Red Hat Enterprise Linux 5 Configuring Identity Management

Managing Identity and Authorization Policies for Linux-Based Infrastructures
Edition 2.1.4

Landmann

Red Hat Enterprise Linux 5 Configuring Identity Management

Managing Identity and Authorization Policies for Linux-Based Infrastructures
Edition 2.1.4

Landmann
rlandmann@redhat.com

Abstract

Identity and policy management — for both users and machines — is a core function for almost any enterprise environment. Identity Management (IPA) provides a way to create an identity domain that allows machines to enroll to a domain and immediately access identity information required for single sign-on and authentication services, as well as policy settings that govern authorization and access. This guide covers Linux client configuration and other service configuration for Linux clients within the IPA domain. For information on creating IPA servers and managing the IPA domain, see the Red Hat 6 Deployment Guide.

Table of Contents

Preface

Identity Management is a Red Hat Enterprise Linux-based way to create a security, identity, and authentication domain. The different security and authentication protocols available to Linux and Unix systems (like Kerberos, NIS, DNS, PAM, and sudo) are complex, unrelated, and difficult to manage coherently, especially when combined with different identity stores.

Identity Management provides a layer that unifies all of these disparate services and simplifies the administrative tasks for managing users, systems, and security. IPA breaks management down into two categories: *identity* and *policy*. It centralizes the functions of managing the users and entities within your IT environment (identity) and then provides a framework to define authentication and authorization for a global security framework and user-friendly tools like single sign-on (policy).

1. Audience and Purpose

This guide is written to cover the basic setup of a Red Hat Enterprise Linux 5 client to operate within the IPA domain.

The full administrative information, including configuring other platforms as clients, is covered in the Red Hat Enterprise Linux 6 *Deployment Guide*.

2. Examples and Formatting

Each of the examples used in this guide, such as file locations and commands, have certain defined conventions.

2.1. Brackets

Square brackets (**`[]`**) are used to indicate an alternative element in a name. For example, if a tool is available in **`/usr/lib`** on 32-bit systems and in **`/usr/lib64`** on 64-bit systems, then the tool location may be represented as **`/usr/lib[64]`**.

2.2. Client Tool Information

The tools for IPA are located in the **`/usr/bin`** and the **`/usr/sbin`** directories.

The LDAP tools used to edit the IPA directory services, such as **`ldapmodify`** and **`ldapsearch`**, are from OpenLDAP. OpenLDAP tools use SASL connections by default. To perform a simple bind using a username and password, use the **`-x`** argument to disable SASL.

2.3. Text Formatting and Styles

Certain words are represented in different fonts, styles, and weights. Different character formatting is used to indicate the function or purpose of the phrase being highlighted.

Formatting Style	Purpose
`Monospace with a background`	This type of formatting is used for anything entered or returned in a command prompt.

Formatting Style	Purpose
Italicized text	Any text which is italicized is a variable, such as *instance_name* or *hostname*. Occasionally, this is also used to emphasize a new term or other phrase.
Bolded text	Most phrases which are in bold are application names, such as **Cygwin**, or are fields or options in a user interface, such as a **`User Name Here:`** field or **`Save`** button. This can also indicate a file, package, or directory name, such as **`/usr/sbin`**.

Other formatting styles draw attention to important text.

Note

A note provides additional information that can help illustrate the behavior of the system or provide more detail for a specific issue.

Important

Important information is necessary, but possibly unexpected, such as a configuration change that will not persist after a reboot.

Warning

A warning indicates potential data loss, as may happen when tuning hardware for maximum performance.

3. Document Change History

Revision 2.1.4-15.400	**2013-10-31**	**Rüdiger Landmann**
Rebuild with publican 4.0.0		
Revision 2.1.4-15	**2012-07-18**	**Anthony Towns**
Rebuild for Publican 3.0		
Revision 5.8-0	**December 15, 2011**	**Ella Deon Lackey**
Release for beta of Red Hat Enterprise Linux 5.8.		

Chapter 1. Introduction to Identity Management

Red Hat Identity Management is a way to create identity stores, centralized authentication, domain control for Kerberos and DNS services, and authorization policies — all on Linux systems, using native Linux tools. While centralized identity/policy/authorization software is hardly new, Identity Management is one of the only options that supports Linux/Unix domains.

Identity Management provides a unifying skin for standards-defined, common network services, including PAM, LDAP, Kerberos, DNS, NTP, and certificate services, and it allows Red Hat Enterprise Linux systems to serve as the domain controllers.

Identity Management defines a domain, with servers and clients who share centrally-managed services, like Kerberos and DNS. This chapter first explains what Identity Management is.

1.1. A Working Definition for Identity Management

At the most basic level, Red Hat Identity Management is a domain controller for Linux and Unix machines. Identity Management defines the domain, using controlling servers and enrolled client machines. This provides centralized structure that has previously been unavailable to Linux/Unix environments, and it does it using native Linux applications and protocols.

Security information frequently relates to *identities* of users, machines, and services. Once the identity is verified, then access to services and resources can be controlled.

For efficiency, risk management, and ease of administration, IT administrators try to manage identities as centrally as possible and to unite identity management with authentication and authorization policies. Historically, Linux environments have had a very difficult time establishing this centralized management. There are a number of different protocols (such as NIS and Kerberos) which define domains, while other applications store data (such as LDAP) and still others manage access (such as sudo). None of these applications talk to each other or use the same management tools. Every application had to be administered separately and it had to be managed locally. The only way to get a consistent identity policy was to copy configuration files around manually or to try to develop a proprietary application to manage identities and policies.

The goal of Identity Management is to simplify that administrative overhead. Users, machines, services, and polices are all configured in one place, using the same tools. Because IPA creates a domain, multiple machines can all use the same configuration and the same resources simply by joining the domain. Users only have to sign into services once, and administrators only have to manage a single user account.

IPA does three things:

- Create a Linux-based and Linux-controlled domain. Both IPA servers and IPA clients are Linux or Unix machines. While IPA can synchronize data with an Active Directory domain to allow integration with Windows servers, it is not an administrative tools for Windows machines and it does not support Windows clients. Identity Management is a management tool for Linux domains.

- Centralize identity management and identity policies.

- Build on existing, native Linux applications and protocols. While IPA has its own processes and configuration, its underlying technologies are familiar and trusted by Linux administrators and are well established on Linux systems.

In a sense, Identity Management isn't making administrators do something new; it is helping them do it better. There are a few ways to illustrate that.

On one extreme is the *low control* environment. Little Example Corp. has several Linux and Unix servers, but each one is administered separately. All passwords are kept on the local machine, so there is no central identity or authentication process. Tim the IT Guy just has to manage users on every machine, set authentication and authorization policies separately, and maintain local passwords. With IPA, things come to order. There is a simple way to have central user, password, and policy stores, so Tim the IT Guy only has to maintain the identities on one machine (the IPA server) and users and policies are uniformly applied to all machines. Using host-based access control, delegation, and other rules, he can even set different access levels for laptops and remote users.

In the middle is the *medium control* environment. Mid-Example Corp. has several Linux and Unix servers, but Bill the IT Guy has tried to maintain a greater degree of control by creating a NIS domain for machines, an LDAP directory for users, and Kerberos for authentication. While his environment is well under control, every application has to be maintained separately, using different tools. He also has to update all of the services manually whenever a new machine is added to his infrastructure or when one is taken offline. In this situation, IPA greatly reduces his administrative overhead because it integrates all of the different applications together seamlessly, using a single and simplified tool set. It also makes it possible for him to implement single sign-on services for all of the machines in his domain.

On the other extreme is the *absent control* environment. At Big Example Corp., most of the systems are Windows based and are managed in a tightly-knit Active Directory forest. However, development, production, and other teams have many Linux and Unix systems — which are basically excluded from the Windows controlled environment. IPA brings native control to the Linux/Unix servers, using their native tools and applications — something that is not possible in an Active Directory forest. Additionally, because IPA is Windows-aware, data can be synchronized between Active Directory and IPA, preserving a centralized user store.

IPA provides a very simple solution to a very common, very specific problem: identity management.

1.2. Relationships Between Servers and Clients

IPA itself defines a *domain*, a group of machines that have shared configuration, policies, and identity stores. This shared configuration allows the machines (and users) within the domain to be aware of each other and operate together. This awareness can be used to enable cross-platform compatibility, like unifying Windows and Linux systems, or to enable infrastructure-wide single sign-on.

A client is simply any machine which is configured to operate within the IPA domain, using its Kerberos and DNS services, NTP settings, and certificate services. That's an important distinction: a client does not require a daemon or (necessarily) an installed product. It requires only system configurations which direct it to use IPA services.

For Red Hat Enterprise Linux systems, a certain number of platform tools are available for IPA to use, such as SSSD. These are *IPA-enabled platform applications*, which is simply a way of saying that they are aspects of the underlying platform that work with IPA services. Other tools, like certain PAM and NSS modules and IPA command-line utilities, are provided as IPA-specific packages that must be installed on the machine. These are *IPA-related components*.

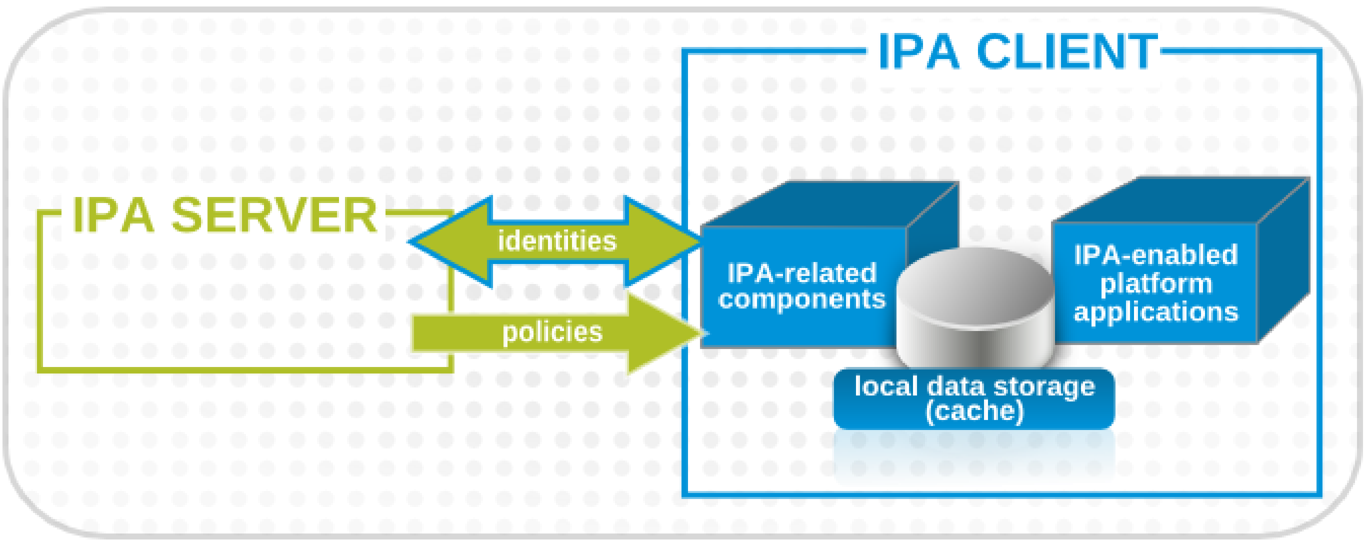


Figure 1.1. Server and Client Interactions

IPA uses the local storage (cache) on a client to improve performance in a few ways:

- Store IPA information when the machine is offline.
- Keep information active beyond its normal timeout period if the client cannot access the central server. The cache is persistent even after rebooting the machine.
- Reduce the round-trip time of requests by checking information locally before looking at the server.

