Docker: FreeIPA

FreeIPA is a collection of Tools for complete management of your Domain (as Active Directory) and a nice Web- GUI to mange the Users, Hosts, PCs and all Services.

As docker Image, it delivers LDAP for central storing of Users/Groups, Hosts and Keys. The bind- DNS will serve Hostnames to IP- Adresses on your local network and the included MIT- Kerberos- Implementation will deliver SSO attached to the Users and integrate your Client- PCs seamlessly into the domain. Further you can create addition Keys for encryption with Dogtag- Service.

This chapter will describe, how to install FreeIPA in a rootless Docker- Environement, use letsencrypt Cretificates for SSL and TLS and how to setup central User-Management with it.

Prerequisites

You will need a Docker- Host, that is rechable from the Internet with its fully qualifierd Domain- Name (FQDN) as described in the chapters before. The given Ports must be reachable from the clients.

Docker composer

FreeIPA will not use a Database - all needed informations are stored into the Docker Data- Volume. Some Services - like LDAP will setup their own DB in that Directory.

First, create a Directory in your Docker-Compose directry that you chose before in
http://obel1x.de/doku.php?id=content:serverbasics:docker#create_a_place_for_yamls

Then, create your `docker-compose.yml` like this:

```
services:
  freeipa:
    image: freeipa/freeipa-server:almalinux-9
    restart: unless-stopped
    hostname: [FQDN_HOSTNAME]
#For dns setup:
#    read_only: true
```

```
    environment:
      IPA_SERVER_HOSTNAME: [FQDN_HOSTNAME]
      TZ: "Europe/Berlin"
      PASSWORD: 'NEWPASSWORD'
#      DEBUG_NO_EXIT: 1
    tty: true
    stdin_open: true
    cgroup: host
    cap_add:
      - NET_ADMIN
    volumes:
      - /etc/localtime:/etc/localtime:ro
      - ipa_data:/data
      - ipa_journal:/var/log/journal
      - /run/user/[UID_OF_DOCKERUSER]/docker.sock:/var/run/docker.sock
      - /sys/fs/cgroup:/sys/fs/cgroup:rw
      - caddy_data:/etc/letsencrypt:ro
    tmpfs:
      - /run
      - /tmp
    sysctls:
      - net.ipv6.conf.all.disable_ipv6=0
      - net.ipv6.conf.lo.disable_ipv6=0
    security_opt:
      - "seccomp:unconfined"
    command:
      - -U
      - --domain=clients.[DOMAINPART_OF_HOSTNAME]
# Must match the last part of the Domain-Name and must be upper case and routed to the domain
      - --realm=[DOMAINPART_OF_HOSTNAME_UPPERCASE]
      - --http-pin=[NEWHTTPDPIN]
      - --dirsrv-pin=[NEWDIRSRVPIN]
#Bind/DNS Setup - use own Server Open Port 53 for this
      - --setup-dns
#      - --no-host-dns
 - --setup-dns
# Save choice:
 - --no-forwarders
```

```
# Will Forward unknow DNS- Queries to something else. May be a security- breach
# - --auto-forwarders
# - --forwarder=192.168.178.1
#NTP - not needed, this is the server which time is taken from /etc/localtime see volumes
#This server has also a chrony-daemon running here to sync time
      - --no-ntp
#      - --ntp-server=172.0.0.11
#Error: Unable to determine the amount of available RAM
      - --skip-mem-check
    ports:
#HTTP(s): Will be proxiesd by Caddy
#      - "8082:80"
#      - "8443:443"
#LDAP (needed)
      - "389:389"
#LDAPs (needed)
      - "636:636"
#Kerberos (needed)
      - "88:88"
      - "464:464"
      - "88:88/udp"
      - "464:464/udp"
#DNS (needed)
      - "53:53"
      - "53:53/udp"
#NTP - not needed, takes chronyd of the host
#      - "123:123/udp"
#Unknown:
#      - "7389:7389"
#      - "9443:9443"
#      - "9444:9444"
#      - "9445:9445"
volumes:
 ipa_data:
 ipa_journal:
 caddy_data:

networks:
```

```
# Still needs to be defined while without it won't enable ipv6
  default:
    driver: bridge
    enable_ipv6: true
```

The caddy_data Volume contains the Certifictes for encryption from Caddy as described in docker-caddy.

