Conformance
- It builds on the widely deployed SPF and DKIM protocols, adding linkage to the author (“From:”) domain name, published policies for recipient handling of authentication failures
- Another IETF standard designed to combat growing spam
- More at http://dmarc.org

# Why is DMARC important

- Allows Domain owners to:
  - Signal that they are using email authentication (SPF, DKIM)
  - Provide an email address to gather feedback about messages using their domain – legitimate or not
  - A policy to apply to messages that fail authentication (report, quarantine, reject)
- Allow Email receivers to:
  - Be certain a given sending domain is using email authentication
  - Consistently evaluate SPF and DKIM along with what the end user sees in their inbox
  - Determine the domain owner's preference (report, quarantine or reject) for messages that do not pass authentication checks
  - Provide the domain owner with feedback about messages using their domain

# DMARC FlowChart



# SPF, DKIM and DMARC

- All published in DNS!
- SPF sample: $dig TXT facebook.com

  “v=spf1 redirect=_spf.facebook.com”
- DKIM sample: $dig google._domainkey.protodave.com TXT

google._domainkey.protodave.com. 3600 IN TXT "v=DKIM1\; k=rsa\;
p=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAhArxYH88+A76Gk7/8ENefN5RhMFhoYJp8T3KLPYY
pejDI45PKWTO+2r8ZJZOtuk7tsG07bmJyU8PFvU48Lf1xtb4WcFxKKjd7N5MF6JcHD51Xb8XDAJA2ldqxH4hBbw9

dRjsT7WBFXbp2x6MSWxgi9f1w+7Z2IFG+AtUjrf8/9N3gLieaZKZT1SEhR8TnhfOm"

"FG0LfMyS0YtfHKrkUkBCEmWBPisB2CcZBShKr6/T8/UB/oZF8XMRd0NOsru9MGx9Yp89jIYS5YRuvbA0/TLgOOi
qrSU5Ms1egMwfFyy4BMDUKayZzF6BxNPc/+UoFrYHKRZpyD/kEd4FXNEddlksQIDAQAB”

- DMARC sample: dig TXT _dmarc.google.com

  "v=DMARC1;p=reject;pct=100;rua=mailto:postmaster@dmarcdomain.com”

# DANE – Encrypting email transfer from sender to recipient

- DNS-based Authentication of Named Entities
- Described in RFC 6698 and proposed as way to authenticated TLS certificates to be bound to DNS using DNSSEC
- Having a DANE Record indicates that a sender of an email must use encryption (TLS) to transmit the email from the sending server to the recipient email
- Using DANE therefore will ensure that the email sent to you was transmitted over TLS (encrypted) and so its much more difficult for an eaves dropper to read your email
- Without DANE, email uses opportunistic encryption to secure SMTP – ie it will be used if available

# Reverse Records

- Have reverse records (PTR) for your mail server so that it is resolvable from the IP
- Mandatory by most servers these days
- Used to verify authenticity of the sending mail server
- The IP Address must resolve back to the mail server name
- You can have multiple reverse records
- You can have an SPF record that states that any IP that has a reverse record can send email from your domain

  IN TXT “v=spf1 ptr:domain.co.tz ip4:1.2.3.4 mx -all”

# Summary of DKIM, DMARC, SPF, ARC, DANE, PTR…

- As a receiver of my email, you can accept it because:
  - I have told you which servers I control – SPF Record
  - My email server has signed the email – DKIM
  - My server's signature can be verified using DNS servers I have configured – DMARC or ARC
  - My email client signed the email with a PGP key
  - My servers have verifiable PTR records
- As a recipient of your email, I can guarantee you that your email was sent to me over a secure channel because
  - DANE – my server only accepts securely sent email and used DNSSEC infrastructure to validate my authenticity
  - MTA-STS – my server only accepts securely sent email and used a Certificate Authority to validate my authenticity

# This is too much for me! Outsource my email?

- But where is your email stored?
- Who has access to it?
- Why is it free? Is it really free??

Now you know
just DO IT ☺

# Use Anti Spam and Anti Virus software

- Will reduce overall spam and email received
- You can also have a mail “firewall” or gateway aka Mail Filter to stop spam before it reaches your server
- Some softwares are:
  - SpamAssassin (AntiSpam) – renowned antivirus
  - Rspamd – powerful antispam Milter service
  - ClamAV (AntiVirus) – renowned antivirus
  - MailScanner and Amavisd (rely on the above)
- When setup try a penetration testing site to see how well your server can protect you from SPAM and Viruses