Information is stored either in an LDB database (similar to LDAP) or the local filesystem (as XML files), depending on the *type* of information.

- Identity information (about users, machines, and groups) is stored in the LDB database, which uses the same syntax as an LDAP directory. This identity information is originally stored in the IPA server's 389 Directory Server instance. Because this information changes frequently and is referenced frequently, it is important to be able to call the more current information quickly, which is possible using an LDB database on the client and the Directory Server on the server.
- Policy information is more static than identity information, and it can include configuration for SELinux or sudo. These policies are set globally on the server and then are propagated to the clients. On the client, the policy information is stored in the filesystem in XML files which can be downloaded and converted into a native file for whatever service is being managed.

A specific set of services on the IPA server interact with a subset of services and modules on the IPA client. A client is any machine (a *host*) which can retrieve a keytab or certificates from the IPA domain.

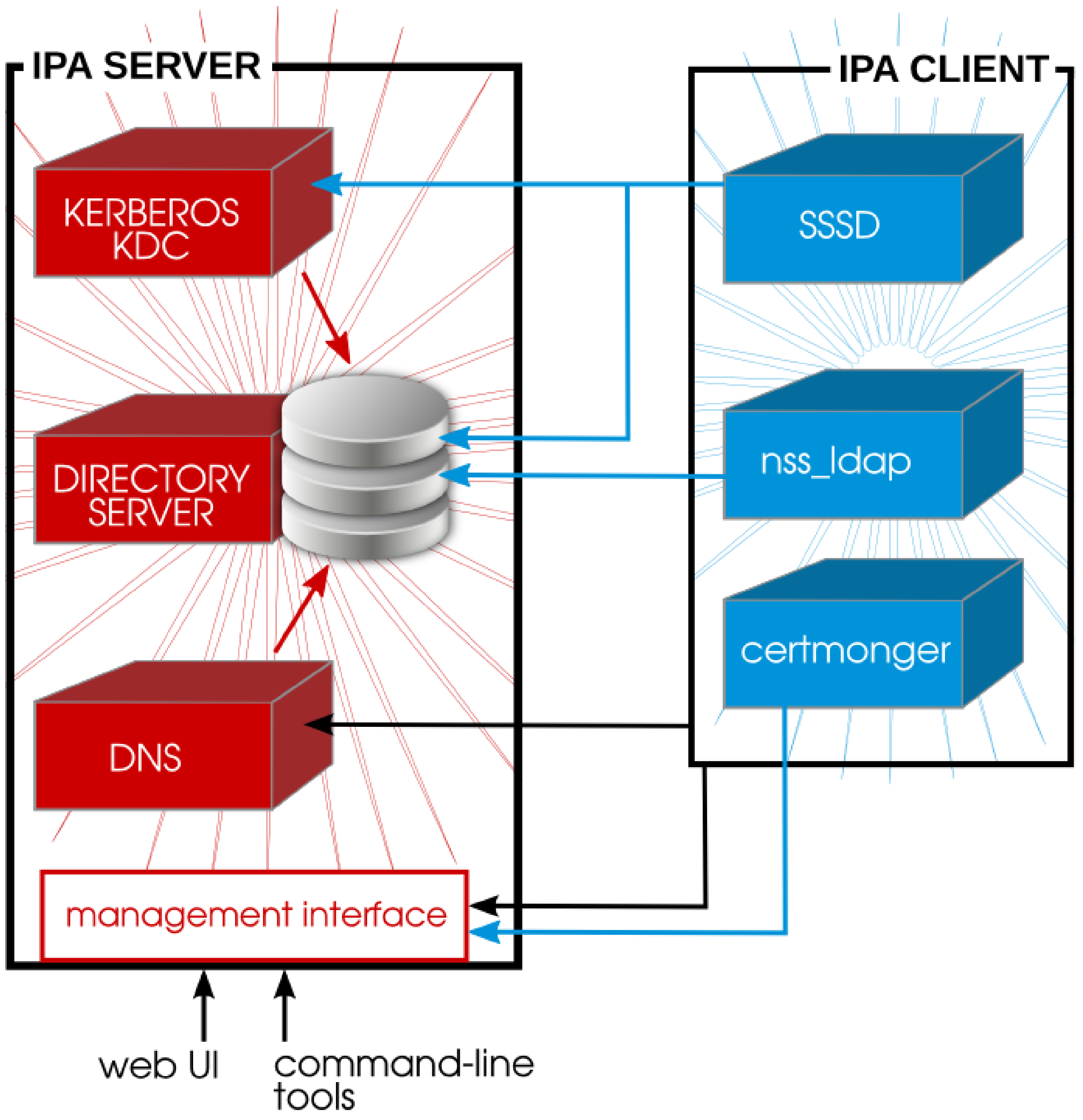


Figure 1.2. Interactions Between IPA Services

Figure 1.2, "Interactions Between IPA Services" shows that Red Hat Enterprise Linux uses two native daemons to interact with the IPA server:

- SSSD provides the user authentication for the machine and enforces host-based access control rules.
- **`certmonger`** monitors and renews the certificates on the client. It can request new certificates for the services on the system, including virtual machines.

When a Red Hat Enterprise Linux client is added to the domain (*enrolled*), its SSSD and **`certmonger`** are configured to connect to the IPA server and the required Kerberos keytab and host certificates are created. (The host certificate is not used directly by IPA; it may be used by other services, such as a web server.)

Chapter 2. Setting up Systems as IPA Clients

A *client* is any system which is a member of the Identity Management domain. While this is frequently a Red Hat Enterprise Linux system (and IPA has special tools to make configuring Red Hat Enterprise Linux clients very simple), machines with other operating systems can also be added to the IPA domain.

One important aspect of an IPA client is that *only* the system configuration determines whether the system is part of the domain. (The configuration includes things like belonging to the Kerberos domain, DNS domain, and having the proper authentication and certificate setup.)

Note

IPA does not require any sort of agent or daemon running on a client for the client to join the domain. However, for the best management options, security, and performance, clients should run the System Security Services Daemon (SSSD).

For more information on SSSD, see the SSSD chapter in the Deployment Guide.

This chapter explains how to configure a system to join an IPA domain.

Note

Clients can only be configured after at least one IPA server has been installed.

2.1. What Happens in Client Setup

Whether the client configuration is performed automatically on Red Hat Enterprise Linux systems using the client setup script or manually on other systems, the general process of configuring a machine to serve as an IPA client is mostly the same, with slight variation depending on the platform:

- Retrieve the CA certificate for the IPA CA.
- Create a separate Kerberos configuration to test the provided credentials. This enables a Kerberos connection to the IPA XML-RPC server, necessary to join the IPA client to the IPA domain. This Kerberos configuration is ultimately discarded.

  Setting up the Kerberos configuration includes specifying the realm and domain details, and default ticket attributes. Forwardable tickets are configured by default, which facilitates connection to the administration interface from any operating system, and also provides for auditing of administration operations. For example, this is the Kerberos configuration for Red Hat Enterprise Linux systems:

```
[libdefaults]
default_realm = EXAMPLE.COM
dns_lookup_realm = false
dns_lookup_kdc = false
rdns = false
forwardable = yes
ticket_lifetime = 24h

[realms]
```

```
EXAMPLE.COM = {
      kdc = ipaserver.example.com:88
      admin_server = ipaserver.example.com:749
      }
[domain_realm]
.example.com = EXAMPLE.COM
example.com = EXAMPLE.COM
```

- Run the **`ipa-join`** command to perform the actual join
- Obtain a service principal for the host service and installs it into **`/etc/krb5.keytab`**. For example, **`host/ipa.example.com@EXAMPLE.COM`**.
- Enable certmonger, retrieve an SSL server certificate, and install the certificate in **`/etc/pki/nssdb`**.
- Disable the nscd daemon.
- Configures SSSD or LDAP/KRB5, including NSS and PAM configuration files.
- Configure NTP.

2.2. Configuring a Red Hat Enterprise Linux System as an IPA Client

There are two elements to prepare before beginning the client setup process for the Red Hat Enterprise Linux client:

- There must be a way to connect the client machine to the Kerberos domain, either by having an available Kerberos identity (such as the admin user) or by manually adding the client machine to the KDC on the server with a one-time password before beginning the enrollment process for the client machine.
- If there is an Active Directory server on the same network that serves DNS records, the Active Directory DNS records could prevent the client from automatically detecting the IPA server address. The **`ipa-client-install`** script retrieves the Active Directory DNS records instead of any records that were added for IPA.

  In this case, it is necessary to pass the IPA server address directly to the **`ipa-client-install`** script.

To configure the client:

1. Install the client packages. These packages provide a simple way to configure the system as a client; they also install and configure SSSD.

   ```
   # yum install ipa-client
   ```

2. If the IPA server is configured as the DNS server and is in the same domain as the client, add the server's IP address as the first entry in the client's **`/etc/resolv.conf`** file.

   Note

   If every machine in the domain will be an IPA client, then add the IPA server address to the DHCP configuration.

3. Run the client setup command.

```
# ipa-client-install --enable-dns-updates
```

The **`--enable-dns-updates`** option updates DNS with the client machine's IP address. This option should only be used if the IPA server was installed with integrated DNS or if the DNS server on the network accepts DNS entry updates with the GSS-TSIG protocol.

When using the **`--server`** option to specify the IPA server to register with, the server name must be a fully-qualified domain name.

Important

This must be a valid DNS name, which means only numbers, alphabetic characters, and hyphens (-) are allowed. Other characters, like underscores, in the hostname will cause DNS failures.

Note

There is an **`--on-master`** option that is used as part of configuring an IPA server (which also is an IPA client, since it is within the domain). This option should *never* be used when configuring a regular IPA client, because it results in slightly different client configuration which may not work on a non-IPA server machine.

4. If prompted, enter the domain name for the IPA's DNS domain.

```
DNS discovery failed to determine your DNS domain
Please provide the domain name of your IPA server (ex: example.com):
example.com
```

5. If prompted, enter the fully-qualified domain name of the IPA server. Alternatively, use the **`--server`** option with the client installation script to supply the fully-qualified domain name of the IPA server.

```
DNS discovery failed to find the IPA Server
Please provide your IPA server name (ex: ipa.example.com):
ipaserver.example.com
```

Important

This must be a valid DNS name, which means only numbers, alphabetic characters, and hyphens (-) are allowed. Other characters, like underscores, in the hostname will cause DNS failures.