STRONG ADVISE: Do not open Ports of your firewall of the services Kerberos, LDAP or DNS until you configured everything first, otherwise your Server will be very insecure at this stage!

Caddyfile

Caddy wil be used for Proxy, so in your Caddyfile (see https://obel1x.de/dokuwiki/doku.php?id=content:serverbasics:docker-caddy#caddy_configuration) use this:

```
# FreeIPA
http://[FQDN_HOSTNAME]:80 {
        reverse_proxy serverpc-freeipa-1:80
}
https://[FQDN_HOSTNAME]:443 {
        header Strict-Transport-Security max-age=31536000;
        reverse_proxy https://serverpc-freeipa-1:443 {
            transport http {
                tls
                tls_insecure_skip_verify
            }
        }
}
```

As the internal Certificate of FreeIPA will be self-signed, the verification is turned off first. Later the Cert is replaced by the ACME- letsencrypt- Certificate of Caddy, so you may turn this on again. But there is no benefit, as the SSL Connection is always internally proxied by Caddy, so there will be NO insecured Connections to the net.

Certificate- Setup - SSL for LDAP and Kerberos

First thing you should do, is to secure the (Kerberos and LDAP)- ports with the certificate from letsenrcypt that Caddy gave you when openining the Webservice for IPA at [FQDN_HOSTNAME]. Without those matching certificates in place, Kerberos later won't accept the self signed- certificates that FreeIPA will create during install.

Your Docker-Yaml should have mounted the Certificates from Caddy to /etc/letsencrypt. To use the Certificates, you need to split the files named *.crt into the included parts.

The Parts will be: 1. Certificate for your FQDN 2. Certificate of the Issuer

For setting up the Truststore, we will nee the Certificate of the Issuer in one file and than import it to the keystore of freeipa webserver and ldap-server.

To do this, root-bash into your Container of running FreeIPA and do the following:

```
[root@ipa ~]# mkdir /data/caddy_certs_import
[root@ipa ~]# touch /data/caddy_certs_import/caddy_certs_import.sh
[root@ipa ~]# chmod u+x /data/caddy_certs_import/caddy_certs_import.sh
```

Now, paste the following in that file:

```
#!/usr/bin/bash
IPAPASS='[ipaAdminPassword]'
DOMAIN=$(hostname -f)
DAYMONTH=$(date +%d%m)
cd /data/caddy_certs_import
# We need to downlod and install the ISG-Root-Certificates of Letsencrpyt first
# Renew them once a year
if [ "${DAYMONTH}" -eq "2503" ]; then
 echo "Renew ISG-Root..."
 rm isrgrootx1.pem
fi
if [ ! -f isrgrootx1.pem ]; then
 curl https://letsencrypt.org/certs/isrgrootx1.pem> isrgrootx1.pem
fi
# Will result in Files cert00 cert01 cert02
csplit -f cert /etc/letsencrypt/caddy/certificates/acme-v02.api.letsencrypt.org-directory/${DOMAIN}//${DOMAIN}.crt '/-----
BEGIN CERTIFICATE-----/' '{*}'
```

```
# Install the Root-Certificate - this needs to be done every time (?)
ipa-cacert-manage install isrgrootx1.pem
# This will add the Letsencrypt-CAs- Certificate to the IPA- Certificate- Store for validiation of the Server-Certificate
ipa-cacert-manage -p ${IPAPASS} -t C,, install cert02
#Before this is usable, need to update the certificate-store in ipa
ipa-certupdate
#Now replace the Servers Certificate for Kerberos and HTTP plus LDAP - The Pin is only needed when the Key is encrypted by password
ipa-server-certinstall -w -d --pin='' -p ${IPAPASS} /etc/letsencrypt/caddy/certificates/acme-v02.api.letsencrypt.org-directory/${DOMAIN}/${DOMAIN}.crt /etc/letsencrypt/caddy/certificates/acme-v02.api.letsencrypt.org-directory/${DOMAIN}/${DOMAIN}.key
# Finally, restart IPA-Service
ipactl restart
```

Replace the [ipaAdminPassword] by your Admin- Password of FreeIPA (the one given in docker service as PASSWORD: …), then execute the script in the Container. It will take some time to restart the Services.

