Red Hat Identity Management (IdM) - FreeIPA Identity Policy Audit

IdM server with integrated DNS using FreeIPA which incorporates Kerberos, LDAP, TLS CA, NTP, and BIND in one install.

Installation:

Begin with getting the needed packages:

```
sudo dnf install freeipa freeipa-server bind bind-utils bind-dyndb-ldap krb5-server krb5-libs chrony
```

Next, run the installer script. This provides integrated DNS which will be relied on by other FreeIPA components

```
sudo ipa-server-install --enable-dns (--enable-dns is not needed in RHEL8 and beyond)
```

During the run of the installer script, you will be asked for the domain name (realm), you'll need to set a Directory Manager password and a primary administrator password. You will also be asked for DNS settings like type (usually choose BIND, if it asks to), forwarders (8.8.8.8 is fine for non-enterprise or testing installs), and reverse DNS (use the in-addr.arpa reverse version of our primary IP address, i.e., 192.168.1.123 would be 123.1.168.192.in-addr.arpa). You can also expect to see some choices and setting for Kerberos integration, NTP , database type (often PostgreSQL), and LDAP.

Here is what we have after that completes:
Dogtag Certificate System: Certificate Authority & Registration Authority for certificate management
LDAP Server: Employs 389 Directory Server for user and group management.
MIT KDC: Kerberos Key Distribution Center is the basis for single sign-on
Apache: IdM administration functionalities need a built-in webserver, so there it is.
NTP (chrony in RHEL 8/9): Sets up the Network Time Protocol service
BIND: Integrates the BIND DNS server with the FreeIPA environment for DNS management
SSSD - client side component employing FreeIPA as authentication & identity provider superior to NSS & PAM.

FreeIPA SystemD Services:
FreeIPA Server: freeipa-server.service, ipactl.service, freeipa-healthcheck.service
Kerberos Key Distribution Center: krb5kdc.service
Kerberos DB Administration: kadmin.service
Directory Services (LDAP): slapd.service
DNS Server: named.service

Important Configuration Files:

The primary config file - /etc/ipa/default.conf

Server Settings:

realm (Required): Defines your FreeIPA realm name (e.g., EXAMPLE.COM).
server_principal: (Optional) Specifies name for FreeIPA server. Autogenerates as host/<hostname>@<realm>
server_cert: (Optional) Path to FreeIPA server certificate file
server_key: (Optional) Path to private key file associated with the server certificate
ca_cert: (Optional) Path to CA certificate used to sign the FreeIPA server certificate
offline: (Optional) True means disable communication with other FreeIPA servers (isolated deployments)

DNS Settings:

enable_dns: (Optional) Set to true to enable the integrated FreeIPA DNS server. Defaults to false
dns_forwarder: (Optional) Comma-separated list of IPs of DNS servers to forward unresolved queries to
dns_allow_update: (Optional) IPs or networks allowed to update DNS records. Default is 127.0.0.1
disable_anonymous_bind: Restrict anonymous BIND queries, improving security
forwarder_permit: Define IPs or networks allowed for DNS forwarding requests (prevents open relays)

Security Settings:

password_minimum_length: Set a minimum password length
password_require_mixed_case: Require mixed case (uppercase/ lowercase)
password_require_numeric: Require at least one number
password_require_special: Require at least one special character
user_enable_lockout: Enable user account lockout after failed login attempts
user_lockout_duration: Define the duration (minutes) a locked account remains inaccessible
allow_unsafe_kerberos_keytypes: Leaving this disabled prevents weak Kerberos encryption types
ca_cert_subject: Defines the subject info for a custom CA certificate
server_cert_subject: Defines the subject info for a custom FreeIPA server certificate
db_type: Specifies the database backend used by FreeIPA (defaults to postgresql)
allow_weak_password: Disabling this enforces strong passwords for IPA clients (keep false).

Debug and Logging:

debug_level: Sets the debug logging level (higher values provide more detailed logs).
log_file: Path to the log file for FreeIPA server events.

