SSSD

Client side identity management

LinuxDays 2012

Jakub Hrozek

20. října 2012

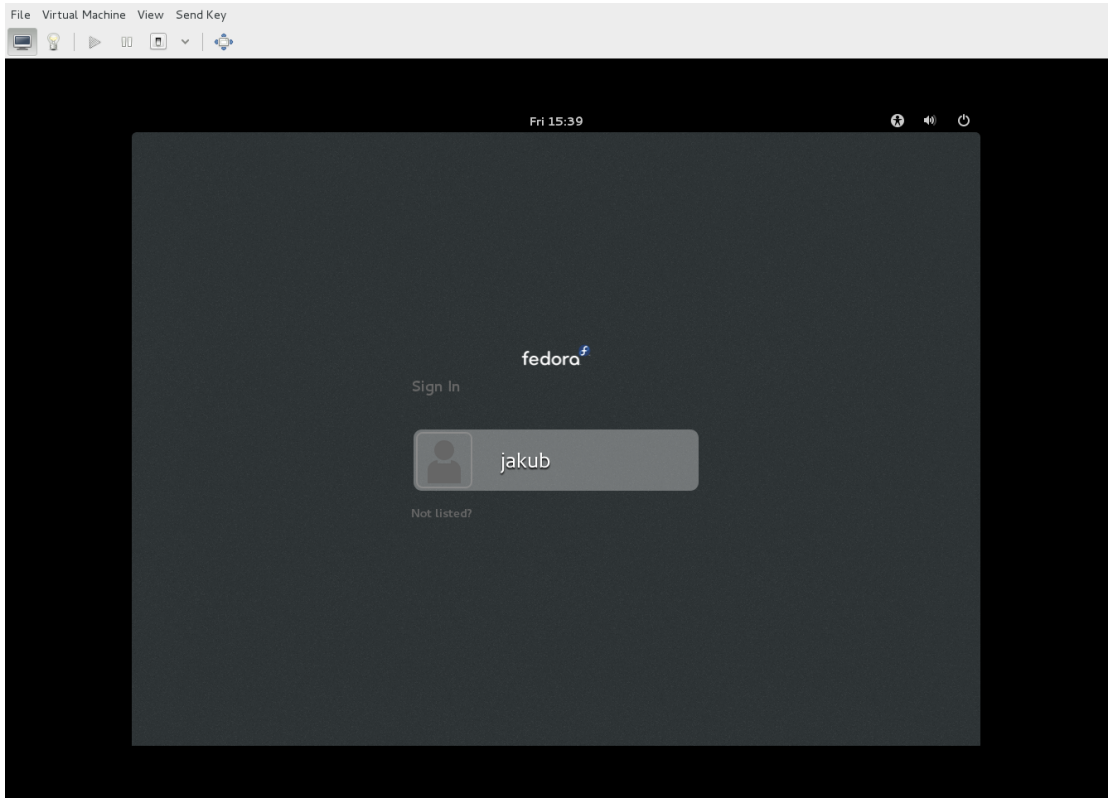
Section 1

User login in Linux

redhat.

User login in Linux

- How does the GDM know about all the users on the system?
- How does the GDM know about properties of a user?
  - home directory, groups, shell, . . .
- How does it verify the password?



redhat.

The login process in general

- GDM, ssh, login, ...
- let's break up the process into multiple steps
  1. gather information about the user
  2. authenticate the user

redhat.

Identifying the user

- In UNIX, everything is a file, right?
  - users stored in `/etc/passwd`, host names in `/etc/hosts`, ...
  - so let's just read the file
- But what if the system needs access outside the files?
  - users in LDAP, host names in DNS, ...
- there needs to be an API
  - usable for all applications transparently - libc
  - the application doesn't usually care about the source of data
  - pluggable - to easily support new databases

redhat.

Name Service Switch

- part of the C library
- modular - a module represents a way to access a database from a source
  - retrieve a *user* from *files*
  - retrieve a *host name* from *DNS*
- several known databases, several widely used sources
  - passwd, group, hosts, services, ...
  - stored in files, LDAP, NIS, ...
- config file `/etc/nsswitch.conf`
  - modules evaluated in the order specified in the config file
  - we want the local root, not the one from LDAP

redhat.

Authenticating the user

- each application can do auth on its own
  - read a password, hash, compare hashes
  - insecure, only handles passwords
- applications need to be abstracted from the means of authentication
  - high level API
  - configurable low-level stack accessible by the API

redhat.

PAM - Pluggable Authentication Modules

- A programmable stack that provides several steps

account - Is the user valid?
auth - Can the user authenticate?
session - Post-login management
password - Password management

redhat.