You need tho setup a cron-job daily after your Backup- Time- Slot to execute that Script every day.

Caution: Not renewing those Certificates will LOCK YOU OUT OF FREEIPA COMPLETEY with NOT OPTION to correct that after the certificates have expired!

Explanation of the Commands, Checks and Debugging

The first ipa-cacert-manage will install the intermediate- certificate of letsencrypt.

You can check ift its the right one, by using `openssl x509 -in cert-02 -text -noout` and check if the subject is like

```
Subject: C = US, O = Let's Encrypt, CN = E6
```

If the Subject contains you FQDN , its the wrong certificate.

When ipa-server-certinstall has been sucessfully run, your server should use those Certificates for Kerberos, LDAP and your Webservice.

Test if your Certificates for LDAP and Web do match:

openssl s_client -connect [FQDN_HOSTNAME]:636 -showcerts </dev/null

should be the same as:

openssl s_client -connect [FQDN_HOSTNAME]:443 -showcerts </dev/null

If not, check the above Certificates and loctaions to be from letsencrpt wih openssl. Mind that „openssl x509“ will only accept the first Certificate in a file. You can use „openssl certstore“ to ceck everything in the file, e.g.:

```
openssl storeutl -text -noout -certs ca.crt
```

Setup Cronjob on Host for Certificate- Renewal

for setting up a cronjob, use:

```
docker@servername:~/docker_compose/ipa> crontab -e
0 5 * * * /home/docker/docker_compose/ipa/ipa_certinstall.sh
```

using this Content:

```
docker@servername:~/docker_compose/ipa> cat ipa_certinstall.sh
#!/bin/bash
#Script to install certificates of Cadd/ACME/Letsencrypt to IPA in Container
LOGFILE='/home/docker/docker_compose/ipa/ipa_certinstall.log'
echo "ERROR in executing $(pwd)$0 without output!">${LOGFILE}
#
# Execute Renewal-Script in Docker-Container
/home/docker/bin/docker exec servername-freeipa-1 /data/caddy_certs_import/caddy_certs_import.sh>${LOGFILE} 2>&1
#
# Check the Result
SUCCSTRG=$(grep "ipa-server-certinstall command was successful" ${LOGFILE})
if [ ! -n "${SUCCSTRG}" ]; then
 # Result does not contain what is expected
 echo "ERROR when renewing Certificates $(pwd)$0"
 echo "."
 echo "THIS SHOULD BE FIXED, because it will break FreeIPA some Day and will cause very much trouble for that Service!"
 echo "Please Fix! The Log was:"
 echo "."
```

```
 cat ${LOGFILE}
fi
```

What if...

you did not setup that cron or there was an error and now everything is not working any more?

This may look like:

```
cannot connect to 'https://ipa.domain.tld:443/acme/directory': [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed:
certificate has expired (_ssl.c:1129)
The ipa-server-certinstall command failed.
```

Well, you're in trouble. The only way, i found to fix this, is to adjust the Hosts time to some time before that expiration- date, start the Container with „DEBUG_NO_EXIT: 1“ and run the script manally.

Web- Gui / Logon

Now, you should be able to login to your instance using User admin and the Password of your docker-compose file at https://ipa.domain.tld

LDAP - Zentrales AD

The LDAP- Servie ist the most basic and most important Service that the whole Domain relies on. It will contain all Details about Users, PCs (Server and Clients), Groups, Keys and Services. So pay strong Attention to the Security.

As of now, you should be able to query the LDAP already with your certificates in Place using your Admin- Password:

```
docker@servername:~> ldapsearch -xv -W -H ldaps://ipa.domain.tld -b "dc=clients,dc=domain,dc=tld" -D "cn=Directory Manager"
"objectclass=account"
ldap_initialize( ldaps://ipa.domain.tld:636/??base )
Enter LDAP Password:
filter: objectclass=account
```

```
requesting: All userApplication attributes
# extended LDIF
#
# LDAPv3
# base <dc=clients,dc=domain,dc=tld> with scope subtree
# filter: objectclass=account
# requesting: ALL
#

# search result
search: 2
result: 32 No such object
matchedDN: dc=domain,dc=tld

# numResponses: 1
```

This will return empty, but no errors.

