Identity Management Consulting Field Guide

Jason Ritenour

description

Document Information

Originator

Jason Ritenour - Red Hat Consulting

Dan Lavu - Red Hat Quality Engineering

Owner

Red Hat Consulting – Confidential. Restricted distribution.

At this point in time, this document should be considered **internal use only**.

Copyright

Distribution

The source asciidoc files can be obtained at: https://github.com/rhtconsulting/idm_guide

Intro to Red Hat Identity Management

About the Red Hat Consulting IdM Field Guide

What it is: This guide is a collaborative effort between individuals within Red Hat Consulting and Engineering resources to seed basic knowledge of Identity Management throughout the consulting organization. As IdM's feature set expands, we are seeeing more and more interest from customers in deploying IdM as a method of providing centralized access to RHEL hosts in their environment. Scoping & designing an IdM environment is not exactly common knowledge - we want to help expand the field of consultants and and architects who are able to do so. In addition, this guide will also act as a field manual for installation, configuration, integration, and basic troubleshooting that a consultant may encounter during an IdM deployment.

What it is not: This document is not intended to be an authoritative technical reference for IdM, or any of its components, or replace any official form of documentation or training. As this is geared toward consultants, it is not intended to be shared with customers at this itme.

What Do I Need To Know To Deploy IdM?

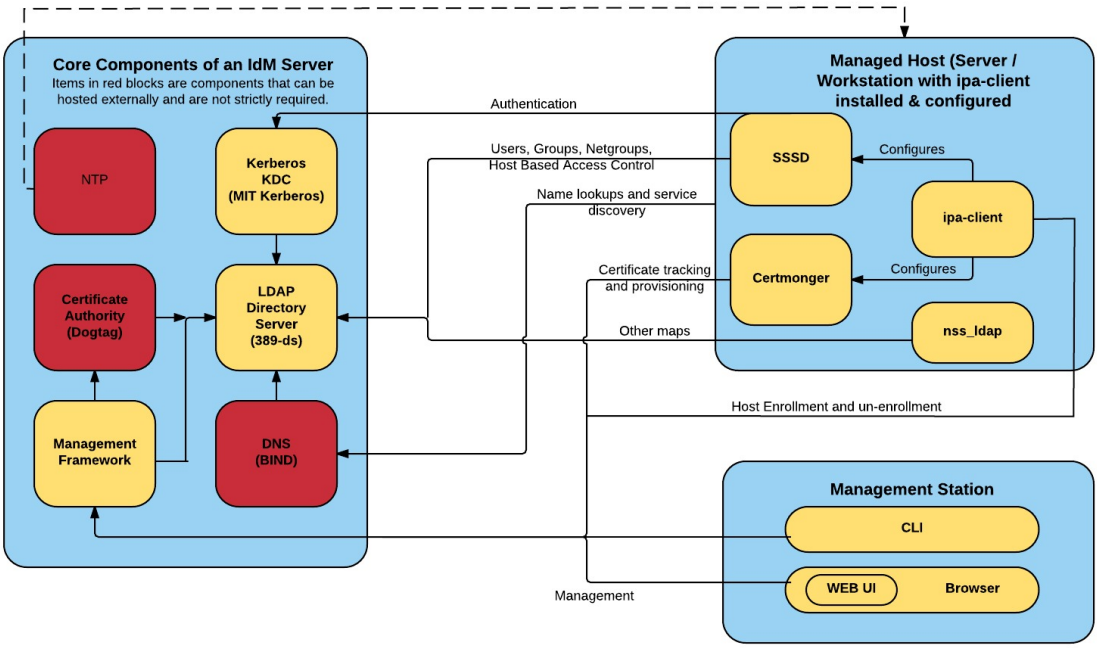
IdM consists of several upstream projects:

- NTP
- MIT Kerberos
- Dogtag (Certificate Authority)
- 389-ds (LDAP Directory Server)
- bind (DNS)
- SSSD

You should have at least a basic understanding of what each of those components do and how they inter-operate to deliver a basic IdM engagement. For example, you don't necessarily need to be PKI expert, but you should understand the implications of using a self-signed cert in IdM versus having a certificate signed by an external (presumably trusted) CA. And you shoud understand what a CA is.