6. The client script then prompts for a Kerberos identity to use to contact and then join the Kerberos realm. When these credentials are supplied, then the client is able to join the IPA Kerberos domain and then complete the configuration:

```
Continue to configure the system with these values? [no]: yes
User authorized to enroll computers: admin
```

```
Password for admin@EXAMPLE.COM:
Enrolled in IPA realm EXAMPLE.COM
Created /etc/ipa/default.conf
Configured /etc/sssd/sssd.conf
Configured /etc/krb5.conf for IPA realm EXAMPLE.COM
SSSD enabled
Kerberos 5 enabled
NTP enabled
Client configuration complete.
```

7. Test that the client can connect successfully to the IPA domain and can perform basic tasks. For example, check that the IPA tools can be used to get user and group information:

   ```
   $ id
   $ getent passwd userID
   $ getent group ipausers
   ```

8. Set up NFS to work with Kerberos.

   Note

   To help troubleshoot potential NFS setup errors, enable debug information in the **`/etc/sysconfig/nfs`** file.

   ```
   RPCGSSDARGS="-vvv"
   RPCSVCGSSDARGS="-vvv"
   ```

   a. On an IPA server, add an NFS service principal for the NFS client.

      ```
      # ipa service-add nfs/ipaclient.example.com@EXAMPLE
      ```

      Note

      This must be run from a machine with the *ipa-admintools* package installed so that the **`ipa`** command is available.

   b. On the IPA server, obtain a keytab for the NFS service principal.

      ```
      # ipa-getkeytab -s ipaserver.example.com -p
      nfs/ipaclient.example.com@EXAMPLE -k /tmp/krb5.keytab
      ```

Note

Some versions of the Linux NFS implementation have limited encryption type support. If the NFS server is hosted on a version older than Red Hat Enterprise Linux 5, use the **-e des-cbc-crc** option to the **ipa-getkeytab** command for any nfs/<FQDN> service keytabs to set up, both on the server and on all clients. This instructs the KDC to generate only DES keys.

When using DES keys, all clients and servers that rely on this encryption type need to have the **allow_weak_crypto** option enabled in the **[libdefaults]** section of the **/etc/krb5.conf** file. Without these configuration changes, NFS clients and servers are unable to authenticate to each other, and attempts to mount NFS filesystems may fail. The client's **rpc.gssd** and the server's **rpc.svcgssd** daemons may log errors indicating that DES encryption types are not permitted.

c. Copy the keytab from the IPA server to the NFS server. For example, if the IPA and NFS servers are on different machines:

```
# scp /tmp/krb5.keytab root@nfs.example.com:/etc/krb5.keytab
```

d. Copy the keytab from the IPA server to the IPA client. For example:

```
# scp /tmp/krb5.keytab root@client.example.com:/etc/krb5.keytab
```

e. Configure the **/etc/exports** file on the NFS server.

```
/ipashare
gss/krb5p(rw,no_root_squash,subtree_check,fsid=0)
```

f. On the client, mount the NFS share. Use the same **-o sec** setting as is used in the **/etc/exports** file for the NFS server.

```
[root@client ~]# mount -v -t nfs4 -o sec=krb5p nfs.example.com:/
/mnt/ipashare
```

2.3. Manually Configuring a Linux Client

The **ipa-client-install** command automatically configures services like Kerberos, SSSD, PAM, and NSS. However, if the **ipa-client-install** command cannot be used on a system for some reason, then the IPA client entries and the services can be configured manually.

1. Install SSSD 1.5.x or later, if it is not already installed.

2. *On an IPA server.* Create a host entry for the client.

```
$ ipa host-add --force --ip-address=192.168.166.31 client1.example.com
```

3. *On an IPA server.* Create keytabs for the client.

   a. Log in as IPA; administrator.

```
$ kinit admin
```

b. Set the client host to be managed by the server.

```
$ ipa host-add-managedby --hosts=ipaserver.example.com
client1.example.com
```

c. Generate the keytab for the client.

```
# ipa-getkeytab -s ipaserver.example.com -p
host/client1.example.com -k /tmp/client1.keytab
```

4. Copy the keytab to the client machine and rename it **`/etc/krb5.ketab`**.

Note

If there is an existing **`/etc/krb5.ketab`** that should be preserved, the two files can be combined using **`ktutil`**.

5. Set the correct user permissions and, if necessary, SELinux contexts for the **`/etc/krb5.ketab`** file.

```
chown root:root 0600
system_u:object_r:krb5_keytab_t:s0
```

6. Configure SSSD by editing the **`/etc/sssd/sssd.conf`** file to point to the IPA domain.

```
[sssd]
config_file_version = 2
services = nss, pam

domains = example.com
[nss]

[pam]

[domain/example.com]
cache_credentials = True
krb5_store_password_if_offline = True
ipa_domain = example.com
id_provider = ipa
auth_provider = ipa
access_provider = ipa
ipa_hostname = client1.example.com
chpass_provider = ipa
ipa_server = ipaserver.example.com
ldap_tls_cacert = /etc/ipa/ca.crt
```

7. Configure NSS to use SSSD for passwords, groups, users, and netgroups.

```
vim /etc/nsswitch.conf
```

```
...
passwd:     files sss
shadow:     files sss
group:      files sss
...
netgroup:   files sss
...
```

8. Configure the **/etc/krb5.conf** file to point to the IPA KDC.

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = EXAMPLE.COM
 dns_lookup_realm = false
 dns_lookup_kdc = false
 rdns = false
 ticket_lifetime = 24h
 forwardable = yes
 allow_weak_crypto = true

[realms]
 EXAMPLE.COM = {
  kdc = ipaserver.example.com:88
  admin_server = ipaserver.example.com:749
  default_domain = example.com
}

[domain_realm]
 .example.com = EXAMPLE.COM
 example.com = EXAMPLE.COM
```

9. Update the **/etc/pam.d** configuration to use the **pam_sss.so** modules.

   - For **/etc/pam.d/fingerprint-auth**:

```
...
account     [default=bad success=ok user_unknown=ignore] pam_sss.so
...
session     optional      pam_sss.so
```

   - For **/etc/pam.d/system-auth**:

```
...
auth        sufficient    pam_sss.so use_first_pass
...
account     [default=bad success=ok user_unknown=ignore] pam_sss.so
...
password    sufficient    pam_sss.so use_authtok
...
session     optional      pam_sss.so
```

- For **`/etc/pam.d/password-auth`**:

```
...
auth         sufficient    pam_sss.so use_first_pass
...
account      [default=bad success=ok user_unknown=ignore] pam_sss.so
...
password     sufficient    pam_sss.so use_authtok
...
session      optional      pam_sss.so
```

- For **`/etc/pam.d/smartcard-auth`**:

```
...
account      [default=bad success=ok user_unknown=ignore] pam_sss.so
...
session      optional      pam_sss.so
```

10. Set up NFS to work with Kerberos.

Note

To help troubleshoot potential NFS setup errors, enable debug information in the **`/etc/sysconfig/nfs`** file.

```
RPCGSSDARGS="-vvv"
RPCSVCGSSDARGS="-vvv"
```

a. On an IPA server, add an NFS service principal for the NFS client.

```
# ipa service-add nfs/ipaclient.example.com@EXAMPLE
```

Note

This must be run from a machine with the *ipa-admintools* package installed so that the **`ipa`** command is available.

b. On the IPA server, obtain a keytab for the NFS service principal.

```
# ipa-getkeytab -s ipaserver.example.com -p
nfs/ipaclient.example.com@EXAMPLE -k /tmp/krb5.keytab
```

Note

Some versions of the Linux NFS implementation have limited encryption type support. If the NFS server is hosted on a version older than Red Hat Enterprise Linux 5, use the **`-e des-cbc-crc`** option to the **`ipa-getkeytab`** command for any nfs/<FQDN> service keytabs to set up, both on the server and on all clients. This instructs the KDC to generate only DES keys.

When using DES keys, all clients and servers that rely on this encryption type need to have the **`allow_weak_crypto`** option enabled in the **`[libdefaults]`** section of the **`/etc/krb5.conf`** file. Without these configuration changes, NFS clients and servers are unable to authenticate to each other, and attempts to mount NFS filesystems may fail. The client's **`rpc.gssd`** and the server's **`rpc.svcgssd`** daemons may log errors indicating that DES encryption types are not permitted.

c. Copy the keytab from the IPA server to the NFS server. For example, if the IPA and NFS servers are on different machines:

```
# scp /tmp/krb5.keytab root@nfs.example.com:/etc/krb5.keytab
```

d. Copy the keytab from the IPA server to the IPA client. For example:

```
# scp /tmp/krb5.keytab root@client.example.com:/etc/krb5.keytab
```

e. Configure the **`/etc/exports`** file on the NFS server.

```
/ipashare
gss/krb5p(rw,no_root_squash,subtree_check,fsid=0)
```

f. On the client, mount the NFS share.

- Always specify the share as *nfs_server:/ /mountpoint*.
- Use the same **`-o sec`** setting as is used in the **`/etc/exports`** file for the NFS server.

