Mail Server Provisioning on RHEL 8/9 (Enterprise-ready)

Installing Postfix, Dovecot with CRAM-MD5 and TLS/SSL (STARTTLS) via SASL with Kerberos integration and SSL

Install Postfix core component for handling SMTP mail delivery. Dovecot for IMAP and POP3 to end user mail clients

```
sudo dnf install postfix dovecot-imap dovecot-pop3d dovecot-sieve
```

Provide SASL and libraries for authentication- plus openSSL's TLS/STARTTLS libraries for encryption.

```
sudo dnf install openssl cyrus-sasl cyrus-sasl-lib cyrus-sasl-devel cyrus-sasl-gssapi cyrus-sasl-ldap cyrus-sasl-scram cyrus-sasl-md5 cyrus-sasl-plain openssl krb5-workstation
```

What is installed:

Postfix (postfix):

/usr/sbin/postfix: The Postfix control program.
/usr/bin/postconf: Configuration utility for simplifying editing /etc/postfix/main.cf (main config file)
/usr/sbin/master: The master daemon that manages Postfix processes.
/usr/sbin/smtpd: The Postfix SMTP server.
/etc/postfix/main.cf: Main Postfix configuration file.
/etc/postfix/master.cf: The configuration file for Postfix daemon processes.
/etc/postfix/sasl_passwd: Stores credentials for Postfix to authenticate to external SMTP servers for relaying outbound mail.
/var/spool/postfix: Mail queue directory.

Dovecot:

/usr/sbin/dovecot: The Dovecot main daemon.
/etc/dovecot: Contains all Dovecot configuration files.
/etc/dovecot/conf.d/*.conf: Individual configuration files for different Dovecot features (e.g., 10-mail.conf, 10-auth.conf).
/var/lib/dovecot: Stores mail user data and mailbox information.
/var/run/dovecot: Location for Dovecot socket files used for communication.

Cyrus SASL:

/usr/bin/saslpasswd2: Manages SASL authentication credentials (if using FreeIPA).
/usr/sbin/saslauthd: The SASL authentication daemon.
/usr/lib/libsasl2.so: Main shared library for SASL.
/usr/lib/sasl2/: This directory holds libraries and plugins for all the SASL mechanisms (see section for those).
/etc/default/saslauthd: Configuration for saslauthd daemon.
/etc/sasl2/smtpd.conf: Common mechanisms and options for SMTP servers
/etc/sasl2/imapd.conf: For configuring SASL for IMAP applications.
/usr/lib/sasl2/libsasl2.conf: Global SASL library settings (rarely changed).
/etc/sasl2/sampleapp.conf: Example configurations for alternate or custom authentication backends like SQL, etc.

OpenSSL:

/usr/bin/openssl: The command-line tool for using OpenSSL's features.
/etc/pki/tls/openssl.cnf: The OpenSSL configuration file.

Commands and Subcommands Used in Configuration and Management

-- Postfix:	
postfix start | stop | reload	Starts and stops the server, or reload from config file without restarting
postfix check	Checks the Postfix configuration for errors.
postconf -p	View current Postfix configuration settings.
postconf -e 'parameter=value'	Set a specific Postfix parameter.
postqueue -f	Show details of a specific mail message in the queue.
postfix flush	Flush the mail queue.
-- Dovecot:	
systemctl stop | start | restart dovecot	Starts, stops, or reloads the Dovecot service.
dovecot -n	Shows the running Dovecot processes and their PIDs.
dovecot -d	Enables debugging modegenerating more detailed logs.
-- SASL Config:	
systemctl start | stop | reload saslauthd	Starts, stops, or reloads the SASL service.
saslpasswd2	Manages SASL authentication credentials.
saslpasswd2 -a mechanism user -p	Creates user password for a specific mechanism (e.g., ldap, etc).
saslauthd -a <daemon>	Starts the SASL auth daemon (e.g., pam, ldap, shadow, sasldb)
-- OpenSSL for Cert Management:	
openssl genrsa -out <key_filename>.key 2048	Generates a private RSA key for server use.
openssl verify <cert_filename>.crt	Verifies the validity of your server certificate.
openssl req -new -key <key_file>.key -out <csr_file>.csr	Make Cert Signing Request (CSR) using private key.
openssl x509 -req -days 365 -in <csr_file>.csr -CA cert.pem -CAkey key.pem -CAcreateserial -out <cert_file>.crt	Signs the CSR to generate a server certificate using a Certificate Authority (CA).