Below is a diagram of the IdM components, and how they work together at a high level.

IdM Product Architecture



Note: This diagram is based on and slightly modified from the a slide deck previously created by Dmitri Pal. I created this diagram from to have it an editable format for incubation team usage.

Scoping & Design

Approach to Scoping

IdM can be either very simple or extremely difficult to implement depending on the specifics of the customer's use case. Therefore, it is absolutely essential to properly scope the deployment. To properly scope the engagement, you need to be aware of the following:

- Why does the customer want to use IdM? (basic LDAP enabled directory, migration from another directory service, back end authentication source for an app, etc.)
- Will IdM need to interface with Active Directory in any way? If so, what specific functionality is required?
- Where will IdM's infrastructure components (DNS, PKI, NTP) fit into their current infrastructure?
- Will they be using any advanced forms of authentication, such as Smart Cards or OTP?
- How many hosts & users?
- Are they planning to leverage HBAC/sudo/selinux rules in IdM?

This is just a high level sample, but getting the answers to questions such as these can mean the difference between a two or six week engagement.

Scoping Questions

Current Customer Environment

- Do you have an existing directory service? Will IdM be integrating with this or replacing it?
  - Do you currently have a Microsoft Active Directory (AD) server in place?
  - If so, is there a requirement to integrate with the existing AD infrastructure?
- Do you have an existing DNS service? Will IdM be integrating with this or replacing it?
  - Will DNS be delegated to IdM for specific domains?
- Do you have an existing certificate authority/private key infrastructure? Will IdM be integrating with this or replacing it?
- Do you have a RHEL 7 baseline/is RHEL 7 approved for usage in your environment?

NOTE	IdM is indeed present in RHEL 6, it is based on the older 3.x version of FreeIPA. There will likely be no major additions to this version, and new features such as smart card auth & OTP will likely never be added to it.

Workload and Configuration

- How many users/hosts will be joined to IdM?

- How many sites will IdM be servicing? - Note: to keep this question simple, let's agree that a "site" is anything separated by a WAN link.
- How do you plan to authenticate to IdM?
  - LDAP
  - Kerberos
  - SSSD
  - HTTP/SAML
  - User certificates
- Do you need to import/re-create any local user accounts from existing systems?
- What client operating systems/versions will need to authenticate to IdM?
- What Red Hat products will use IdM for authentication?
  - CloudForms
  - RHEV
  - Satellite
  - OpenStack
  - OpenShift
- What non-Red Hat productions will use IdM for authentication?
- IdM provides the ability for clients to supplement the local sudo configurations with information stored and maintained within the directory server. Do you currently control local sudoers file?
- If so, have you identified the services users can have access to through the sudo command?
- What are your current Security Policy requirements around password length, complexity, aging, login failure attempts, and auditing?
- IdM can control access to both machines and services running on those machines. Are there any requirements to allow/deny access to certain machines for a particular service? Example would be no ssh to a particular machine for the users group but allowed for the unix_admin group.

Planning Approach

Once you have a better idea of what the customer wants, you can start planning the deployment. A few things to keep in mind right from the beginning:

- Less is more - a single IdM instance can typically serve ~2500-3000 clients without issue
- That said, you sould always go with two at a minimum, for availability purposes
- Things are much smoother if you can get the Client to configure DNS zones/delegations and such ahead of time
- Joining under an existing trusted CA is a bit of a pain and slows the deployment down

(generating CSR, submitting it, waiting for & integrating signed certificate) but if customer has existing CA in place, this is preferred. It's also easier to do up front rather than change to a external CA from a self signed cert later on.

- AD Trusts can be painful, more from a cultural than technical standpoint - Windows admins are usually leary of other directory services.

Sample IdM SOW For "Base Configuration" (Two Week Engagement)

NOTE	The basis for this SOW began with the one the IdM SME community came up with at https://mojo.redhat.com/docs/DOC-1046014. Has been adapted & expanded upon based one what clients typically ask for in my experience with the public sector. YMMV.