[For very specialized configs, an optional /etc/ipa/server.conf can be used for server-specific overrides. it would be read first but is seldom needed and this is simply a footnote to it "being a thing"]

Enforcing standardized user authentication with /etc/ipa/userauth.conf

A general idea of entiries in a /etc/ipa/userauth.conf file. Security management could mandate this be used to emphasize and/or standardize password policy and security configurations (sometimes simple alternates to kerberos). The details for each module is a little out of scope for this writing, but module docs would have specifics and actual items to replace what's below. Many configuration options will be in a module's config file.

Options in /etc/ipa/default.conf can also be over-ridden here, and there are some new ones:
password_history_depth: Define the number of previous passwords a user cannot reuse.
user_enable_lockout: Enable account lockout after a certain number of failed login attempts.
user_lockout_duration: Define the duration (minutes) an account remains locked after failed login attempts.

```
[Service: sudo]       #  Enable RADIUS authentication for sudo service
      authtype = radius
      server = radius.example.com  # Replace with actual RADIUS server address
      shared_secret = (secret)     # Replace with actual shared secret (not recommended in plain text)
      port = 1812                  # Default RADIUS port
      #   timeout = 3                 # RADIUS authentication timeout (seconds)
      #   nas_port_type = 5            # Network Access Server (NAS) port type

[Service: vpn]       #  Enable LDAP authentication for a custom VPN service
      authtype = ldap
      server = ldap.example.com    # Replace with actual LDAP server address
      basedn = dc=example,dc=com   # Replace with appropriate base DN for user search
      binddn = cn=FreeIPA_Bind_User,ou=Service Accounts,dc=example,dc=com  # Replace with bind DN
      bind_password = (secret)     # Replace with actual bind password (not recommended in plain text)
      #   search_scope = subtree       # LDAP search scope (base, onelevel, subtree)
      #   tls_cacertfile = /etc/ipa/certs/ca.crt  # Path to CA certificate for LDAP TLS

[Service: shell]       #  PAM for shell logins
      auth          pam_ServiceName.so

[Service: secureapp]       #  Enable token-based authentication for a specific application (hypothetical)
      authtype = token  # Assuming a token-based authentication module is installed

[Service: shell]       #  Disable alternative authentication for shell logins (only use Kerberos)
      alternative_authentication = false

[Service: console]       #  PIN login module.  Allows users to log in using a PIN instead of a password
      authtype = pin  # Assuming the PIN login module is installed
      #   pin_retries = 3  # Maximum allowed PIN attempts before lockout
      #   pin_length = 6   # Minimum PIN length

[Service: ssh]       #  2FA/OTP (Example: Google Authenticator- others include RSA SecurID, Duo Security, etc.
      require_otp = true  # Enforces OTP for SSH logins
      #  require_mfa = true  # Enforces MFA for SSH logins

[Service: myapp]       #  Social login module (hypothetical - Facebook for a custom web application):
      authtype = social  # Assuming a social login module is installed
      #   provider = facebook  # Specify Facebook as the social login provider
      #   client_id = your_facebook_app_client_id  # Replace with your Facebook App details
      #   client_secret = (secret)  # Replace with your Facebook App secret (avoid plain text)

[Service: sudo]       #  Certificate-based auth module using PKI for sudo service (Example: freeipa-certlogin):
      authtype = cert  # Assuming the freeipa-certlogin module is installed
      #   ca_certfile = /etc/ipa/certs/ca.crt  # Path to the Certificate Authority certificate
      #   require_crl_check = true  # Enforce Certificate Revocation List (CRL) checking

[Service: shell]       #  External database auth module (example: ipa_ldap_sync-  LDAP for shell logins):
      # Users are authenticated against FreeIPA, but user data is synchronized from LDAP server
      uri = ldaps://ldap.example.com:636
```

The IPA commands for user and resource management
ipa <category> <subcommand> [options] [arguments]
<category> is for example user, group, host, etc.) <subcommand> is the action (e.g., add, delete, show, etc.)
Most of categories typically have the subcommands add, delete, modify, show, show all (or list), and find