Important Files and Directories:

-- <u>Postfix:</u>	
/etc/postfix	Contains all Postfix configuration files.
/etc/postfix/main.cf	Main Postfix configuration file.
/etc/postfix/master.cf	Configuration file for Postfix daemon processes.
/var/spool/postfix	Mail queue directory.
-- <u>Dovecot:</u>	
/etc/dovecot	Contains all Dovecot configuration files.
/etc/dovecot/conf.d/*.conf	Configs for features (e.g., 10-auth.conf is authentication, 10-mail.conf is mail storage).
/var/lib/dovecot	Stores mail user data and mailbox information.
/var/run/dovecot	Location for Dovecot socket files used for communication.
Cyrus SASL - /etc/sasl2/smtpd.conf	Configuration for SASL mechanisms and options for SMTP authentication.
OpenSSL - /etc/pki/tls/openssl.cnf	The OpenSSL configuration file.

SASL Mechanism Plugin Libraries Installed by cyrus-sasl Packages

By installing these SASL plugins, you can configure Postfix (or other services using SASL like LDAP) to support a broad range of authentication mechanisms suitable for different security requirements and environments. They get installed in /usr/lib/sasl2/

GSSAPI (Kerberos)	libgssapiv2.so	Use GSSAPI, commonly used with Kerberos for authentication (SSO).
LDAP	libldapdb.so	Authenticate via one communication method but using LDAP to check credentials
SCRAM-SHA	libscram.so	Salted Challenge Response Auth with hashing, a CRAM-MD5 replacement
CRAM/DIGEST-MD5	libcrammd5.so	Challenge-response plus hashing, DIGEST-MD5 for mutual auth, etc.
PLAIN	libplain.so	Plaintext password sent. Use when traffic already encrypted (e.g., TLS) or testing
ANONYMOUS	libanonymous.so	Allows clients to authenticate anonymously, no credentials
LOGIN	liblogin.so	Like PLAIN, plaintext credentials, for legacy systems or basic setups with TLS/SSL

Postfix/Dovecot/SASL Installation and Configuration Guide

These steps provide functional email server setup, which can scale to 500+ users with enough hardware and network

Although mentioned at the very beginning, here again are the items you need to install in one big chunk

Install Postfix core component for handling SMTP mail delivery. Dovecot for IMAP and POP3 to end user mail clients

```
        sudo dnf install postfix dovecot-imap dovecot-pop3d dovecot-sieve
```

Install SASL and libraries for authentication- plus openSSL's TLS/STARTTLS libraries for encryption and kerberos client.

```
        sudo dnf install openssl cyrus-sasl cyrus-sasl-lib cyrus-sasl-devel cyrus-sasl-gssapi cyrus-sasl-ldap cyrus-sasl-scram cyrus-sasl-md5 cyrus-sasl-plain krb5-workstation
```

Generate Server Private Key with OpenSSL

```
        # Generate the server's private key in  /etc/ssl/private/server.key. This way won't ask for passphrase after rebooting
sudo openssl genpkey -algorithm RSA -out /etc/ssl/private/server.key 2048
        # Generate a Certificate Signing Request (CSR) used to generate the certificate:
sudo openssl req -new -key /etc/ssl/private/server.key -out /etc/ssl/certs/server.csr
        # Create a self-signed certificate. For production use, it’s recommended to get a cert from a trusted CA.
sudo openssl x509 -req -days 365 -in /etc/ssl/certs/server.csr -signkey /etc/ssl/private/server.key -out /etc/ssl/certs/server.crt
        # That spit out /etc/ssl/certs/server.crt.  Finally, secure the private key file access is limited to the root user
sudo chmod 600 /etc/ssl/private/server.key
sudo chown root:root /etc/ssl/private/server.key
```