Scope

- Install two (2 IdM Servers in one location)
- Only RHEL 6/7 Clients will be joined
- Create sample HBAC/sudo/automount rules
- Configure one (1) form of multifactor authentication
- Configure one (1) currently deployed Red Hat application (ie Satellite, CloudForms, OpenStack) to perform LDAP authentication using IdM as the provider
- Integration with AD is out of scope. Add 1-2 weeks for that.
- Integration with existing DNS/NTP/CA services, provided Client team is available & able to quickly turn around requests for integration (ie signing CSRs, creating/delegating DNS zones)
- DNSSEC is not in scope. If client insists add a week to timeline.

SOW Task list

- Assist Client in planning a Red Hat Identity Management (IdM) Topology consisting of:
  - Two (2) IdM servers.
  - Design drawing depicting physical & logical location of servers in relation to clients and other infrastructure components
  - Capacity planning aligned to expected grow and feature usage
- Assist Client in the deployment of the following:
  - Two (2) IdM servers
  - Installation & configuration of RHEL on the IdM hosts
  - Subscribing systems to the approriate RHN/Satellite channels
  - IdM package installation and requirements including DNS guidance.
  - LDAP authorization and authentication policies.

    - Configure and review example sudo, automount and HBAC configurations, with validation against a single RHEL6 and/or a single RHEL7 client.
    - Configure and review mult-factor authentication, with validation against a single user signing on to a single host.

- Review with the Client general operations and maintenance procedures for IdM including:
    - LDAP backup and restore procedures.
    - Monitoring and troubleshooting
        - Demonstration of dbmon.sh (memory/cache utilization)
        - Demonstration of logconv.pl (problems in the access log e.g. need indexes, misconfigured clients, etc.)
        - Demonstration of replication monitoring and troubleshooting
    - Demonstrate registration client to IdM server for authentication, using either admin credentials or a pre-created host object's one time password.
    - General IdM maintenance and tuning (sysctl parameters, configuration replication aggreements, etc).
    - Creation of example users, groups, host groups, and password policies.
    - IdM configuration and customization.
    - IdM CLI usage and examples.
    - IdM Web Interface usage and examples.
    - Adding additional IdM hosts/decomming existing hosts

Installing & Configuring IdM

Installation

Before You Get Started

Before you begin installing IdM, be sure you know what your initial topology is going to look like, and how replication is going to flow if you have more than 4 IdM servers.

You should also know what version of RHEL you are planning on deploying on, and ensure the features you need are supported in that version. For example, smart card authentication is only supported in RHEL 7.2 and later (though it is currently planned to be added client side in RHEL 6.8).

As far as hardware requirements, as a general rule of thumb, you can follow the minimum requirements of RHEL 7, but remember that a large ldap directory will require more resources. Official IdM documentation recommends the following:

- For 10,000 users, have 2GB RAM with 1GB of swap space.
- For 100,000 users, have 16GB of RAM and 4GB of swap space.

In general, the larger the deployment, the more RAM you'll want. IdM caches a lot of information.

Generally speaking, you want IdM to have it's own DNS namespace, and where possible, you should let IdM control that namespace. You do not, for example want IdM to be sharing a subdomain with Active Directory, and you especially don't want them to both be creating DNS entries in a subdomain. You also want the primary DNS domain name to match up with the Kerberos realm name. For example, a dns domain name of ipa.redhat.com would have a Kerberos realm name of IPA.REDHAT.COM - the same name in all caps. Deviating from this convention will cause a lot of headaches.

Prior to install, it's a good idea to go ahead and open the necessary ports in any host based or network based firewalls:

- 80/443 TCP for access to the Web UI/api
- 389/636 TCP for LDAP/LDAPS
- 84,464 TCP and UDP for Kerberos
- 53 TCP and UDP for DNS
- 123 UDP for NTP

There are additional ports you may need to open for specific configurations (such as an Active Directory trust) but those will be covered in their relevant sections.

Regardless of whether IdM is going to be hosting the NTP service or not, you should make sure you have time synchronized in your environment. Kerberos has a very low tolerance for discrepancies in time, at 5 minutes by default. Though you can configure a different threshold, this negates a lot of the security of Kerberos as a method to avoid main in the middle/replay attacks.

You should plan which IPA server is going to be the first master. Though IPA is a multi-master service, only one server can host the Certificate Authority's certificate revocation list (CRL.) This can be manually moved at a later time, but it's good to keep in mind that if you will be issuing certificates in IdM, hosts will need to be able to contact that server to validate certs.