```
[root@client ~]# mount -v -t nfs4 -o sec=krb5p nfs.example.com:/
/mnt/ipashare
```

2.4. Performing a Split Enrollment

Enrolling machines as clients in the IPA domain is a two-part process. A host entry is created for the client (and stored in the 389 Directory Server instance), and then a keytab is created to provision the client.

Both parts are performed automatically by the **`ipa-client-install`** command. It is also possible to perform those steps separately; this allows for administrators to prepare machines and IPA in advance of actually configuring the clients. This allows more flexible setup scenarios, including bulk deployments.

When performing a manual enrollment, the host entry is created separately, and then enrollment is completed when the client script is run, which creates the requisite keytab.

Note

There are two ways to set the password. You can either supply your own or have IPA generate a random one.

There may be a situation where an administrator in one group is prohibited from *creating* a host entry and, therefore, from simply running the **`ipa-client-install`** command and allowing it to create the host. However, that administrator may have the right to run the command *after* a host entry exists. In that case, one administrator can create the host entry manually, then the second administrator can complete the enrollment by running the **`ipa-client-install`** command.

1. An administrator creates the host entry on the IPA server.

2. The second administrator installs the IPA client packages on the machine, as in Section 2.2, “Configuring a Red Hat Enterprise Linux System as an IPA Client”.

3. When the second administrator runs the setup script, he must pass his Kerberos password and username (principal) with the **`ipa-client-install`** command. For example:

   ```
   $ ipa-client-install -w secret -p admin2
   ```

4. The keytab is generated on the server and provisioned to the client machine, so that the client machine is not able to connect to the IPA domain. The keytab is saved with **`root:root`** ownership and 0600 permissions.

Chapter 3. Basic Usage

All of the access to Identity Management, both through the web UI and through the command line, is done by a user authenticating to the IPA domain. This chapter covers the basics of setting up browsers to handle Kerberos authentication, logging into Identity Management, and troubleshooting some common connection issues.

3.1. Looking at the IPA UI

The Identity Management web UI is designed for simplicity. This was the primary design goal, and this means that the web UI offers benefits that make using IPA simpler and clearer:

- It shows instant, visual relationships between entries (such as a user and all the groups, sudo rules, netgroups, and policies which are associated with that user).
- All entries are listed immediately without having to run a search. This makes it possible to browse entries. The UI also has a simple search box which quickly filters the list of entries.
- The interface is intuitive to use, without having to learn the command-line tools.
- The web UI can be accessed from machines outside the IPA domain, so the domain can be managed from anywhere.

  Using the web UI requires a valid Kerberos ticket for the IPA domain (by default), meaning that it can only be accessed from a machine within the IPA domain. Alternatively, the web UI can be configured to allow password authentication along with Kerberos authentication (Section 3.3.5, “Enabling Username/Password Authentication in Your Browser”), or a machine outside the IPA can be configured to work with Kerberos (Section 3.3.4, “Using a Browser on Another System”).

3.1.1. The UI Layout

The web UI has three major functional areas which correspond to each of the major functions of IPA: identity management, policy management, and domain configuration.

Table 3.1. Configuration Areas Per Tab

Main Menu Tab	Configuration Areas
Identity	• User entries • User groups entries • Host/client entries • Host group entries • Netgroups entries • Domain services entries • DNS (if configured)
Policy	• Host-based access control • Sudo rules • Automount • User password policies • Kerberos ticket policy

Main Menu Tab	Configuration Areas
Access controls within Identity Management	- Role-based access control (permissions based on group membership) - Self permissions - Delegations (user access control over other users)

The *main menu* at the top of every page has three tabs which correspond to the functional areas listed in Table 3.1, “Configuration Areas Per Tab”. When a tab is selected, there is a submenu of the different configuration areas. Some configuration areas may have multiple possible entries; for example, role-based access controls define user roles/groups, the areas that access can be granted or denied (privileges), and then the permissions granted to those areas. Each separate configuration entry has its own task area beneath the primary configuration area.

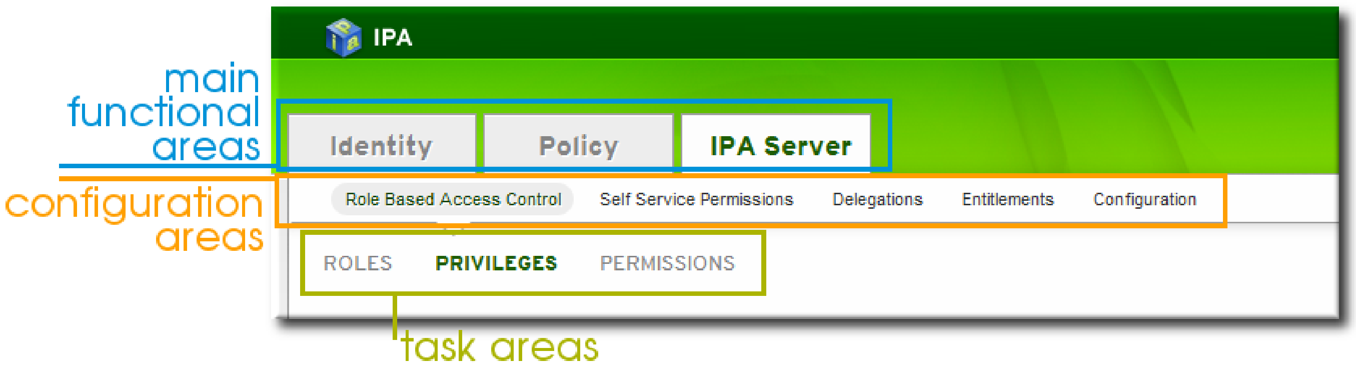
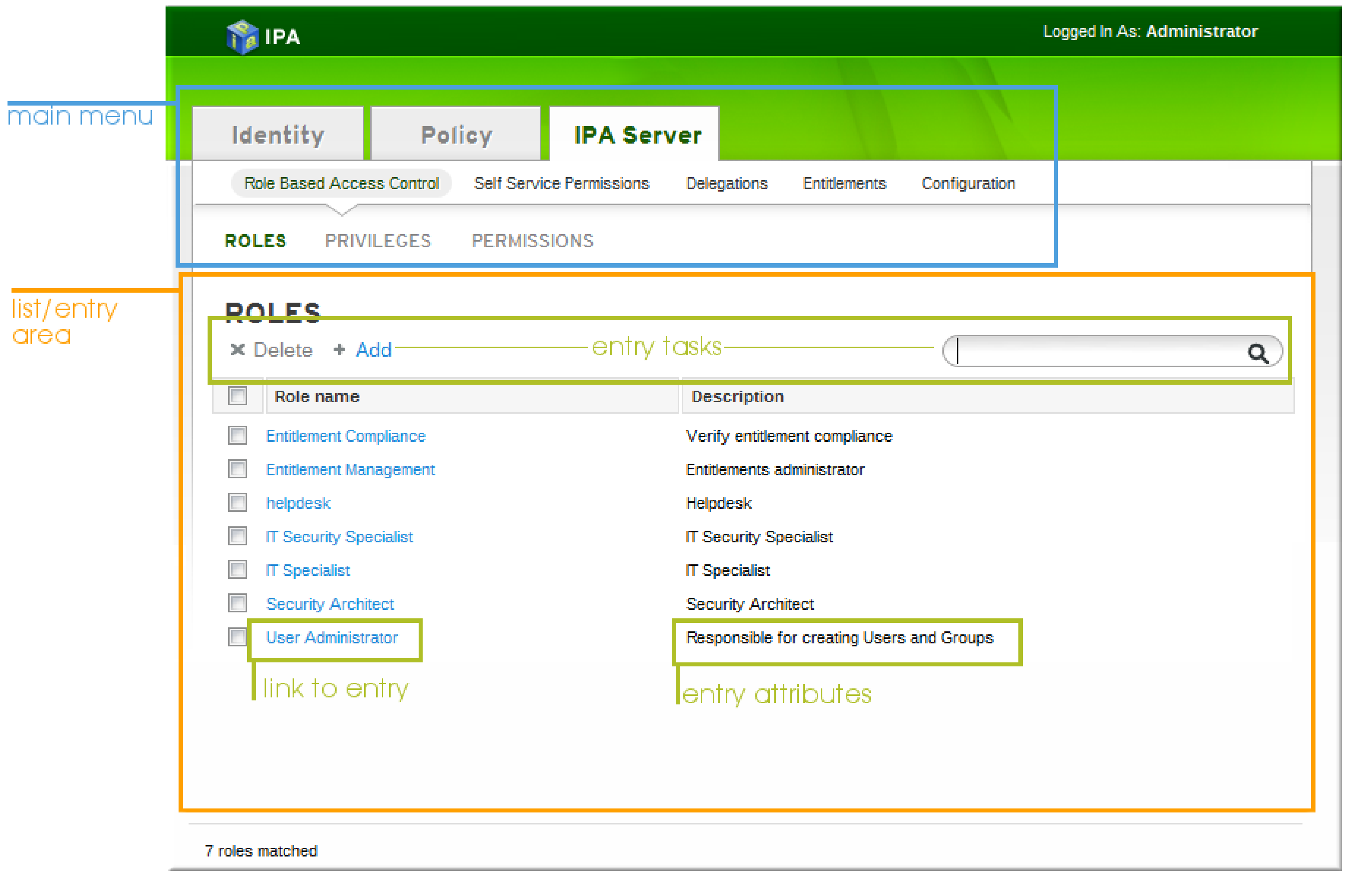


Figure 3.1. The Main Menu

All entries for a configuration area are listed together on the main page for that area. This page provides direct links to individual entry pages, as well as basic information (the *attributes*) about the entry. (This is usually just the description, but user entries show a lot more information.)