Kerberos Client Configuration

Edit /etc/krb5.conf and replace the following with your kerberos server's information:

```
[libdefaults]
   default_realm = YOUR.REALM
   dns_lookup_realm = false
   dns_lookup_kdc = true
[realms]
   YOUR.REALM = {
      kdc = kdc.example.com
      admin_server = kdc.example.com
   }
[domain_realm]
   .example.com = YOUR.REALM
   example.com = YOUR.REALM
```

Save the config, then create a service principal for Postfix by running ktpasswd (generaties key):

```
        ktpasswd -q -h /etc/postfix/sasl/postfix.keytab postfix@EXAMPLE.COM
```

SASL Base Configuration: saslauthd

This example prioritizes using Kerberos first to check credentials, then LDAP if kerberos isn't working

Edit /etc/sysconfig/saslauthd:

```
            # Enable saslauthd on startup
START=yes
            # Specify the authentication mechanisms
MECHANISMS="kerberos5:ldap"
            # Often this will be set to "ldap" which is fine. I want my config to try kerberos first then use ldap if it can't.
OPTIONS="-c -m /var/run/saslauthd -r"
# Here, the -c enables cache, -m sets the directory for the mux, and -r includes realm in username
```

Mux in this context refers to the Unix domain socket directory that is used to multiplex authentication requests from different client applications. This allows for secure and efficient IPC (Inter-Process Communication) for authentication purposes.

```
            # LDAP settings for saslauthd
ldap_servers: ldap://ldap.example.com
ldap_search_base: ou=users,dc=example,dc=com
ldap_filter: (uid=%u)
ldap_bind_dn: cn=admin,dc=example,dc=com
ldap_password: your_password
```

Configure Postfix [Mail Transfer Agent (MTA) responsible for receiving and sending emails]

Edit the main Postfix configuration file /etc/postfix/main.cf

Add or update the following settings:

```
            # Set the hostname for the mail server
myhostname = mail.example.com
            # Define the domain name
mydomain = example.com
            # Set the origin domain
myorigin = $mydomain
            # Specify the network interfaces Postfix will listen on (localhost and all IPv4 addresses)
inet_interfaces = all
            # Restrict to the domain and subdomains specified in mydestination
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain
            # Define the mailbox location for users
home_mailbox = Maildir/
            # Virtual aliases mapping
virtual_alias_maps = hash:/etc/postfix/virtual
            # Enable SMTP TLS
smtpd_tls_cert_file = /etc/ssl/certs/server.crt
smtpd_tls_key_file = /etc/ssl/private/server.key
            # Use TLS for incoming connections
smtpd_use_tls = yes
smtpd_tls_security_level = encrypt  # Changing to 'may' makes TLS optional
smtpd_tls_auth_only = yes
smtpd_tls_loglevel = 2       # Logging level for TLS transactions
smtpd_tls_received_header = yes   # Add TLS status to the received header
            # Session caching to improve performance
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
            # Disable outdated SSL protocols
smtpd_tls_mandatory_protocols = !SSLv2, !SSLv3
smtpd_tls_protocols = !SSLv2, !SSLv3
smtpd_recipient_restrictions = permit_mynetworks, reject_unauth_destination
```

Configure support for SASL

Edit /etc/postfix/sasl/smtpd.conf and put in the following

```
            # Specify SASL mechanisms Postfix should support
pwcheck_method: saslauthd
mech_list: GSSAPI SCRAM-SHA-256 SCRAM-SHA-1 DIGEST-MD5 CRAM-MD5
            #  mechlist order for comm methods- saslauthd says try kerberos for checking credentials, use LDAP if it fails
            # GSSAPI (for kerberos) Configuration, tell it where the server key is
keytab: /etc/postfix/sasl/postfix.keytab
```