Finally on that note, you need to know how you will be configuring the internal IdM Certificate Authority - whether you will be using a self-signed certificate, one from an external CA, or if you will be configuring IdM to not issue certificates at all. Note that a "CA-less" install will still require supplying certificates for the Web UI & for LDAPS.

Preparing To Install

The only required channel for IdM is the base RHEL channel, ie rhel-7-server-rpms (or rhel-6-server-rpms if using RHEL 6).

Once properly registered through RHN/subscription manager, install the ipa-server and ipa-server-dns (if IdM will be managing DNS)

```
yum install ipa-server ipa-server-dns
```

This installs all the services necessary for IdM (ie named, Kerberos, 389-ds) along with all dependencies.

After the packages are installed, the next step is to run the ipa-client-install script on the first IdM master.

The ipa-server-install Script

The ipa-server-install script is a python script that acts a front end for configuring IdM. If you do not specify any flags, it will ask you several questions in order to configure the environment, and list the defaults in brackets. For example, the first question you will see is:

```
Do you want to configure integrated DNS BIND)? [no]:
```

You'll then be asked to confirm the server's host name, the domain/realm name you'll be using (which defaults to using whatever the FQDN of the server is), and specify passwords for the Directory Manager & admin users. Once you've confirmed all the setup options, IdM will begin it's configuration process, which takes a few minutes. You will see feedback on specific steps on screen.

```
The following operations may take some minutes to complete.
Please wait until the prompt is returned.

Configuring NTP daemon (ntpd)
  [1/4]: stopping ntpd
  [2/4]: writing configuration
  [3/4]: configuring ntpd to start on boot
  [4/4]: starting ntpd
Done configuring NTP daemon (ntpd).
Configuring directory server (dirsrv): Estimated time 1 minute
  [1/38]: creating directory server user

...

Restarting the directory server
Restarting the KDC
Restarting the certificate server
```

NOTE	If installing in a VM, you may see a warning stating the system is running out of entropy, and you may experience delays. Generally, I have found the best way to mitigate this is to install & run the havaged service, though this is only available in the EPEL repoistory in RHEL/CentOS.

Though you can install/configure IdM without specifying any flags, there are a handful that I will explicitly point out:

- --setup-dns: This instructs IdM to setup integrated DNS and allow dynamic updates by default.
- --external-ca: This will instruct IdM to generate a CSR to pass to an external certificate authority in order to create a certificate. The installation process will stop at that point, and can be resumed once the signed certificate is in place. See next section for more information.
- --forwarder=**ip address**: This allows you to point named to another DNS server for any domains IdM is not authoritative for, so that all client machines can use IdM (and only IdM) as its source for DNS.
- --idstart: This allows you to set the starting number for UID & GID in IdM. By defaul it selects a random number beyond 100,000.

NOTE	If you want to perform a silent install, you need to supply a minimum the realm name (-r EXAMPLE.COM), the Directory manager password (-p password), the admin user password (-a password) and the unattended flag (-U).

Certificate Authority Options

External Certificate Authority

By default, ipa-server-install will setup an internal CA using a self-signed certificate. However, if the customer has an existing Certificate Authority in place, they may want IdM to be a subordinate

CA under that hierarchy, or they may not want IdM to issue certificates at all. This section will cover the options for configuring the CA.

NOTE	If IdM is configured as a CA, it will automatically add itself to the trusted CA store on all clients. This is one benefit to having an integrated CA.

You can configure IdM to generate a CSR by running **ipa-client-install** with the **--external-ca** flag. This will generate an **ipa.csr** file under your home directory. You can then take that to an external CA either in house, or from a 3rd party certificate vendor, and generate a certificate for the IdM server.

Then, you'll take the signed IdM certificate, along with either the external CA's root certificate, or an intermediate certificate and re-run **ipa-server-install** with the **--external-cert-file** flags - two instances, one for the IdM cert, and once for the CA/intermediate cert. Setup will resume from the CA configuration steps, and complete as normal.

Installing Without a CA

Though here is no flag to not configure a CA in ipa-server-install, if you explicitly supply a certificate and key for both Apache and LDAP, along with the full certificate chain of the CA that issued those certificate, you can effectively install IdM without a configured certificate authority. There are a few limitations to be aware of, however.