# Grey Listing

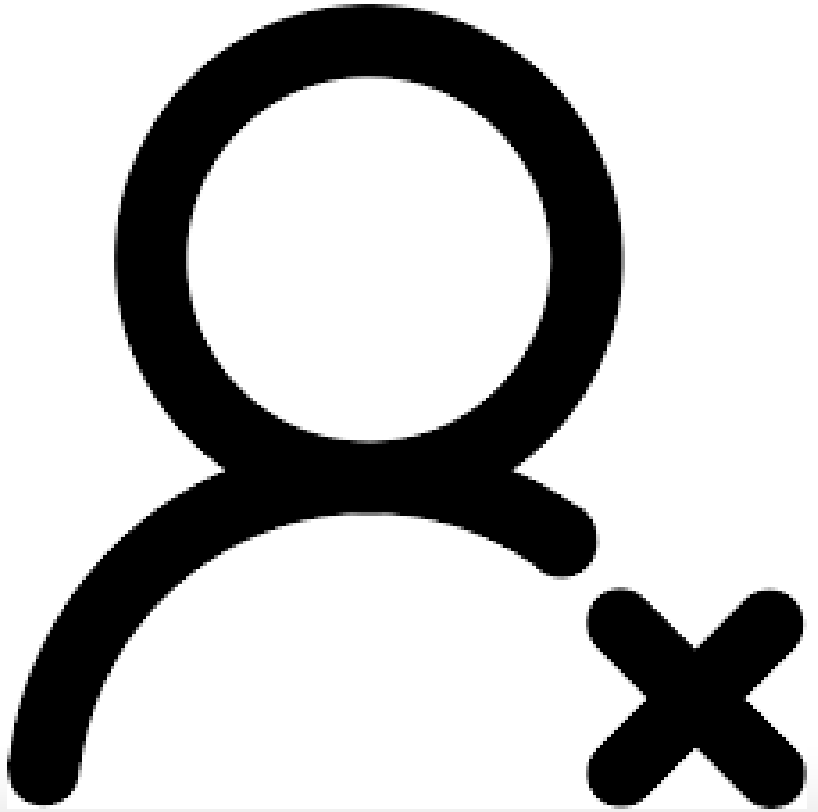
- Valid mail servers will have no problem if the receiving gives a soft error (4xx)
- They will attempt to send the mail again after some time
- Greylisting configured on a receiving mail server will give a soft error (4xx) to the sending server and store the IP/Hostname of the sending server in a file
- If the sending server returns again after some time (can be specified usually 5min) the email is accepted
- Used as a measure to deny mail from bots that are compromised to send mass mail. They often do not try again if the server did not accept the mail

# Accept only well formatted messages

- Sender must be a valid name not an IP ie not user@192.14.5.6
- Mail server HELO name must be resolvable ie FQDN
- Server identification must resolve ie HELO/EHLO name must be resolveable
- Email should be from a valid email address format eg: from tom@example.com and not from tom@example

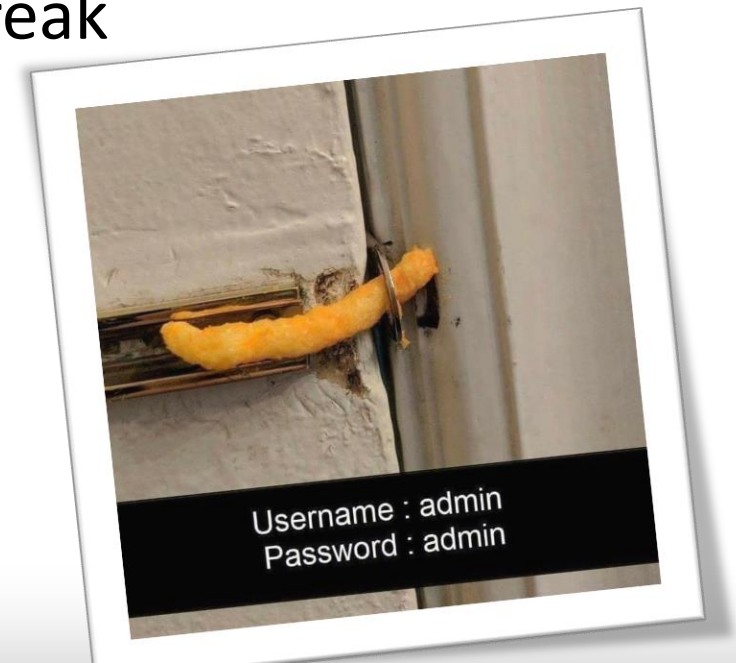
# Use Blacklist databases

- Use DNSBL – DNS Based Blackhole Lists or RBL (Real Time Blackhole lists) to deny mail from well known spamming machines
- Some well known good ones are
  - SORBS – http://sorbs.net
  - SPAMHAUS – http://spamhaus.org
  - SPAMCOP – http://spamcop.net
  - MANITU – http://manitu.net



# Require strong Passwords

- Advise users to use strong passwords or passphrases for their email
- Alphanumeric passwords are better than normal passwords ie combine letters with numbers
- Passphrases are even better, more difficult to break



# Backup and Redundancy

- Have multiple MX records so that your server is not the only one able to receive mail for you
- Backup your mail, use tools like Rsync to copy mail to another server as often as you can
- Ensure your DNS records (MX, NS etc ) are correct and test them when you complete you setup
- Use online tests like
  - http://intodns.net

# The question of Ethics

- As an email administrator, its easy to view other people's email at any time with admin rights
- Emails are intended by the sender for the recipient(s) and many senders are oblivious to the fact that their email can be intercepted along the way
- Hence the need for encryption ☺
- As an email administrator, you should be professional and maintain ethics and etiquette ☺

# My References:

**Kevin Chage slides for afnog workshops ☺**