The page also has some tasks that can be performed on it. For a list page that shows entries, this can be creating or deleting an entry. For a list page for groups, the tasks are for establishing relationships between entities, either by adding (*enrolling*) or removing an entity from that group. Both individual entries and groups can be searched for through the list page.

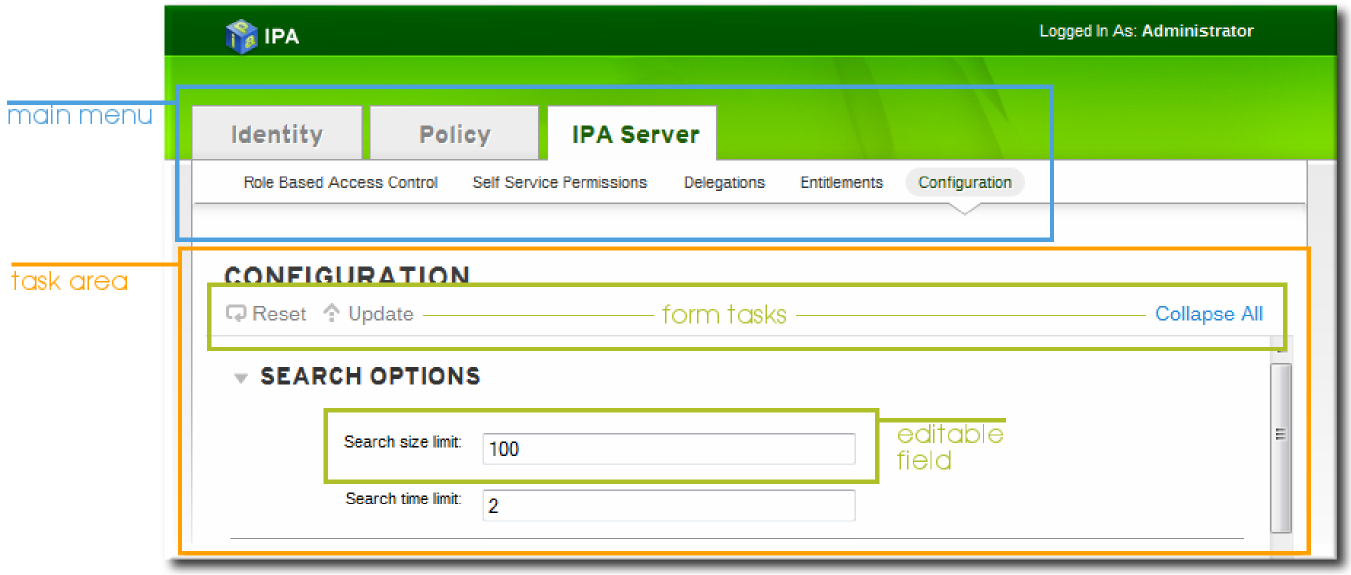


Role name	Description
Entitlement Compliance	Verify entitlement compliance
Entitlement Management	Entitlements administrator
helpdesk	Helpdesk
IT Security Specialist	IT Security Specialist
IT Specialist	IT Specialist
Security Architect	Security Architect
User Administrator	Responsible for creating Users and Groups

7 roles matched

Figure 3.2. List View

Each entry page is a form which allows that entry to be edited. This is done by editing text *fields* or by selecting items from drop-down menus.



Search size limit: 100

Search time limit: 2

Figure 3.3. Form/Entry View

3.1.2. Page Elements

The web UI uses common elements on all pages.

The most basic is that all blue text is a link to an entry or to an action.

When a task like adding an entry or saving a change is possible, the task link it blue. When it is not possible (such as no items have been selected to be deleted) then the task is grayed out.

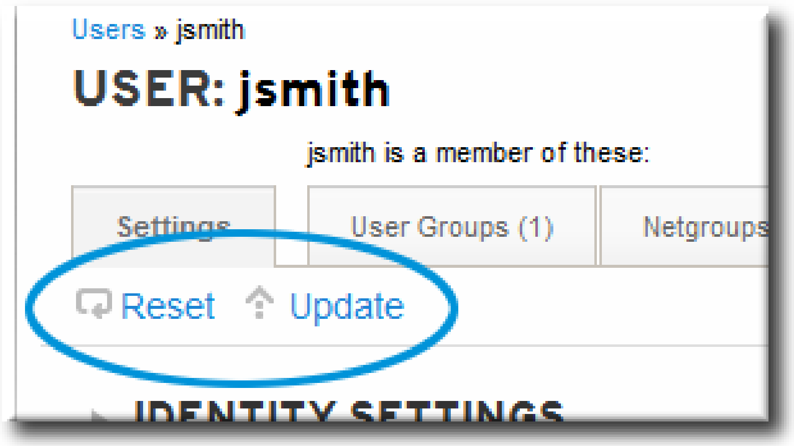


Figure 3.4. Active Task Link

All list pages display direct links to entry pages. However, some entries are essentially nested. For example, in automount configuration, the primary entry is the location, and then keys, mount points, and maps are associated with that location as children entries. This hierarchy is reflected in breadcrumb navigation near the top of the page, so it is easy to identify where you are in the UI and how this entry relates to any other related entries.

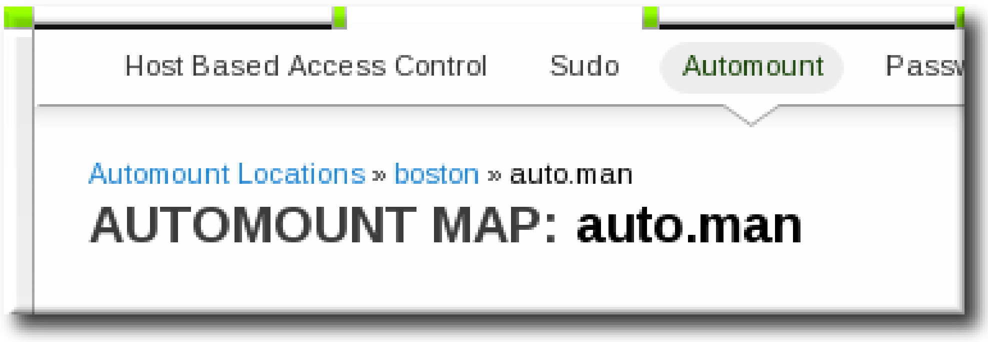

Host Based Access Control
Sudo
Automount
Passw
Automount Locations » boston » auto.man
AUTOMOUNT MAP: auto.man

Figure 3.5. Entry Breadcrumbs

Most entries have a variety of different configuration areas. A simple user entry has account activity settings, personal information, address information, organizational information, and other contact information. Related attributes are grouped together logically in the UI. These entry form areas can be collapsed or expanded using the arrows to control the amount of information displayed on the page.

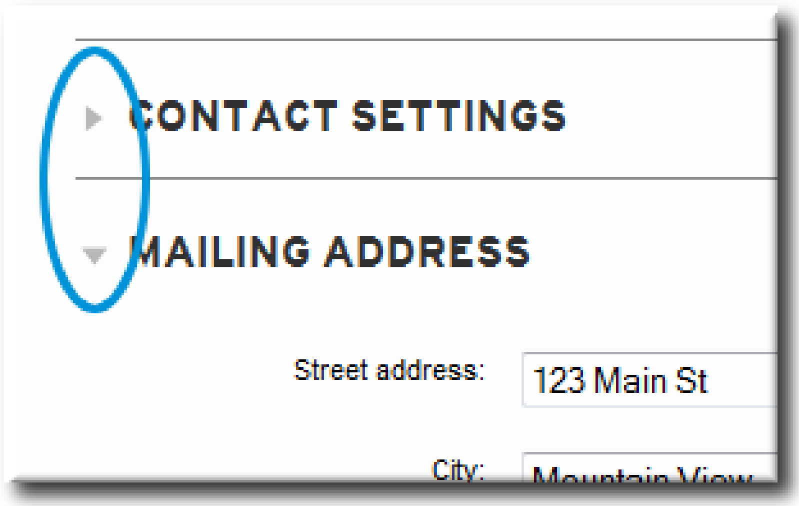


Figure 3.6. Collapsing and Expanding Form Elements

When entries are created, they are added with only the required attributes. Additional attributes can be added manually. Some attributes have default values added to the entry and simply need to be edited; other attributes may not exist at all in the new entry and need to be added.

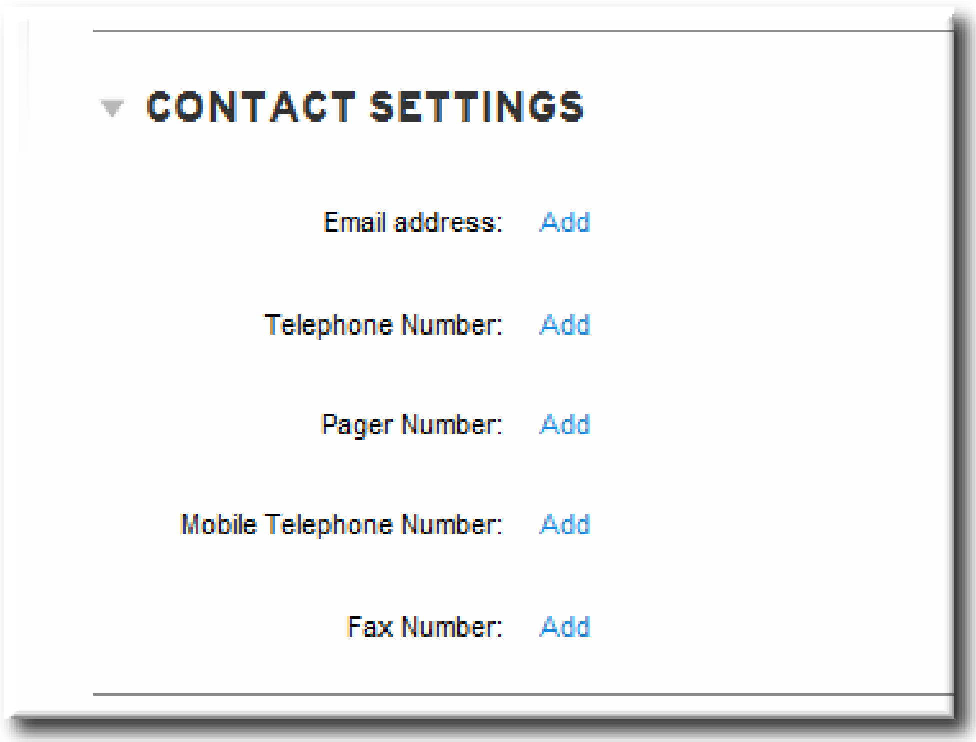
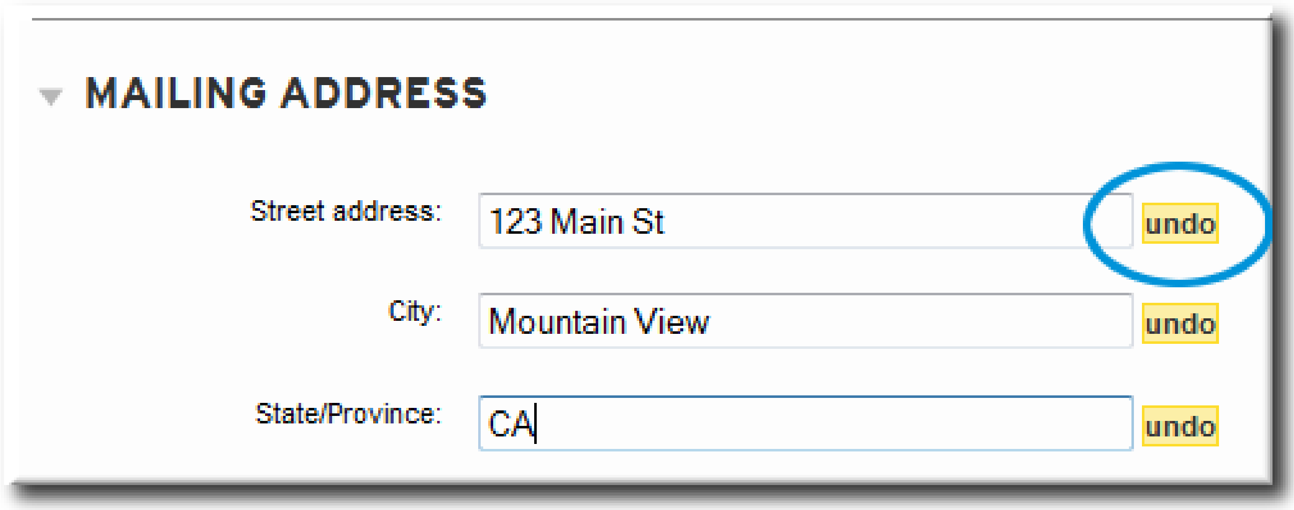


Figure 3.7. Add an Attribute

Any changes to any attribute can be undone. A single attribute change can be undone by clicking the dynamic **undo** button; all changes can be undone by clicking the `Reset` link at the top of the entry details page.



Street address: 123 Main St

City: Mountain View

State/Province: CA

Figure 3.8. Undo Edits

3.1.3. Showing and Changing Group Members

Members can be added to a group through the group configuration. There are tabs for all the member types which can belong to the group, and an administrator picks all of the marching entries and adds them as members.

However, it is also possible for an entity to be added to a group through its own configuration. Each entry has a list of tabs that displays group types that the entry can join. The list of all groups of that type are displayed, and the entity can be added to multiple groups at the same time.

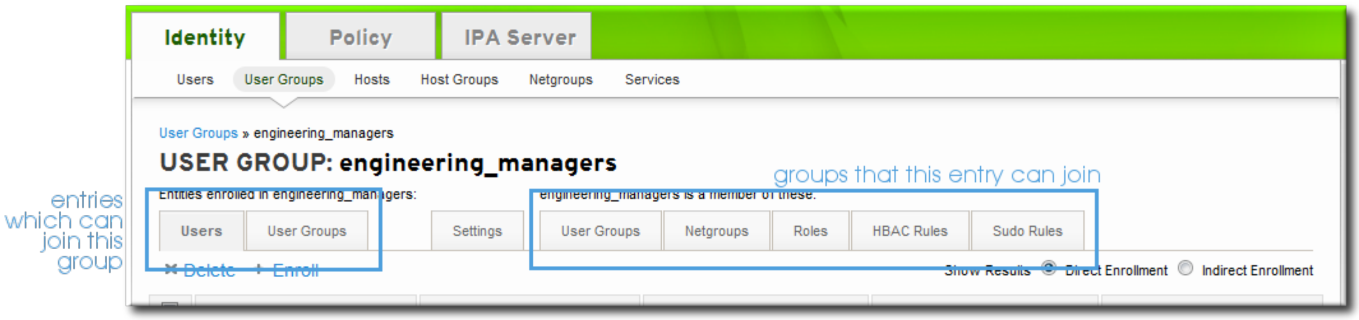

Identity
Policy
IPA Server
Users
User Groups
Hosts
Host Groups
Netgroups
Services
User Groups » engineering_managers
USER GROUP: engineering_managers
entries which can join this group
groups that this entry can join
Entities enrolled in engineering_managers:
Users
User Groups
Settings
engineering_managers is a member of these:
User Groups
Netgroups
Roles
HBAC Rules
Sudo Rules
Delete
Enroll
Show Results
Direct Enrollment
Indirect Enrollment

Figure 3.9. Member Of...

3.1.4. Looking at Search Results

Searches can be performed on attributes that are not displayed in the UI. This means that entries can be returned in a search that do not appear to match the given filter. This is especially common if the search information is very short, which increases the likelihood of a match.

3.2. Logging into IPA

Users are authenticated to IPA services, including the command-line tools and the web UI, using Kerberos authentication. This means that logging into Identity Management requires running **`kinit`**.

Running **kinit** issues the user a Kerberos *ticket*. This ticket is checked by any IPA or Kerberos-aware service, so that a user only needs to log in once to access all domain services. Domain services include the IPA web UI, mounted file shares, wikis, or any other application which uses IPA as its identity/authentication store.

3.2.1. Logging into IPA

Logging into Identity Management requires running **kinit** on a client within the IPA domain.