- There will be no expiration warning for either the Apache or LDAP certificates inside of IdM, as certmonger will not be tracking them.
- It will not be possible to renew these certificates through IdM.
- ipa cert-* tools cannot be used to view/manage certificates.
- All host/service certificates must be manually requested, generated, and uploaded. Will will affect ipa host-add and other management tools.
- Any removed certificates are not automatically revoked.

To install without a CA, use the following flags.

```
ipa-server-install --http-cert-file /tmp/server.crt --http-cert-file /tmp/server.key
--http-pin secret --dirsrv-cert-file /tmp/server.crt --dirsrv-cert-file
/tmp/server.key --dirsrv-pin secret --ca-cert-file ca.crt
```

Verifying Install

After completion, there are several steps you can take to verify your IdM server is functioning as expected.

Verify you can acquire a Kerberos ticket from the IdM server's cli with kinit:

```
[root@jr-idm1 ~]# kinit admin
Password for admin@JMR.LAB:
[root@jr-idm1 ~]# klist
Ticket cache: KEYRING:persistent:0:0
Default principal: admin@JMR.LAB

Valid starting       Expires              Service principal
02/18/2016 13:10:31  02/19/2016 13:10:29  krbtgt/JMR.LAB@JMR.LAB
```

You can use either **systemctl** or **service** to check the status of the entire IPA stack.

```
[root@jr-idm1 ~]# systemctl status ipa
● ipa.service - Identity, Policy, Audit
   Loaded: loaded (/usr/lib/systemd/system/ipa.service; enabled; vendor preset:
disabled)
   Active: active (exited) since Tue 2016-02-16 10:59:08 EST; 2 days ago
  Process: 74949 ExecStop=/usr/sbin/ipactl stop (code=exited, status=0/SUCCESS)
  Process: 75034 ExecStart=/usr/sbin/ipactl start (code=exited, status=0/SUCCESS)
 Main PID: 75034 (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/ipa.service

Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting Directory Service
Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting krb5kdc Service
Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting kadmin Service
Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting named Service
Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting ipa_memcached Service
Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting httpd Service
Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting pki-tomcatd Service
Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting ipa-otpd Service
Feb 16 10:59:08 jr-idm1.jmr.lab ipactl[75034]: Starting ipa-dnskeysyncd Service
Feb 16 10:59:08 jr-idm1.jmr.lab systemd[1]: Started Identity, Policy, Audit.
```

For a more concise view, you can use **ipactl**

```
[root@jr-idm1 ~]# ipactl status
Directory Service: RUNNING
krb5kdc Service: RUNNING
kadmin Service: RUNNING
named Service: RUNNING
ipa_memcached Service: RUNNING
httpd Service: RUNNING
pki-tomcatd Service: RUNNING
ipa-otpd Service: RUNNING
ipa-dnskeysyncd Service: RUNNING
ipa: INFO: The ipactl command was successful
```

NOTE	Always interact with the services that make up IdM through either ipactl or using systemctl/service against the ipa service object. Trying to interact with services such as named or the kdc directly can lead to unexpected issues.

Next, try logging into your ipa server's Web interface at https://nameofidmserver.

Verify you can do a DNS lookup against your IdM server (assuming IdM is hosting DNS).

```
[jritenou@jritenour-work ~]$ dig jmr.lab @jr-idm1.jmr.lab

; <<>> DiG 9.10.3-P3-RedHat-9.10.3-10.P3.fc23 <<>> jmr.lab @jr-idm1.jmr.lab
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34164
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;jmr.lab.                       IN      A

;; AUTHORITY SECTION:
jmr.lab.                3600    IN      SOA     jr-idm1.jmr.lab. hostmaster.jmr.lab.
1455638324 3600 900 1209600 3600

;; Query time: 68 msec
;; SERVER: 10.15.74.15#53(10.15.74.15)
;; WHEN: Thu Feb 18 13:24:15 EST 2016
;; MSG SIZE  rcvd: 91
```