You can also check the certificates of that service:

```
docker@servername:~> openssl s_client -connect ipa.domain.tld:636 </dev/null
CONNECTED(00000003)
depth=2 C = US, O = Internet Security Research Group, CN = ISRG Root X1
verify return:1
depth=1 C = US, O = Let's Encrypt, CN = E5
verify return:1
depth=0 CN = ipa.domain.tld
verify return:1
---
Certificate chain
 0 s:CN = ipa.domain.tld
   i:C = US, O = Let's Encrypt, CN = E5
   a:PKEY: id-ecPublicKey, 256 (bit); sigalg: ecdsa-with-SHA384
   v:NotBefore: Feb 10 20:49:39 2025 GMT; NotAfter: May 11 20:49:38 2025 GMT
 1 s:C = US, O = Let's Encrypt, CN = E5
   i:C = US, O = Internet Security Research Group, CN = ISRG Root X1
   a:PKEY: id-ecPublicKey, 384 (bit); sigalg: RSA-SHA256
   v:NotBefore: Mar 13 00:00:00 2024 GMT; NotAfter: Mar 12 23:59:59 2027 GMT
```

```
 2 s:C = US, O = Internet Security Research Group, CN = ISRG Root X1
   i:C = US, O = Internet Security Research Group, CN = ISRG Root X1
   a:PKEY: rsaEncryption, 4096 (bit); sigalg: RSA-SHA256
   v:NotBefore: Jun  4 11:04:38 2015 GMT; NotAfter: Jun  4 11:04:38 2035 GMT

.....
```

If there is another Certificate in place, check for the renewal above.

Security: No Anonymous access

By default, sadly that service is open to anonymous access, check out:

```
docker@servername:~> ldapsearch -xv -H ldaps://ipa.domain.tld:636 -b "dc=clients,dc=domain,dc=tld"
ldap_initialize( ldaps://ipa.domain.tld:636/??base )
filter: (objectclass=*)
requesting: All userApplication attributes
# extended LDIF
#
# LDAPv3
# base <dc=clients,dc=domain,dc=tld> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# search result
search: 2
result: 0 Success

# numResponses: 1
```

This is partly important, because Clients need to retrieve some basic Informations about the Service before Connecting to it with user- atuhentication. But you need to diable the full anonymous access by doing:

```
docker@servername:~> sudo dsconf -D "cn=Directory Manager" -W ldaps://ipa.domain.tld config replace 'nsslapd-allow-
anonymous-access=rootdse'
```

```
Enter password for cn=Directory Manager on ldaps://ipa.domain.tld:
SELinux is disabled, will not relabel ports or files.
Successfully replaced "nsslapd-allow-anonymous-access"
```

And now you see:

```
docker@servername:~> ldapsearch -xv -H ldaps://ipa.domain.tld:636 -b "dc=clients,dc=domain,dc=tld"
ldap_initialize( ldaps://ipa.domain.tld:636/??base )
filter: (objectclass=*)
requesting: All userApplication attributes
# extended LDIF
#
# LDAPv3
# base <dc=clients,dc=domain,dc=tld> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# search result
search: 2
result: 48 Inappropriate authentication
text: Anonymous access is not allowed.

# numResponses: 1
```

Security: Disable non encrypted access

Your Service is now reachable with SSL on Port 636 and with plain connection to port 389. The plain connection my by necessary for services, that do TLS, which is basically the same as SSL, but after the connection has been made to the plain port.

Without any further settings, the plain port will also accept non- encrypted connections and user- requests. That way, anyone could sniff your passwords:

```
docker@servername:~> ldapsearch -xv -w '123' -H ldap://ipa.domain.tld -b "dc=clients,dc=domain,dc=tld" -D "cn=Directory
Manager"
ldap_initialize( ldap://ipa.domain.tld:389/??base )
ldap_bind: Invalid credentials (49)
```

As you can see, the password is checked - which would alread be to late - it could already be sniffed.