PAM - Pluggable Authentication Modules

- many different modules available
- it is possible to configure different aspects of the login process
  - authenticate using `/etc/shadow`, LDAP, Kerberos
  - password quality checks, UID range checks, time base checks
  - ...
- configurable using files in `/etc/pam.d`

redhat.

Checkpoint - user login

- In order to log in a user, we need to:

  Obtain information - Name Service Switch
  Authenticate - PAM stack

redhat.

Section 2

Centralized user databases

redhat.

User accounts in a large environment

- it is not practical to distribute files
  - synchronization problems
  - retention
- large organizations need to centralize user information
- usually not only identities but also policies
- several industry-standard solutions
  - UNIX/Linux – LDAP, LDAP + Kerberos, NIS
  - Windows – Active Directory (LDAP + Kerberos)
  - LDAP is the most common identity store

redhat.

Basic LDAP client configuration overview

- using NSS and PAM modules

  NSS - `nss_pam_ldapd` using `/etc/nslcd.conf`

  PAM - `pam_ldap` using `/etc/ldap.conf`

redhat.

The trouble with `nss_pam_ldapd` and `pam_ldap`

- logging in as accounts from different organizations on a single client
- how does one ensure 1:1 mapping between identities and authentication?
- server redundancy and fail over
- what if the servers are not reachable
  - network down, roaming laptop, disconnected corporate VPN

redhat.

The trouble with `nss_pam_ldapd` and `pam_ldap`

- several existing solutions:
  - `ldapsearch | awk > /etc/passwd` in a cronjob :-)
  - local LDAP replica
  - persistent nscd cache
- usually replica of the *whole directory*
  - all entries, potentially huge
  - including attributes we are not interested in
- still need to solve server fail over

redhat.

The real LDAP clint configuration overview

- using traditional NSS and PAM modules

NSS - nss_pam_ldapd using `/etc/nslcd.conf`

id cache - nslcd

PAM - pam_ldap using `/etc/ldap.conf`

auth cache - pam_ccreds

SUDO - sudo using `/etc/sudo-ldap.conf`

automounter - autofs using `/etc/sysconfig/autofs`

redhat.

Integration is the key

An admin can build his own identity management solution, but..

- Bad level of abstraction
  - admin wants to *enroll a client*, not mess around with LDAP
  - the admin needs to understand and master several non-trivial technologies
  - configuration scattered across the system
- admins get frustrated with all the config options..

redhat.

Integration is the key

An admin can build his own identity management solution, but..

- Bad level of abstraction
  - admin wants to *enroll a client*, not mess around with LDAP
  - the admin needs to understand and master several non-trivial technologies
  - configuration scattered across the system
- admins get frustrated with all the config options..

Section 3

SSSD

redhat.

System Security Services Daemon

- `http://fedorahosted.org/sssd`
- a system daemon that provides access to remote identity and authentication services
- developed since Sep 2008

redhat.

SSSD

- a system daemon that provides access to identity and authentication remote resource
- communicates with the rest of the system using its own Name Service Switch module and a PAM module
  - modules only act as forwarders
  - the logic is in the daemon
- supports several 3rd party applications
- the project began as a FreeIPA client but can be (and is) used standalone.

redhat.

Back ends supported by the SSSD

- the currently supported back ends are:
  - LDAP for both identity and authentication
  - Kerberos for authentication
  - IPA
  - Active Directory
  - proxy

redhat.

The benefits of SSSD

- on-disk persistent cache
  - reduces server load
  - seamless offline support, including authentication
- stateful, keeps track of state of remote servers
  - supports server fail over
  - detects networking change to retry operations over the network
- multiple identity information sources (domains)
- only one connection to the LDAP server is open
- automatic Kerberos ticket acquisition
  - passwords stored in kernel keyring when logging in offline
- automatic Kerberos ticket renewal
  - KDC must issue renewable tickets

redhat.

Advanced SSSD features

- In addition to providing identity lookups and authentication
- IPA specific features
  - Host Based Access Control
  - SELinux user mapping
  - OpenSSH host key caching
- support for 3rd party applications that store data in LDAP
- SSSD acts as a transparent proxy and looks up data on behalf of the applications
  - Caching of sudo rules
  - Caching of autofs maps

redhat.

Advanced SSSD features

- access providers
  - simple, per-service, per-host, IPA-specific
- Cross-realm Kerberos trust support
- pre-seeding of users for first boot

redhat.

Advanced SSSD features

- access providers
  - simple, per-service, per-host, IPA-specific
- Cross-realm Kerberos trust support
- pre-seeding of users for first boot

redhat.