Finally, verify you can do an ldapsearch against your IdM server.

```
[root@jr-idm1 ~]# ldapsearch  -h localhost -D "cn=directory manager" -W -b
"cn=users,cn=accounts,dc=jmr,dc=lab" filter "dn"
Enter LDAP Password:
# extended LDIF
#
# LDAPv3
# base <cn=users,cn=accounts,dc=jmr,dc=lab> with scope subtree
# filter: (objectclass=*)
# requesting: filter dn
#

# users, accounts, jmr.lab
dn: cn=users,cn=accounts,dc=jmr,dc=lab

# admin, users, accounts, jmr.lab
dn: uid=admin,cn=users,cn=accounts,dc=jmr,dc=lab

# jritenour, users, accounts, jmr.lab
dn: uid=jritenour,cn=users,cn=accounts,dc=jmr,dc=lab

# realm-capsule, users, accounts, jmr.lab
dn: uid=realm-capsule,cn=users,cn=accounts,dc=jmr,dc=lab

# oseadmin, users, accounts, jmr.lab
dn: uid=oseadmin,cn=users,cn=accounts,dc=jmr,dc=lab

# search result
search: 2
result: 0 Success

# numResponses: 6
# numEntries: 5
```

That’s about it for all the "pre-flight checks", so to speak. Next, you’ll want to try practical usage, such as adding a new user/groups, and joining a host to Idm - both of which will be covered later.

Installing Additional IdM Servers

Generally speaking, installing an IdM replica is similar to installing the first master, but tends to be faster as there is not as much initial configuration to perform. Additionally, it may be desired to have some replicas that aren’t hosting all services (such as a replica that isn’t hosting DNS or a CA, for example).

To set up an IdM replica, you begin by running **idm-replica-prepare** on an existing IdM server. Supply the fully qualify domain name of the new server, along with it’s IP address, and this will create the host record, add it to DNS, and generate a gpg file under /var/lib/ipa to use on the new replica server to complete the process. The intended replica server should **not be joined to IdM** prior to running this step.

Next, take the /var/lib/ipa/replica-info-<hostname>.gpg file and copy it to /var/lib/ipa on your intended replica.

If it hasn't already been done yet, install the ipa-server packages (and ipa-server-dns if you want to host DNS on this server as well).

Next, run ipa-replica-install with the gpg key file specified

```
ipa-replica-install /var/lib/ipa/replica-info-jr-idm2.jmr.lab.gpg
```

This will establish a connection to the master and verify communication, then bring up all the required services.

NOTE	ipa-replica-install accepts most of the same parameters ipa-server-install does.

There are several ways to test replication within LDAP, but for the purpose and scope of this guide, the quickest way to do so is to create an object on on IdM server, and verify it appears on the other server.

```
[root@jr-idm1 ~]# ipa user-add wcobb
First name: Wilbur
Last name: Cobb
------------------
Added user "wcobb"
------------------
  User login: wcobb
  First name: Wilbur
  Last name: Cobb
  Full name: Wilbur Cobb
  Display name: Wilbur Cobb
  Initials: WC
  Home directory: /home/wcobb
  GECOS: Wilbur Cobb
  Login shell: /bin/sh
  Kerberos principal: wcobb@JMR.LAB
  Email address: wcobb@jmr.lab
  UID: 914200007
  GID: 914200007
  Password: False
  Member of groups: ipausers
  Kerberos keys available: False

[root@jr-idm2 ~]# ipa user-show  wcobb
  User login: wcobb
  First name: Wilbur
  Last name: Cobb
  Home directory: /home/wcobb
  Login shell: /bin/sh
  Email address: wcobb@jmr.lab
  UID: 914200007
  GID: 914200007
  Account disabled: False
  Password: False
  Member of groups: ipausers
  Kerberos keys available: False
```

Configuration

Tuning

Replication Agreements

SUDO rules

HBAC

Automount

Certificate Profiles

Multifactor Authentication

- DNS
- LDAP
- PKI
  - External
- Trust
  - Conf
- Kerberos
  - Conf
- NTP

Config

- SUDO
- Automount
- HBAC

Integration

Red Hat Products

CloudForms

Satellite 6

OpenStack

OpenShift

ActiveDirectory

Via Trust

Via Synchronization

Other

HTTP

Troubleshooting

Tools to Monitor IdM

Understanding & consuming logs

logconv.pl

dbmon.sh

Monitoring Replication

Common Setup Mistakes

Known Issues/Work Arounds