Set up virtual alias mapping:

Create or edit the /etc/postfix/virtual file to define email address mappings:

```
sudo nano /etc/postfix/virtual
            # Add entries to map email addresses to local usernames
user1@example.com user1
user2@example.com user2
            # Save it, then update the Postfix virtual alias database:
sudo postmap /etc/postfix/virtual
```

Configure Dovecot [IMAP and POP3 services for retrieving emails]

Edit the main Dovecot configuration file /etc/dovecot/dovecot.conf

Add or update the following settings:

```
            # Enable IMAP and POP3 protocols
protocols = imap pop3
            # Specify the location for mailbox data
mail_location = maildir:~/Maildir
            # Set the hostname for IMAP and POP3 services
hostname = mail.example.com
            # Enable the use of SSL/TLS
ssl = yes
ssl_cert = </etc/ssl/certs/server.crt
ssl_key = </etc/ssl/private/server.key
            # Allow all users to access their mail
userdb {
  driver = passwd
}
```

Specify location for mail storage

Edit /etc/dovecot/conf.d/10-mail.conf

```
            # Add this:
mail_location = maildir:~/Maildir
```

Configure Dovecot’s authentication options

Open /etc/dovecot/conf.d/10-auth.conf:

```
            # Ensure the following settings are enabled:
disable_plaintext_auth =  yes
            # Enable specified authentication mechanisms
auth_mechanisms = gssapi scram-sha-256 scram-sha-1 digest-md5 cram-md5
            # Why no LDAP above? auth_mechanisms are communication methods used- not to check or store credentials

            # LDAP Authentication Settings
passdb {
  driver = ldap
  args = /etc/dovecot/dovecot-ldap.conf.ext
}
userdb {
  driver = ldap
  args = /etc/dovecot/dovecot-ldap.conf.ext
}
            # Add this if Dovecot needs to directly talk to kerberos without SASL (or using Postfix as a proxy)
auth_gssapi_hostname = yourhostname.example.com
```

Make or edit config extention file for basic LDAP info

Edit /etc/dovecot/dovecot-ldap.conf.ext to specify the LDAP server details for verifying credentials

```
            # Add the following items:
hosts = ldap.example.com
dn = cn=admin,dc=example,dc=com
dnpass = your_password
base = ou=users,dc=example,dc=com
scope = subtree
user_attrs = uid=user,homeDirectory=home,uidNumber=uid,gidNumber=gid
user_filter = (&(objectClass=posixAccount)(uid=%u))
pass_filter = (&(objectClass=posixAccount)(uid=%u))
```

Dovecot Alternate Config (Use PAM to use kerberos, then try LDAP)

I wanted to prioritize kerberos over LDAP, and then try other methods after trying those two first. This meant invoking /etc/sysconfig/saslauthd to handle the order of things. A second solution was to pass things to a PAM config for Dovecot which would then handle things in the order I wanted them.

-- /etc/dovecot/conf.d/10-auth.conf

```
# Enable specified authentication mechanisms for Dovecot
auth_mechanisms = gssapi scram-sha-256 scram-sha-1 digest-md5 cram-md5
# Set up password database to use pam, which refers to saslauthd
passdb {
  driver = pam
  args = session=yes dovecot
}
# User database using system's passwd file (could also be ldap if needed)
userdb {
  driver = passwd
}
```

[The file /etc/dovecot/dovecot-ldap.conf.ext is still relevant, but identical and removed here since it is redundant]

-- Configure PAM to for Dovecot to delegate order of operations

PAM configuration files (stored in /etc/pam.d) generally have three domains of a programs operations they address:

- Authentication management specifies how to verify the user's identity- to authenticate the user based on local system credentials, kerberos, LDAP, biometric input, SSO, or MFA, etc
- Account Management verifies if the user's account is in good standing and can be used for login (is not disabled or expired)
- Session Management configures settings for the session after successful login, to manage session environment, apply system resource limits, set environment variables, configure user namespaces for the session, or configure other session-related aspects.