ipa config: Manage FreeIPA server configuration files
ipa package: Manage FreeIPA packages (installation, updates)
ipa profile: Manage FreeIPA server profiles (configurations)
ipa server: Manage the FreeIPA server itself (installation, configuration)
ipa vpnconfig: Manage VPN configuration options within FreeIPA
ipa trust: Manage trust relationships (e.g., with Active Directory)
ipa host: Manage FreeIPA hosts (machines joining the identity domain)
ipa hostgroup: Manage groups specifically for FreeIPA hosts (machines)
ipa interface: Manage network interfaces on the FreeIPA server
ipa nfsserver: Manage FreeIPA's NFS server configuration
ipa service: Manage FreeIPA services (applications requiring identity management)
ipa join: Joins a machine to a FreeIPA domain without using the client installation command.
ipa domain: Manage FreeIPA domains (logical groupings of identities)
ipa fqdn: Manage Fully Qualified Domain Names (FQDNs) associated with FreeIPA
ipa domaindns: Manage DNS domains integrated with FreeIPA
ipa dnskey: Manage DNS keys used for DNS signing (important for DNSSEC)
ipa dbbackup: Manage database backups and restores
ipa dnstable: Manage FreeIPA's internal data tables (use with caution)
ipa restore/backup: Create or load a backup of an IPA config into FreeIPA
ipa sync: Synchronize data with external directory services
ipa topology: Manage FreeIPA's server topology (replica management)
ipa vault: Manage FreeIPA vaults (secure storage for secrets)
ipa ca: Manage Certificate Authority operations (for internal PKI)
ipa cert: Manage certificates used by FreeIPA (server TLS, user certificates)
ipa tls: Manage Transport Layer Security (TLS) certificates
ipa kerberos: Manage Kerberos tickets and keytabs
ipa servicedelegationrule: Manage service delegation rules (allow services to request Kerberos tickets)
ipa servicedelegationtarget: Manage service delegation targets (used with service delegation rules)
ipa realmdomains: Manage realm domains used for Kerberos authentication
ipa diagnose: Perform diagnostic operations on the FreeIPA server
ipa monitor: Monitor the FreeIPA server's health and status
ipa find: Search for users, groups, hosts, and other FreeIPA objects
ipa user: Manage FreeIPA users - ipa userpolicy: Manage user password policies - ipa group: Manage groups
ipa passwd: Reset or change passwords for FreeIPA users and services
ipa pwpolicy (alias for userpolicy): Manage password policies (password complexity)
ipa shadow: Manage shadow password information (use with caution)
ipa permission: Manage individual permissions assigned to users or groups
ipa rightsource: Manage rights sources used for access control
ipa role: Manage FreeIPA roles (sets of permissions)
ipa relation: Manage relationships between FreeIPA objects (e.g., user-group membership)
ipa selinuxusermap: Manage SELinux user maps
ipa sshkey: Manage SSH keys for FreeIPA users and services
ipa hbacrule: For Host-Based Access Control (HBAC) - ipa hbacsvc (services) - ipa hbacsvcgroup (groups)
ipa sudocmd: Manage commands usable w/ sudo - ipa sudocmdgroup: for sudo command groups
ipa idrange: Manage ID ranges (used for ID mapping)
ipa locale: Manage locales used within the FreeIPA server

For details, you can use "ipa help <category>" for any of these.

Open ports for FreeIPA functionality:
TCP ports: 80, 443 (HTTP/S for web interface), 389, 636 (LDAP/S), 53 (DNS), 88 (Kerberos for Windows clients)
UDP ports: 88 (Kerberos), 53 (DNS), 67 and 68 (DHCP)
RPC and rstatd use random port numbers. Unless you have multiple FreeIPA servers or modules that need it, you are probably safe to not worry about opening ports- addressing this issue is outside the scope of this writing.

Important files and directories
/etc/ipa: This directory contains configuration files for FreeIPA.
/etc/ipa/client.conf: optional- for FreeIPA client on the server itself. Has location of the server and realm info
/etc/ipa/userauth.conf (Optional) - Defines authentication backends and policies for user login.
/etc/ipa/authpolicy.conf (Optional) - Configures authentication policy for FreeIPA services.
/etc/ipa/db.conf - If you have database configuration info outside of default.conf they would go in this