Disable this:

```
docker@servername:~> sudo dsconf -D "cn=Directory Manager" -W ldaps://ipa.domain.tld config replace nsslapd-require-secure-
binds=on
[sudo] Passwort für root:
Enter password for cn=Directory Manager on ldaps://ipa.domain.tld:
SELinux is disabled, will not relabel ports or files.
Successfully replaced "nsslapd-require-secure-binds"
```

And now:

```
docker@pcserver2023:~> ldapsearch -xv -w '123' -H ldap://ipa.domain.tld -b "dc=clients,dc=domain,dc=tld" -D "cn=Directory
Manager"
ldap_initialize( ldap://ipa.domain.tld:389/??base )
ldap_bind: Confidentiality required (13)
        additional info: Operation requires a secure connection
```

Now, finally test if the TLS is still working:

```
docker@servername:~> ldapsearch -xv -Z -W -H ldap://ipa.domain.tld -b "dc=clients,dc=domain,dc=de" -D "cn=Directory Manager"
ldap_initialize( ldap://ipa.domain.tld:389/??base )
Enter LDAP Password:
filter: (objectclass=*)
requesting: All userApplication attributes
# extended LDIF
#
# LDAPv3
# base <dc=clients,dc=domain,dc=tld> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# search result
search: 3
```

```
result: 32 No such object
matchedDN: dc=domain,dc=tld

# numResponses: 1
```

Usage: CN=Config

To see your full config:

```
docker@servername:~> ldapsearch -xv -H ldaps://ipa.domain.tld:636 -W -b "cn=config" -D "cn=Directory Manager"> cn_config.txt
```

or only the root-dse with anonymous access:

```
docker@servername:~> ldapsearch -H ldaps://ipa.domain.tld -x -b "" -s base "objectclass=*"
# extended LDIF
#
# LDAPv3
# base <> with scope baseObject
# filter: objectclass=*
# requesting: ALL
#

#
dn:
objectClass: top
...
```

You will find many docs like https://docs.redhat.com/en/documentation/red_hat_directory_server/11/html/administration_guide/examples-of-common-ldapsearches in the internet how to use ldap.

Test User- Access

Now you can add an User in the FreeIPA- Web-GUI and test the connection.

To do this, logon to FreeIPA- Gui as Admin, than create a new user with some stupid password (not the real one!). You may change the e-Mail instantly after creation

and don't forget to save after changes at the user.

Than, logout from FreeIPA and login as your new User. You now need to change the password to the real password you want.

After that, the LDAP- Query should work. Test it:

```
docker@servername:~> ldapsearch -xv -W -H ldaps://ipa.domain.tld -b "cn=users,cn=accounts,dc=domain,dc=tld" -D
"uid=USERID,cn=users,cn=accounts,dc=domain,dc=tld"
ldap_initialize( ldaps://ipa.domain.tld:636/??base )
Enter LDAP Password:
filter: (objectclass=*)
requesting: All userApplication attributes
# extended LDIF
#
# LDAPv3
# base <cn=users,cn=accounts,dc=domain,dc=tld> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# users, accounts, domain.tld
dn: cn=users,cn=accounts,dc=domain,dc=tld
objectClass: top
objectClass: nsContainer
cn: users

...

# search result
search: 2
result: 0 Success

# numResponses: 4
# numEntries: 3
```

Jxplorer - GUI for LDAP

A very nice tool for exploring your LDAP- Tree is: http://jxplorer.org/index.html

Port opening

Now, the Server should be secured enough - you can open the firewall at the external ports in docker-compose and test the access from somewhere not in your network.

Central DNS for the Domain

Finally, the Hosts in the Domain will be managed by FreeIPA- DNS- Service, meaning every host that is integrated in the Domain will get a name (can be chosen freely) and will get an host-entry where the secure key for that host is attached to. To get this working, FreeIPAs DNS needs to manage the Sub-Domain you created in dokcer-compose.

In our Example, the subdamin would be: clients.domain.tld

Every Host will then get somename.clients.domain.tld as hostname.

That means, you have to tell your internet DNS- Provider, that clients.domain.tld will be managed by DNS- Server ipa.domain.tld

This is done by adding an NS- Entry to you Internet-DNS. Please check yout ISP- Docs for how to add that NS entry for your domain:

```
clients.domain.tld  42363 IN NS ipa.domain.tld.
```

Check out, if this works by:

```
dig +noall +answer +multiline clients.domain.tld NS
```

And at last:

```
nslookup ipa-ca.clients.domain.tld
```

This should return the IP of yout IPA- Server. If this works, you are done with the basic setup.