```
$ kinit
```

The **kinit** command must be run from a machine which has been configured as a client within the IPA domain, so that the client retrieves authenticates with the IPA KDC.

Simply running **kinit** logs into IPA as the currently logged-in user account. This user account must also be an IPA user for them to authenticate to the IPA Kerberos domain successfully. For example, if you are logged into the machine as **jsmith**:

```
$ kinit
Password for jsmith@EXAMPLE.COM:
```

Note

If SSSD or **pam_krb5** is configured on the IPA client machine, then when a user logs into the machine, a ticket is created which can be used for machine services which require authentication, such as **sudo**.

3.2.2. Logging in When an IPA User Is Different Than the System User

To specify an IPA username — because a person's system username is different then the IPA username or to switch IPA user accounts — simply rerun the **kinit** command, specifying the new user. For example:

```
$ kinit userName
Password for userName@EXAMPLE.COM:
```

When the server was first set up, an administrative user, **admin**, is created to perform normal administrative activities. To authenticate as the admin user, use the name admin when running **kinit**:

```
$ kinit admin
```

Note

Only one set of tickets can be stored per logged-in user. The current stored credentials are the ones that will be used when accessing IPA services.

If you were already connected to the IPA web UI as another user, refresh the browser to display the updated details for the new user.

3.2.3. Checking the Current Logged in User

Use the **`klist`** command to verify the identity and the ticket granting ticket (TGT) from the server:

```
$ klist
Ticket cache: FILE:/tmp/krb5cc_500
Default principal: ipaUser@EXAMPLE.COM

Valid starting     Expires            Service principal
11/10/08 15:35:45  11/11/08 15:35:45  krbtgt/EXAMPLE.COM@EXAMPLE.COM

Kerberos 4 ticket cache: /tmp/tkt500
klist: You have no tickets cached
```

It's important to know who the authenticated user is because the currently-authenticated user is the only one who can access the IPA services. The Kerberos client libraries for **`kinit`** have some limitation, one of them being that the current ticket is overwritten with any new invocation of **`kinit`**. Authenticating as User A and then authenticating as User B overwrites User A's ticket.

To allow there to be multiple authenticated users on a machine, set the **`KRB5CCNAME`** environment variable. This variable keeps credential caches separate in different shells.

3.2.4. Caching User Kerberos Tickets

Only one set of tickets can be stored per logged-in user. The current stored credentials are the ones that will be used when accessing IPA services.

For example, if you authenticated as **`admin`**, added a new user, set the password, and then tried to authenticate as that user, the administrator's ticket is lost.

To keep separate credential caches in different shells, a special environment variable, **`KRB5CCNAME`**, can be used.

3.3. Using the IPA Web UI

In order to use the web UI, the user must be authenticated with the IPA Kerberos domain and have an active Kerberos ticket (Section 3.2, “Logging into IPA”). Generally, the web UI can only be accessed from an IPA server or client machine and the user must be locally authenticated. There are a couple of ways to work around this, either by configuring Kerberos on a non-domain machine to connect to the Kerberos domain (Section 3.3.4, “Using a Browser on Another System”) or by enabling password authentication to the UI (Section 3.3.5, “Enabling Username/Password Authentication in Your Browser”).

3.3.1. Supported Web Browsers

The only supported browser to access the IPA web UI is Firefox 3.x or 4.x.

3.3.2. Opening the IPA Web UI

The browser must be properly configured, as described in Section 3.3.3, “Configuring the Browser”, to support Kerberos authentication so that the user can connect to the UI.

To open the web UI:

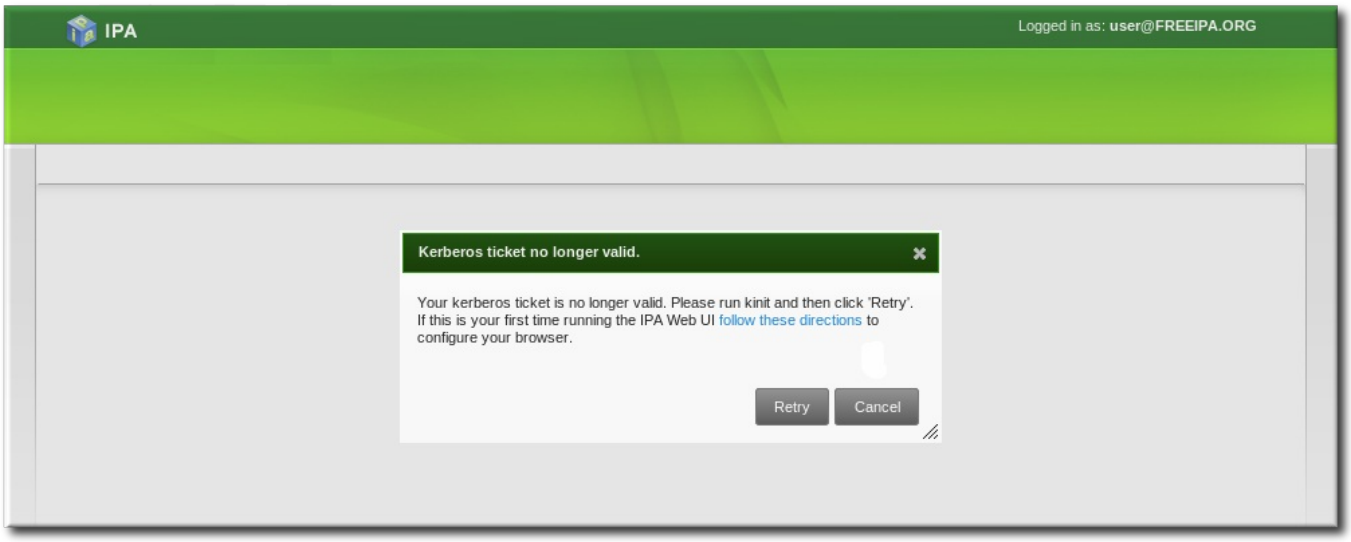
1. Get a valid Kerberos ticket using **`kinit`**, as in Section 3.2, “Logging into IPA”.

2. Open the IPA URL. The full URL is **`https://`***IPAserver-FQDN***`/ipa/ui`**, but this service is also accessed simply by opening **`https://`***IPAserver-FQDN*. For example:

```
https://server.example.com
https://server.example.com/ipa/ui
```

3.3.3. Configuring the Browser

Firefox can use Kerberos credentials to authenticate to the IPA UI, but Kerberos negotiation needs to be configured to use the IPA domain. At the first log-in attempt, if Firefox has not been configured to support Kerberos authentication, then an error message appears.



Kerberos ticket no longer valid.

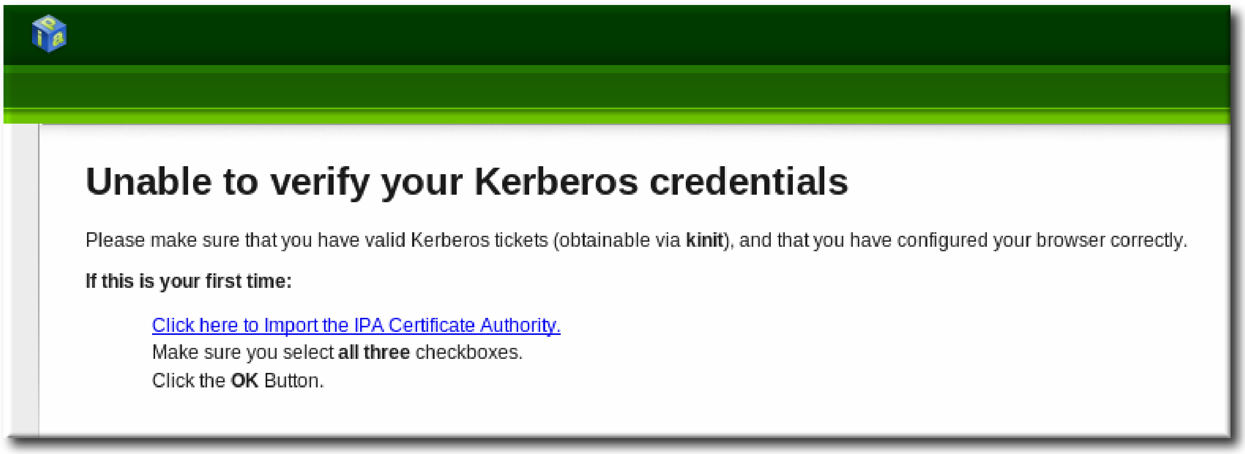
Your kerberos ticket is no longer valid. Please run kinit and then click 'Retry'. If this is your first time running the IPA Web UI follow these directions to configure your browser.

Figure 3.10. Kerberos Authentication Error

If you see that error, then the IPA web UI can perform the required configuration:

1. Click the **`follow these directions`** link.

2. Click the link to import the CA certificate for the IPA server.



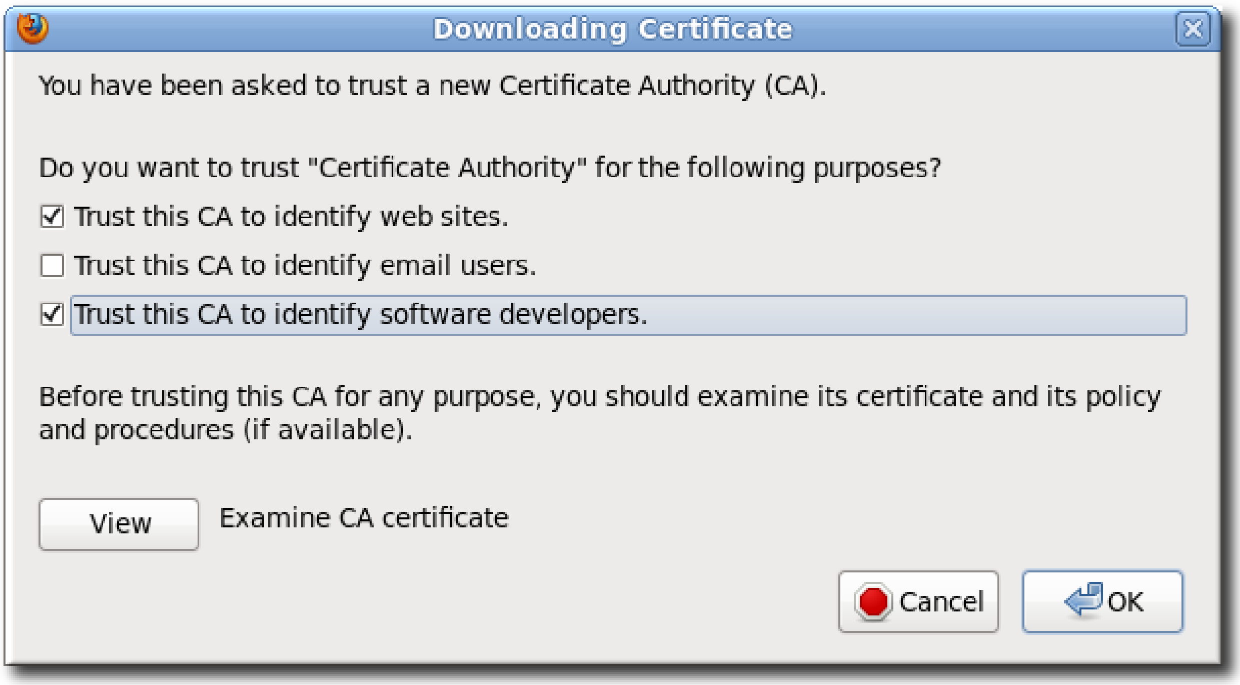
Unable to verify your Kerberos credentials

Please make sure that you have valid Kerberos tickets (obtainable via **kinit**), and that you have configured your browser correctly.

If this is your first time:

Click here to Import the IPA Certificate Authority.
Make sure you select **all three** checkboxes.
Click the **OK** Button.

3. Set the web site and software developer (first and last) trust bits for the CA certificate.


Downloading Certificate

You have been asked to trust a new Certificate Authority (CA).

Do you want to trust "Certificate Authority" for the following purposes?

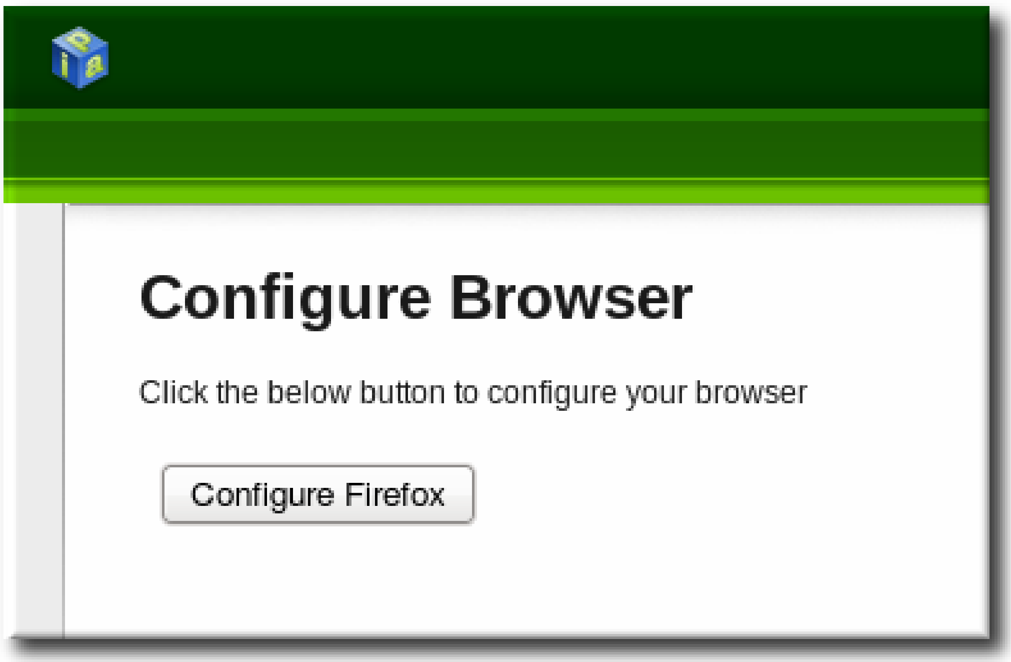
☑ Trust this CA to identify web sites.

☐ Trust this CA to identify email users.

☑ Trust this CA to identify software developers.

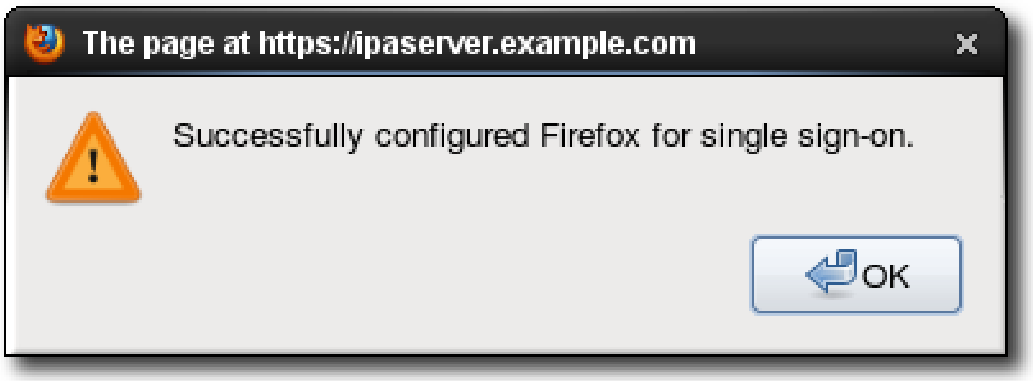
Before trusting this CA for any purpose, you should examine its certificate and its policy and procedures (if available).

4. Click the **`Configure Firefox`** button. This automatically fills out all the **`negotiate`** settings in the Firefox configuration to use the IPA domain settings.


Configure Browser

Click the below button to configure your browser

When the process is complete, a success box pops up saying that Firefox has been configured for single sign-on. For there, you are redirected to the IPA web UI.


The page at https://ipaserver.example.com

Successfully configured Firefox for single sign-on.

This can also be done manually:

1. Open Firefox.

2. Type **`about:config`** in the address bar.

3. In the **`Search`** field, type **`negotiate`** to filter out the Kerberos-related parameters.

4. On Red Hat Enterprise Linux, enter the domain name for the URI parameters, including the preceding period (.) and set the **`gsslib`** parameter to true:

   ```
   network.negotiate-auth.trusted-uris  .example.com
   network.negotiate-auth.delegation-uris  .example.com
   network.negotiate-auth.using-native-gsslib true
   ```

5. Open the web UI by going to the fully-qualified domain name of the IPA server such as **`http://ipaserver.example.com`**. Make sure that you can open the web UI and that there are nc Kerberos authentication errors.

6. Next, download the IPA server's CA certificate from **`http://ipa.example.com/ipa/config/ca.crt`**.

7. Select the first (**`Trust this CA to identify web sites`**) and third (**`Trust this CA to identify software developers`**) check boxes.

3.3.4. Using a Browser on Another System

It is possible to connect to the Identity Management web UI from a system which is *not* a member of the IPA domain. In this case, it is possible to specify an IPA-specific Kerberos configuration file on the external (non-IPA) machine before running **`kinit`**, and then the user can authenticate against the IPA server domain.

This is especially useful there are multiple realms or overlapping domains across your infrastructure.

1. Copy the **`/etc/krb5.conf`** file from the IPA server.

   ```
   # scp /etc/krb5.conf
   root@externalmachine.example.com:/etc/krb5_ipa.conf
   ```

   Warning

   Do not overwrite the existing **`krb5.conf`** file.

2. On the external machine, set the terminal session to use the copied IPA Kerberos configuration file:

```
$ export KRB5_CONFIG=/etc/krb5_ipa.conf
```

3. Configure Firefox on the external machine as in Section 3.3.3, “Configuring the Browser”.

3.3.5. Enabling Username/Password Authentication in Your Browser

If Kerberos authentication fails, then browser login also fails. That prevents access to the IPA web UI. Configuring username/password authentication for the UI allows users to log in even if there are problems with the Kerberos service.

1. Open the **ipa.conf** file used by the Apache web service.

```
vim /etc/httpd/conf.d/ipa.conf
```

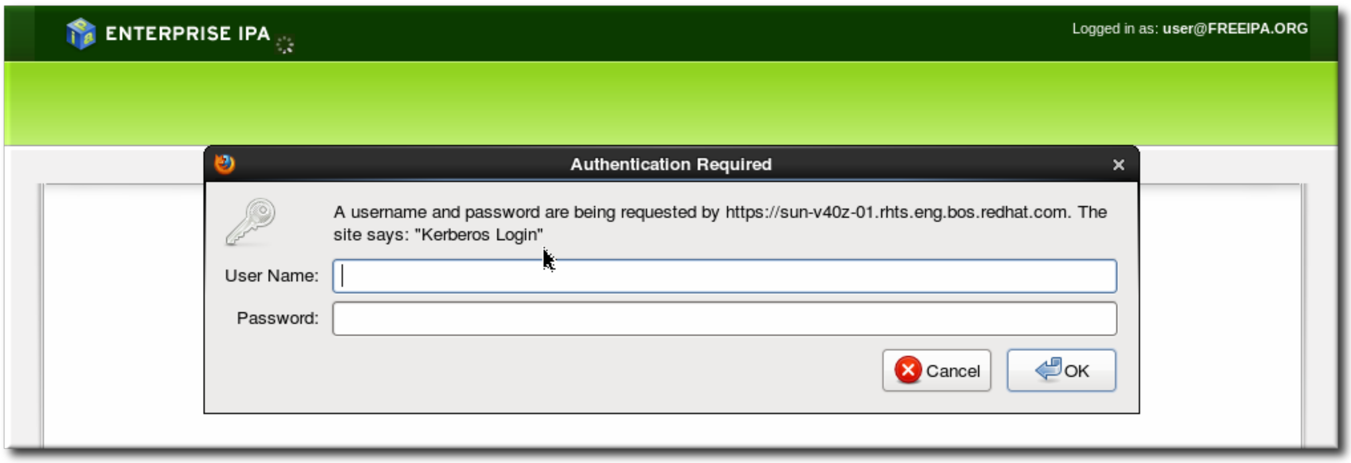
2. In the **<Location "/ipa">** location definition, change the ***KrbMethodK5Passwd*** attribute from off to on.