/var/lib/ipa: This directory contains data files for FreeIPA, including LDAP databases and Kerberos keytabs.
/etc/krb5.conf: Config for the Kerberos client, with location of Kerberos keytabs and realm information.
/etc/pki/pki-tomcat: Directory for the Dogtag CA Certificate Authority service
/etc/pki/pki-tomcat/alias: Contains the certificate database used by Dogtag.
/etc/ipa/certs/: Holds FreeIPA server user and service certificates, private keys
/root/cacert.p12: Admin access certificate - default name, PKCS#12 for Public Key Cryptography Stds #12
/etc/ipa/ca.crt: The CA certificate file used by clients to verify the FreeIPA server's identity.
/etc/ipa/nssdb: Contains the NSS (Network Security Services) database used for storing certificates and keys.
/var/lib/freeipa/dns/: Holds zone files managed by FreeIPA's internal DNS server. Each domain/subdomain gets a zone file.
The TLD zone file is the primary, containing records for users, computers, and other services.
FreeIPA manages user/group information and dictates DNS records while BIND takes directions

/var/log/ipa: This directory contains log files related to FreeIPA operations.
/var/log/ipa-server-install.log: Log file for FreeIPA server installation.
/var/log/ipa-client-install.log: Log file for FreeIPA client installation.

Applying basic system security mechanisms:

<u>Hardening FreeIPA with SELinux</u>
Install the package: dnf install policycoreutils-selinux-freeipa
SELinux types and contexts:

- ipa_var_lib_t: Files under /var/lib/ipa
- ipa_var_run_t: Files under /var/run/ipa
- ipa_log_t: Logs under /var/log/ipa
- ipa_tmp_t: Temporary files
- ipa_exec_t: Executable files

Applying contexts and set booleans:

Item	Context to apply	Save what was applied
/var/lib/ipa	semanage fcontext -a -t ipa_var_lib_t "/var/lib/ipa(/.*)?"	restorecon -Rv /var/lib/ipa
/var/run/ipa	semanage fcontext -a -t ipa_var_run_t "/var/run/ipa(/.*)?"	restorecon -Rv /var/run/ipa
/var/log/ipa	semanage fcontext -a -t ipa_log_t "/var/log/ipa(/.*)?"	restorecon -Rv /var/log/ipa
/tmp/ipa	semanage fcontext -a -t ipa_tmp_t "/tmp/ipa(/.*)?"	restorecon -Rv /tmp/ipa
Executables	semanage fcontext -a -t ipa_exec_t "/usr/libexec/ipa(/.*)?"	restorecon -Rv /usr/libexec/ipa
Allow LDAP over SSL (boolean)	setsebool -P allow_ipa_ldap_ssl 1	

Verifying SELinux contexts and booleans:
```
ls -Z /var/lib/ipa  &&  getsebool -a | grep ipa
```

To identify and resolve denials:
```
grep "denied" /var/log/audit/audit.log | audit2allow -M mypol
semodule -i mypol.pp
```

<u>Example configuration for firewalld:</u>
```
firewall-cmd --zone=internal --add-source=10.0.10.0/24 --permanent
firewall-cmd --zone=internal --add-source=172.16.20.0/24 --permanent  # Add 2nd subnet to internal zone
firewall-cmd --zone=internal --add-port=443/tcp --permanent  # HTTPS
firewall-cmd --zone=internal --add-port=80/tcp --permanent    # Optional for web interface
firewall-cmd --zone=internal --add-port=389/tcp --permanent  # LDAP
firewall-cmd --zone=internal --add-port=636/tcp --permanent  # LDAPS
firewall-cmd --zone=internal --add-port=88/udp --permanent  # Kerberos
firewall-cmd --zone=internal --add-port=88/tcp --permanent  # Kerberos (optional for Windows clients)
firewall-cmd --zone=internal --add-port=53/udp --permanent  # DNS
firewall-cmd --zone=internal --add-port=53/tcp --permanent  # DNS
firewall-cmd --zone=internal --add-port=67/udp --permanent  # DHCP server broadcasts
firewall-cmd --zone=internal --add-port=68/udp --permanent  # DHCP clients leases
firewall-cmd --permanent --default-zone=internal    # Set internal zone as default
firewall-cmd --reload    # Reload firewall configuration
```