Create or modify the PAM configuration file for Dovecot, /etc/pam.d/dovecot:

-- /etc/pam.d/dovecot:

```
#%PAM-1.0
# Authentication management (these are listed in order they are to be attempted)
auth       required     pam_unix.so nullok
auth       sufficient   pam_krb5.so use_first_pass
auth       sufficient   pam_ldap.so use_first_pass
auth       requisite    pam_succeed_if.so uid >= 1000 quiet_success
auth       required     pam_deny.so

# Account management
account    required     pam_unix.so
account    sufficient   pam_krb5.so
account    sufficient   pam_ldap.so

# Session management
session    required     pam_limits.so
session    sufficient   pam_krb5.so
session    optional     pam_ldap.so

# pam_unix.so: Checks the local /etc/passwd and /etc/shadow files.
# pam_krb5.so: Integrates Kerberos for authentication.
# pam_ldap.so: Integrates LDAP for authentication.
# use_first_pass: Passes the password from the first module to subsequent ones to avoid prompting the user multiple times.
```

-- Restart Services:

After editing the PAM configuration, restart the relevant services to apply the changes.

```
systemctl restart saslauthd
systemctl restart dovecot
```

You can test PAM functionality with tools like `dovecot auth test <username>`

Configuration Tips for Large Scale Deployments

Increase Connection and Message Limits (Postfix):

Adjust parameters in /etc/postfix/main.cf to handle more connections and larger message volumes:

```
default_process_limit = 100
smtpd_recipient_limit = 1000
smtpd_client_connection_count_limit = 50
smtpd_client_message_rate_limit = 100
```

Optimize queue management to handle large volumes of email efficiently (Postfix):

```
queue_minfree = 10000000
```

Optimize to handle a high number of simultaneous IMAP/POP3 connections (Dovecot):

```
service imap-login {
   process_min_avail = 16
   service_count = 0
   client_limit = 4096
}
service pop3-login {
   process_min_avail = 16
   service_count = 0
   client_limit = 4096
}
```

Enable mailbox indexing to speed up mailbox operations (Dovecot):

```
mail_plugins = $mail_plugins imap_quota
```

Leveraging iRedMail for Administrative Tasks

- Ease of use streamlines managing a robust mail server
- Simplifies implementing measures such as SPF, DKIM, and DMARC; spam filtering and user management
- A web-based administration panel (iRedAdmin) which simplifies the management of domains, user accounts, and mail server settings. Can integrate webmail clients like Roundcube and SOGo, which provide users with a feature-rich email interface.

Installation

```
wget https://github.com/iredmail/iRedMail/archive/refs/tags/1.5.2.tar.gz
tar zxvf 1.5.2.tar.gz
cd iRedMail-1.5.2
```

Run the Installation Script: 'sudo bash iRedMail.sh'
The interactive installation wizard will prompt for various configurations, such as mail storage path. web server selection, database selection for user and domain management (various SQL and LDAP options), domain and admin email settings. Post-installation, iRedMail will provide you with URLs for accessing the web admin interface (iRedAdmin) and webmail. Save the generated credentials for the admin user. Restart the server if necessary to apply all changes.

Integrating iRedMail Security Features

SPF (Sender Policy Framework) configures SPF automatically. This record specifies that only the mail servers specified in the MX records are allowed to send email for your domain. Ensure your DNS records include SPF settings:

```
example.com. IN TXT "v=spf1 mx -all"
```

DKIM (DomainKeys Identified Mail):

During the iRedMail installation, DKIM keys are generated and configured.
Add the DKIM public key to your DNS records as instructed in the post-installation steps it will be something like this:

```
default._domainkey.example.com   IN   TXT   "v=DKIM1; k=rsa;
p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDYmG... (rest of the public key)"
```

The part of that that says default can be any random string... unique is prefered but not manditory, allows for multiple keys so you can give them different selector names to help differenciate them