```
KrbMethodK5Passwd on
```

3. Restart the **httpd** service:

```
# service httpd restart
```

When this is configured, the web UI prompts for an IPA username and password if it can't detect any Kerberos credentials.



A username and password are being requested by https://sun-v40z-01.rhts.eng.bos.redhat.com. The site says: "Kerberos Login"

Figure 3.11. IPA Password Prompt

Note

This must be configured on every IPA server in the domain.

3.3.6. Using the UI with Proxy Servers

Proxy servers can be used to access the web UI without any additional configuration in IPA.

Port forwarding is not supported with the IPA server. However, because it is possible to use proxy servers with IPA, an operation similar to port forwarding can be configured using proxy forwarding with OpenSSH and the SOCKS option. This is described in http://www.meadowy.org/~gotoh/ssh/openssh-socks.html.

3.3.7. Troubleshooting UI Connection Problems

If negotiate authentication is not working, turn on verbose logging for the authentication process to help diagnose the issue:

1. Close all browser windows.
2. In a terminal, set the new log levels for Firefox:

   ```
   export NSPR_LOG_MODULES=negotiateauth:5
   export NSPR_LOG_FILE=/tmp/moz.log
   ```

   This enables verbose logging and logs all information to **`/tmp/moz.log`**.
3. Restart the browser from the same terminal window and attempt t .

Some of the common error messages and workarounds are in Table 3.2, “UI Error Log Messages”.

Table 3.2. UI Error Log Messages

Error Log Message	Description and Fix

`-1208550944[90039d0]: entering nsNegotiateAuth::GetNextToken()`
`-1208550944[90039d0]: gss_init_sec_context() failed: Miscellaneous failure`
`No credentials cache found`
	There are no Kerberos tickets. Run **`kinit`**.

`-1208994096[8d683d8]: entering nsAuthGSSAPI::GetNextToken()`
`-1208994096[8d683d8]: gss_init_sec_context() failed: Miscellaneous failure`
`Server not found in Kerberos database`
	This can occur when you have successfully obtained Kerberos tickets but are still unable to authenticate to the UI. This indicates that there is a problem with the Kerberos configuration. The first place to check is the **`[domain_realm]`** section in the **`/etc/krb5.conf`** file. Make sure that the IPA Kerberos domain entry is correct and matches the configuration in the Firefox negotiation parameters. For example:
	`.example.com = EXAMPLE.COM`
	`example.com = EXAMPLE.COM`

Nothing is in the log file.	It is possible that you are behind a proxy which is removing the HTTP headers required for negotiate authentication. Try to connect to the server using HTTPS instead, which allows the request to pass through unmodified. Then check the log file again.

Chapter 4. Configuring Red Hat Enterprise Linux 5 Servers for IPA Domain Services

4.1. Client Configuration for sudo Rules

This example specifically configures a Red Hat Enterprise Linux 5 client for sudo rules. The configuration file in step 4 is different, depending on the platform.

1. Configure **sudo** to look to LDAP for the **sudoers** file.

   ```
   vim /etc/nsswitch.conf

   sudoers:  files ldap
   ```

   Leaving the **files** option in place allows **sudo** to check its local configuration before checking the LDAP-based IPA configuration.

2. Enable debug logging for **sudo** operations in the **/etc/ldap.conf** file. If this file does not exist, it can be created.

   ```
   vim /etc/ldap.conf

   sudoers_debug: 1
   ```

   Note

   Adding the ***sudoers_debug*** parameter helps with troubleshooting. Valid values for this parameter are 0,http://jboss-on-docs.etherpad.corp.redhat.com/10 1, and 2. The **sudo** documentation at http://www.gratisoft.us/sudo/readme_ldap.html has more information on debugging the process.

3. Optionally, enable debugging in SSSD to show what LDAP settings it is using.

   ```
   vim /etc/sssd/sssd.conf

   [domain/IPADOMAIN]
   debug_level = 6
   ....
   ```

   The LDAP search base used by SSSD for operations is recorded in the **sssd_*DOMAINNAME*.log** log.

4. Edit the NSS/LDAP configuration file and add the following **sudo**-related lines to the **/etc/nss_ldap.conf** file:

   ```
   sudoers_base ou=SUDOers,dc=example,dc=com
   binddn uid=sudo,cn=sysaccounts,cn=etc,dc=example,dc=com
   bindpw sudo_password
   ssl start_tls
   tls_cacertfile /etc/ipa/ca.crt
   ```

```
tls_checkpeer yes
bind_timelimit 5
timelimit 15
uri ldap://ipaserver.example.com ldap://backup.example.com:3890
```

Multiple LDAP servers can be configured in a space-separated list, and other options (like SSL and non-standard ports) can be used with the LDAP URL. The **sudo** LDAP configuration is covered in the **sudo** manpages, http://www.sudo.ws/sudo/man/1.8.2/sudoers.ldap.man.html.

Important

The ***`uri`*** directive must give the fully-qualified domain name of the LDAP server, not an IP address. Otherwise, **sudo** fails to connect to the LDAP server.

5. Create a symlink between the **`nss_ldap`** module configuration file and the system LDAP configuration file:

   ```
   # ln -s /etc/nss_ldap.conf /etc/ldap.conf
   ```

6. Set a name for the NIS domain in the **sudo** configuration. **sudo** uses NIS netgroups, so the NIS domain name must be set in the system configuration for **sudo** to be able to find the host groups used in the IPA **sudo** configuration.

   a. Open the **`/etc/rc.d/rc.local`** file. Setting the NIS domain name in this file allows the value to persist between reboots.

      ```
      # vim /etc/rc.d/rc.local
      ```

   b. Add the command to set the NIS domain name.

      ```
      nisdomainname example.com
      ```

Important

Even though **sudo** uses NIS-style netgroups, **it is not necessary to have a NIS server installed**. Netgroups require that a NIS domain be named in their configuration, so **sudo** requires that a NIS domain be named for netgroups. However, that NIS domain does not actually need to exist.

4.2. Configuring autofs

1. Edit the **`/etc/sysconfig/autofs`** file to specify the schema attributes that autofs searches for:

   ```
   #
   # Other common LDAP naming
   #
   MAP_OBJECT_CLASS="automountMap"
   ```

```
ENTRY_OBJECT_CLASS="automount"
MAP_ATTRIBUTE="automountMapName"
ENTRY_ATTRIBUTE="automountKey"
VALUE_ATTRIBUTE="automountInformation"
```

2. Specify the LDAP configuration. There are two ways to do this. The simplest is to let the automount service discover the LDAP server and locations on its own:

   ```
   LDAP_URI="ldap:///dc=example,dc=com"
   ```

   Alternatively, explicitly set which LDAP server to use and the base DN for LDAP searches:

   ```
   LDAP_URI="ldap://ipa.example.com"
   SEARCH_BASE="cn=location,cn=automount,dc=example,dc=com"
   ```

   Note

   The default value for *location* is **`default`**. If additional locations are added, then the client can be pointed to use those locations, instead.

3. Edit the **`/etc/autofs_ldap_auth.conf`** file so that autofs allows client authentication with the IPA LDAP server. Change ***`authrequired`*** to yes and set the principal to the Kerberos host principal:

   ```
   <autofs_ldap_sasl_conf
        usetls="no"
        tlsrequired="no"
        authrequired="yes"
        authtype="GSSAPI"
        clientprinc="host/server.example.com@EXAMPLE COM"
        />
   ```

   If necessary, run **`klist -k`** to get the exact host principal information.

4. Check the **`/etc/nssswitch.conf`** file, so that LDAP is listed as a source for automount configuration:

   ```
   automount: files ldap
   ```

5. Restart autofs:

   ```
   # service autofs restart
   ```

6. Test the configuration by listing a user's **`/home`** directory:

   ```
   # ls /home/userName
   ```

   If this does not mount the remote file system, check the **`/var/log/messages`** file for errors. If necessary, increase the debug level in the **`/etc/sysconfig/autofs`** file by setting the ***`LOGGING`*** parameter to **debug**.

Note

If there are problems with automount, then cross-reference the automount attempts with the 389 Directory Server access logs for the IPA instance, which will show the attempted access, user, and search base.

It is also simple to run automount in the foreground with debug logging on.

```
automount -f -d
```

This prints the debug log information directly, without having to cross-check the LDAP access log with automount's log.

4.3. Setting up NFS

4.3.1. Setting up a Kerberized NFS Server

1. If the NFS host machine has not been added as a client to the IPA domain, then create the host entry.

2. Create the NFS service entry in the IPA domain.

3. Generate an NFS service keytab for the NFS server. If this command is run on the NFS server, then save the keys directly to the host keytab. For example:

   ```
   # ipa-getkeytab -s server.example.com -p nfs/nfs-server.example.com -k
   /etc/krb5.keytab -e des-cbc-crc
   ```

   Note

   Only DES keys are supported on Red Hat Enterprise Linux 5.

   If this command is run on a different machine:

   a. Save the keytab to a temporary file. For example:

      ```
      ... -k /tmp/nfs.keytab
      ```

   b. Copy the keytabs over to the NFS machine.

   c. Set the file permissions to 0700.

   d. On the NFS host machine, add the service key to the keytab file.

      ```
      #  (  echo rkt /tmp/nfs.keytab; echo wkt /etc/krb5.keytab)
      |ktutil
      ```

4. Install the NFS packages. For example:

```
# yum install nfs-utils
```

5. Edit the **krb5.conf** file to allow weak crypto. This is required for every NFS client if *any* client in the domain will use older encryption options like DES.

   ```
   # vim /etc/krb5.conf

   allow_weak_crypto = true
   ```

6. Edit the NFS server configuration to use NFSv4 security by uncommenting the **SECURE_NFS** line.

   ```
   # vim /etc/sysconfig/nfs

   SECURE_NFS="yes"
   ```

7. If the NFS server and client are in different DNS domains, then configure the NFS domain.

   ```
   # vim /etc/idmapd.conf

   Domain = example.com
   ```

8. Edit the **/etc/exports** file and add the Kerberos information:

   ```
   /export  *(rw,sec=sys:krb5:krb5i:krb5p)
   ```

9. Restart the NFS server.

   ```
   # service nfs restart
   ```

10. Configure the NFS server as an NFS client, following the directions in Section 4.3.2, “Setting up a Kerberized NFS Client”.

4.3.2. Setting up a Kerberized NFS Client

1. If the NFS client is not enrolled as a client in the IPA domain, then set up the required host entries.

   a. Create the host entry.

   b. Generate host keytab for the NFS client. If this command is run on the NFS clien, then save the keys directly to the host keytab. For example:

      ```
      # ipa-getkeytab -p host/nfs-client-
      server.example.com@EXAMPLE.COM -k /etc/krb5.keytab -e des-cbc-
      crc
      ```

      Note

      Only DES keys are supported on Red Hat Enterprise Linux 5.

      If this command is run on a different machine:

a. Save the keytab to a temporary file. For example:

```
... -k /tmp/nfs.keytab
```

b. Copy the keytabs over to the NFS machine.

c. Set the file permissions to 0700.

d. On the NFS host machine, add the service key to the keytab file.

```
# ( echo rkt /root/nfs-client.keytab; echo wkt
/etc/krb5.keytab) |ktutil
```

2. Edit the **krb5.conf** file to allow weak crypto. This is required for every client if *any* client in the domain will use older encryption options like DES.

```
# vim /etc/krb5.conf

allow_weak_crypto = true
```

3. Edit the NFS common configuration to enable client-side secure NFS, by uncommenting the **SECURE_NFS** line.

```
# vim /etc/sysconfig/nfs

SECURE_NFS="yes"
```

4. If the NFS server and client are in different DNS domains, then configure the NFS domain. The **idmapd.conf** must be the same on the NFS client as it is on the NFS server.

```
# vim /etc/idmapd.conf

Domain = example.com
```

5. Start the GSS daemon.

```
# service rpc.gssd start
```

6. Mount the directory.

```
# echo "$NFSSERVER:/this /mnt/this nfs4
sec=krb5i,rw,proto=tcp,port=2049"  >>/etc/fstab
# mount -av
```

Index