Client Setup

to fully use Single Sign On on your Client, you will need to

1. Setup your Browser to trust your IPA-Server
2. Install a working Kerberos-Client and enroll your PC to the Domain
3. Logon to your Linux- PC using SSSD/Kerberos

Install Kerberos-Client and enroll your PC to the Domain

Currently unfortunatelly i was not able to get Leap 15.6 working with freeipa-client (which worked for 15.5). So i switched to fedora (the KDE- Spin is very nice)

On Fedora, you can archive the Installation like this (Docs at https://www.freeipa.org/page/ConfiguringFedoraClients

```
#:> sudo yum install ipa-client
```

After that, go on with the next chapter https://obel1x.de/dokuwiki/doku.php?id=content:serverbasics:docker-freeipa#integrate_to_the_domain

---

Using Ferdora, skip this!

After a fresh Install of OpenSUSE, you frist need to get the Package freeipa-client.

I personally made it working that way:

- Open Yast
- Choose User and Group- Management
- Go to Authentication Settings
- Select SSSD and Add a new FreeIPA-Domain.
    - Enable Domain Logons

- Create Home-Directory = yes
- Sync Users and Groups
- SSH + Sudo = yes

As IPA- Server use `ipa.domain.tld`, no hostname set. The Network- Domain is `clients.domain.tld`

After those settings, exit from Yast. Do not care about Errors, that SSSD is not working: Your Client is missing important Domain Integration, so the service will fail.

As the time beeing, there is no official Package for Leap 15.6. So you may use mine:

```
zypper addrepo https://download.opensuse.org/repositories/home:obel1x/15.6/home:obel1x.repo
zypper refresh
zypper install freeipa-client

#Add Additional Packages / setup some needed files
pip3 install ifaddr
ln /usr/lib/mit/bin/kdestroy /usr/bin/kdestroy
```

Integrate to the Domain

After that, you need to setup your Client maybe with this small script, called ipa_register_host.sh which you can put to /root:

```
#!/bin/bash
TLDOMAIN=domain.tld
DOMAIN=clients.${TLDOMAIN}
SERVERFQDN=ipa.${TLDOMAIN}
#Serialnr of this device
HOSTNM=pc$(dmidecode -t system|grep -i "serial"|sed 's/Serial Number: //'|xargs|cut -c1-60|tr '[:upper:]' '[:lower:]'|sed
's/[^0-9a-z]*//g')
FQDN=${HOSTNM}.${DOMAIN}
echo "${FQDN} wird der Domain ${DOMAIN} hinzugefügt"
hostnamectl set-hostname ${HOSTNM}
#Check, if hostname is resolvable to this host - if not, add entry to /etc/hosts
if ! grep -q ${FQDN} "/etc/hosts"; then
  echo "Adding Host ${FQDN} to /etc/hosts"
  echo """>>"/etc/hosts"
```

```
  echo "127.0.0.1   ${FQDN} ${HOSTNM}">>"/etc/hosts"
fi
INSTCMD="ipa-client-install --mkhomedir --force-join --no-ntp --principal=admin --domain=${DOMAIN} --server=${SERVERFQDN} --
hostname=${FQDN}"
echo ${INSTCMD}
${INSTCMD}
echo "Ende Installation, sie können das Fenster schließen"
```

This script will integrate your PC into your IPA- Domain. Have the Password of your IPA- Admin ready.

After that, the SSSD should start, you may start, check and enable the Service. ATTENTION: When you now restart your PC and you have only WLAN enabled, the login WILL FAIL, because WLAN is activated AFTER logon. So SSSD cannot check your Domain and your Account and thus user logon will fail.

To prevent this, use a network- cable and configure Networking at system start, OR configure your wireless lan to be setup first. Or first logon as root, then as User.

You should frist check on non-graphical terminal if this will work, because errors will be shown there. Good Luck.

Setup your Browser to trust your IPA-Server

This one is on Firefox, as it works.