DMARC (Domain-based Message Authentication, Reporting, and Conformance) This record specifies that emails failing DMARC checks should be quarantined and reports sent to the specified address.
Create a DMARC record in your DNS to enforce policies on email sending:

```
_dmarc.example.com. IN TXT "v=DMARC1; p=quarantine; rua=mailto:dmarc-reports@example.com"
```

Spam Filtering:

iRedMail uses tools like Amavisd, SpamAssassin, and ClamAV. These are pre-configured, fine-tuning can be done

Adjusting Postfix and Dovecot Security Configs

Iredmail can handle generating SSL keys and certificates as part of its setup process, with the option to use self-signed, Let's Encrypt, or a commercial CA-provided cert. These should be added to your mail server configs. Be sure to comment the old lines out so they don't get lost

Postfix TLS/SSL configuration edit /etc/postfix/main.cf

```
smtpd_tls_cert_file = /etc/ssl/certs/iRedMail.crt
smtpd_tls_key_file = /etc/ssl/private/iRedMail.key
smtpd_tls_CAfile = /etc/ssl/certs/iRedMail_CA.crt
smtpd_use_tls = yes
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
smtpd_tls_security_level = encrypt
smtp_tls_security_level = encrypt
```

Dovecot SSL/TLS configuration:

It says to put this in /etc/dovecot/conf.d/10-ssl.conf but you can just put them in /etc/dovecot/dovecot.conf

```
ssl = required
ssl_cert = </etc/ssl/certs/iRedMail.crt
ssl_key = </etc/ssl/private/iRedMail.key
ssl_ca = </etc/ssl/certs/iRedMail_CA.crt
```

Restart the services to apply the changes:

```
sudo systemctl restart postfix dovecot
```

Other mail server-related topics of interest

DKIM (DomainKeys Identified Mail):

DKIM is a crucial email authentication standard that helps prevent email spoofing and phishing attacks. It uses digital signatures to verify the sender's identity. When you receive an email, the DKIM signature is checked against the sender's domain to ensure it wasn't forged. A failing DKIM signature might indicate a spoofing attempt (someone else trying to impersonate the sender's email address).

DMARC (Domain-based Message Authentication, Reporting, and Conformance):

DMARC is a powerful tool that builds on SPF and DKIM to provide email authentication and reporting. DMARC reports tell you how email receivers handled your emails (passed authentication, rejected, etc.). This allows you to identify potential spoofing attempts or delivery issues. DMARC reports provide lots of information during an email security investigation. They can reveal authentication failures, unauthorized use of your domain for sending emails, and potential phishing attempts impersonating your domain.

Mail Header Analysis for Advanced Troubleshooting:

Email headers contain a wealth of information about the email's journey. Analyze fields to diagnose delivery issues, identify spam characteristics, and verify message integrity.

Some of these include:

- Received (multiple times): Traces the email's path through mail servers, including timestamps and authentication methods used at each hop. This helps identify delays, routing issues, and potential spam characteristics (e.g., excessive hops through unknown servers).
- DKIM-Signature: Contains the digital signature added by the sender using their DKIM key for message verification. A failing DKIM signature might indicate a spoofing attempt.
- SPF (visible through Received headers): Summarizes the results of the Sender Policy Framework (SPF) authentication check. This helps verify if the sender's domain authorized sending the message.
- DMARC (visible through Received headers): If DMARC is implemented, these headers might indicate how the receiver handled the message based on authentication results (e.g., passed, rejected).
- Authentication-Results: Summarizes the authentication checks performed on the email by different mail servers. Provides a comprehensive overview of the email's authentication status.
- Received-SPF: Specifically shows the results of the SPF authentication check.
- X- Headers: These custom headers can provide additional insights depending on the service or application used.

MX Toolbox: https://mxtoolbox.com/
Spamhaus Project: https://www.spamhaus.org/
DNSBL (DNS BlackList/ RBL) lookup and FCrDNS: https://multirbl.valli.org/
DMARC.org Tools: https://dmarc.org/resources/deployment-tools/
The SPF Project: http://www.open-spf.org/Sitemap/