Example configuration for Iptables:

```
# Chain for internal subnet 1 traffic
iptables -A INPUT -i eth0 -s 10.0.10.0/24 -p tcp --dport 443 -j ACCEPT
iptables -A INPUT -i eth0 -s 10.0.10.0/24 -p tcp --dport 80 -j ACCEPT  # for web interface
iptables -A INPUT -i eth0 -s 10.0.10.0/24 -p tcp --dport 389 -j ACCEPT
iptables -A INPUT -i eth0 -s 10.0.10.0/24 -p tcp --dport 636 -j ACCEPT
iptables -A INPUT -i eth0 -p udp -s 10.0.10.0/24 --dport 88 -j ACCEPT  # Kerberos
iptables -A INPUT -i eth0 -p tcp -s 10.0.10.0/24 --dport 88 -j ACCEPT  # Kerberos (optional for Windows clients)
iptables -A INPUT -i eth0 -p udp -s 10.0.10.0/24 --dport 53 -j ACCEPT  # for DNS
iptables -A INPUT -i eth0 -p tcp -s 10.0.10.0/24 --dport 53 -j ACCEPT  # for DNS
iptables -A INPUT -i eth0 -p udp -s 10.0.10.0/24 --dport 67 -j ACCEPT   # see DHCP server broadcasts
iptables -A OUTPUT -i eth0 -p udp -s 10.0.10.0/24 --sport 67 --dport 68 -j ACCEPT  # DHCP leases, broadcasts
# Allow established connections for subnet 1
iptables -A INPUT -i eth0 -s 10.0.10.0/24 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -o eth0 -d 10.0.10.0/24 -m state --state ESTABLISHED,RELATED -j ACCEPT

#  Chain for internal subnet 2 traffic
iptables -A INPUT -i eth0 -s 172.16.20.0/24 -p tcp --dport 443 -j ACCEPT
iptables -A INPUT -i eth0 -s 172.16.20.0/24 -p tcp --dport 80 -j ACCEPT  # for web interface
iptables -A INPUT -i eth0 -s 172.16.20.0/24 -p tcp --dport 389 -j ACCEPT
iptables -A INPUT -i eth0 -s 172.16.20.0/24 -p tcp --dport 636 -j ACCEPT
iptables -A INPUT -i eth0 -p udp -s 172.16.20.0/24 --dport 88 -j ACCEPT  # Kerberos
iptables -A INPUT -i eth0 -p tcp -s 172.16.20.0/24 --dport 88 -j ACCEPT  # Kerberos (for Windows clients)
iptables -A INPUT -i eth0 -p udp -s 172.16.20.0/24 --dport 53 -j ACCEPT  # for DNS
iptables -A INPUT -i eth0 -p tcp -s 172.16.20.0/24 --dport 53 -j ACCEPT  # for DNS
iptables -A INPUT -i eth0 -p udp -s 172.16.20.0/24 --dport 67 -j ACCEPT   # see DHCP server broadcasts
iptables -A OUTPUT -i eth0 -p udp -s 172.16.20.0/24 --sport 67 --dport 68 -j ACCEPT  # send DHCP leases, BC
# Allow established connections for subnet 2
iptables -A INPUT -i eth0 -s 172.16.20.0/24 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -o eth0 -d 172.16.20.0/24 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Post-installation checklist:

- Lock down the FreeIPA system with the options provided in the hardening section
- Create admin accounts with strong passwords for managing the FreeIPA domain. Organize them into groups
- Set up other users and groups to maintain consistency and organization within your domain.
- Define settings like lifetime for certificates issued by your CA if you're using an internal PKI for authentication.
- Issue server certificates for the FreeIPA server and others for secure communication within the domain.
- If needed, integrate FreeIPA with your existing DNS infrastructure for automatic DNS record management.
- Define Kerberos realm settings within FreeIPA if you plan to use Kerberos for authentication.
- Enroll machines (clients and servers) into the FreeIPA domain using the ipa join command.
- Install and configure FreeIPA client software on domain members
- Implement monitoring/logging solutions for server activity, identify potential issues, and ensure secure operation.
- Establish a regular backup and restore strategy for server configuration and data for disaster recovery
- Create user documentation explaining logging in, password resets, and accessing resources.
- Verify that users and groups are created and configured correctly within the FreeIPA domain.
- If using Kerberos, test user authentication using Kerberos tickets to ensure functionality.
- Verify that client software on domain members is functioning correctly and users can access resources