SSSD Configuration

- a single config file `/etc/sssd/sssd.conf`

/etc/sssd/sssd.conf

```
[sssd]
domains = LDAP.EXAMPLE.COM

[domain/LDAP.EXAMPLE.COM]
id_provider = ldap
ldap_uri = ldaps://ldap.example.com
ldap_search_base = ou=accounts,dc=example,dc=com
cache_credentials = true
```

redhat.

Active Directory Integration

- SID to UID and GID mapping
- tokenGroups support
- Range retrieval support
- Native AD schema mapping

redhat.

Joining an Active Directory Domain

- provided by the realmd project
- a new package, under active development
- very easy to use
  - `yum install realmd`
  - `realm join --user Username ad.example.com`
- both server and desktop use case

redhat.

The availability of SSSD

- stable release 1.9.2 - AD provider, Sudo, SELinux, . . .
- LTM release 1.8.5
- SSSD is part of all the major Linux distributions
  - Fedora, RHEL, Ubuntu, Debian, Gentoo, FreeBSD ports

redhat.

Future directions

- further AD integration improvements
- Smart Card support
- Two Factor Authentication
- Desktop integration with general D-Bus interface
- Monitoring of expiring tickets
- RADIUS authentication provider

The end.

Thanks for listening.

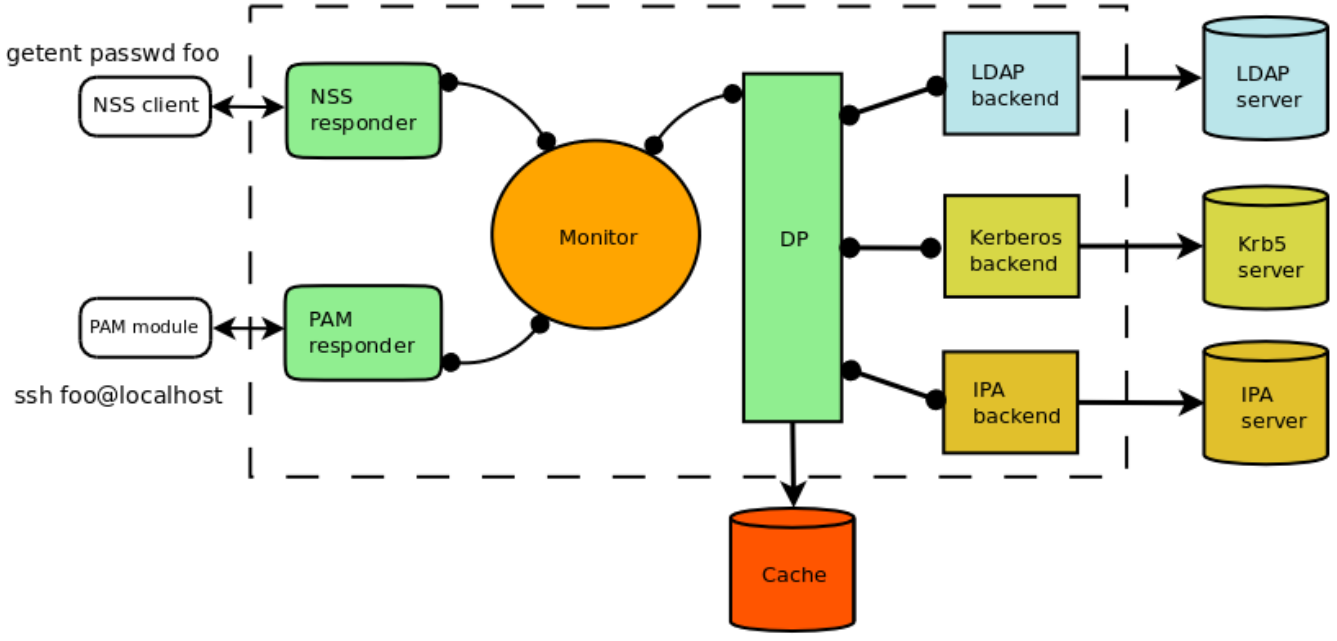
redhat.

SSSD Architecture

- monitor - central process monitoring other worker processes
- the services itself run in separate processes
  - NSS responder responds to identity information coming from the `nss_sss` module
  - PAM responder performs authentication on behalf of the `pam_sss` module
  - each domain runs in a separate process as well
- processes communicate using D-Bus protocol

redhat.

SSSD Architecture



redhat.

talloc

- hierarchical, reference counted memory pool system with destructors

Code example

```
struct foo *X = talloc(mem_ctx, struct foo);
X->name = talloc_strdup(X, "foo");
talloc_free(X);
```

- `talloc_free(X->name)` != `talloc_free(X)` != `talloc_free(mem_ctx)`
- n-ary tree where you can free any part of the tree with `talloc_free`
- provides destructors
- provides means to "steal" pointers from one context to another