Go to your IPAs ipa.domain.tld/ipa/config/ssbrowser.html website. You can also find the LInk at the initial Logon-Page.

For me, the Button `Import Certificate` did not install automagically - use right- click and save to a file named ipa.crt.

Than open Firefox settings, Privacy and Security, Authorities- Tab and select Import. Use the downloaded file and select all Checkboxes. This installs your IPA-Authority to your Browser as trusted CA.

Do Steps 2 - 5 as described.

After that, and after loggon to your pc with your FreeIPA-User, the User should automagically be logged on when you open ipa.domain.tld.

If not, check if your klist shows some vaild Tickets. Otherwise inspect if this works:

```
HOSTNAME:~ # kinit admin
```

```
Password for admin@DOMAIN.TLD:
HOSTNAME:~ # klist
Ticket cache: KEYRING:persistent:0:0
Default principal: admin@DOMAIN.TLD

Valid starting     Expires            Service principal
04/07/25 12:58:27  04/08/25 12:40:12  krbtgt/DOMAIN.TLD@DOMAIN.TLD
```

This should be all needed to work for Firefox.

Setup Sudoers with FreeIPA/SSSD

This is quite a cool feature to have client admin- users managed by putting them in an IPA- group. When Loggin in with SSSD they will get added to the sudoers, making them admin on the given machines. Check this out: https://www.howtoforge.de/anleitung/wie-integriere-ich-sudoers-in-den-freeipa-server/

Additional Groups

You can also add System- Groups in IPA, that the client may have. E.g. a very nice group to have, would be a group named „wheel“. That group enables all users in it to install Software without beeing asked for a password.

You can add the clientadmins- group to the wheel- group so all users of the clientadmins group will be in wheel to (check in IPA with the „indirect members“ view, if wheel has all users, which clientsadmins has as „direct members“ to make it work !).

Next Steps

Next, you can integrate a Middleware for Authentication. You could, but you should NOT use FreeIPAs LDAP- Service directly as Authentication- Source for anything, as LDAP is very costy and would not deliver all needed APIs e.g. for SSO. This is part of your Middleware, so checkout docker-authentik to read further.

Special Annotations

Here are some Points, tha may be relevant in special Cases.

Backup and Restore

If you ever need to restore your IPA- Volumes (wihcih may be for e.g. after broken Updates), be very careful about ownership of the files. IPA contains many Services, that are critical about which user owns the configurationfiles. When you are Backing up with Nextclouds-Borg, you CANNOT restore those files 1:1 from your Host itself - as this may destroy ownerships.

Here are a few special files and users to pay attention to:

User Dirsrv

```
# chgrp named /data/etc/named.conf
# chown named:named /etc/named.keytab
# chown root:named -R -h -L /data/etc/named
# chown named:named -R -h -L /data/var/named

# chown dirsrv:dirsrv -R -h -L /data/var/lib/dirsrv
# chown dirsrv:dirsrv -R -h -L /data/var/log/dirsrv
# chown dirsrv:dirsrv -R -h -L /data/etc/dirsrv

# chown root:pkiuser /data/var/lib/ipa/pki-ca/publish -h -L
# chown pkiuser:pkiuser /data/var/lib/ipa/pki-ca/publish/* -h -L
# chown pkiuser:pkiuser /data/etc/sysconfig/pki-tomcat -h -L -R
# chown pkiuser:pkiuser /data/etc/sysconfig/pki/tomcat/pki-tomcat -h -L -R
# chown pkiuser:pkiuser /data/etc/pki/pki-tomcat -h -L -R
# chown pkiuser:pkiuser /data/etc/pki/pki-tomcat -h -L -R
# chown pkiuser:pkiuser /data/var/lib/pki/pki-tomcat -h -L -R
# chown pkiuser:pkiuser /data/var/log/pki/pki-tomcat -h -L -R

# chown root:named -h -L /etc/rndc.key

# chown root:ipaapi /data/var/lib/ipa/ra-agent.* -h -L
```

so e.g.:

```
# ls -lZ /etc/dirsrv/ds.keytab
-rw——-. dirsrv dirsrv system_u:object_r:dirsrv_config_t:s0 /etc/dirsrv/ds.keytab
```