Large deployment of GNOME from the administrator’s perspective

Mini Debconf Paris 2012

25 November 2012
Debian is awesome to use in a 1000+ machines environment
- Automated deployment tools: FAI, debian-installer
- Customization: custom APT repositories
- Administration tools, and our famous reliability!

Workstations are a good use case, with GNOME as the desktop
- The easy way: leave users with self-administration permissions
  - But it doesn’t scale very well in terms of support
- Our way: standard workstations with no specific permissions

In order to ship the best systems for users:
- How does GNOME actually work on the inside?
- Where are important places to look for a configuration / a problem?
- What can I tweak on my systems?
OUTLINE

1. The base plumbing for the desktop
   DBus, PolicyKit, ConsoleKit

2. User settings
   GConf and GSettings

3. Login and password management
   The display manager & the keyring

4. Networking with GNOME
   Configuring and delegating the network with Network-Manager
   The virtual filesystem layer

5. Miscellanea
   Other plumbing
   Using the plumbing in custom scripts
   Deploying the configuration on workstations
D-Bus

- D-Bus is the basis for inter-process communications between GNOME applications and the underlying system
  - Based on a typed messaging system over Unix sockets
  - Implements an asynchronous RPC mechanism
  - The system bus is started at boot and never restarted
  - The session bus is started before the session manager by X11 scripts
- Services can either
  - Start by themselves and register a name, e.g. org.freedesktop.NetworkManager
  - Be auto-spawned by the DBus daemon
    → /usr/share/dbus-1/services/*.service and /usr/share/dbus-1/system-services/*.service
- Basic permissions management in /etc/dbus-1/*.conf
  - Most relevant daemons use PolicyKit instead
ConsoleKit and PolicyKit

- **ConsoleKit** keeps track of users logged on. Try the command: `ck-list-sessions`
  - Can be queried to know which user is physically logged on (active = TRUE)
  - In jessie, will be replaced by a similar systemd service
  - Default action: udev-acl (see `/lib/udev/rules.d/70-acl.rules`)
    - Sets permissions dynamically on a number of devices like `/dev/snd/*`
    - Most specific groups (audio, video, netdev…) are obsolete.

- **PolicyKit** adds complex permissions management to D-Bus
  - Can wrap any D-Bus call, invisible from the application

![Diagram showing the interaction between Application, D-Bus, PolicyKit, System service, User PolicyKit agent, ConsoleKit, and the default policy.](image-url)
Tuning the default policy

- Ship a file in `/etc/polkit-1/localauthority/30-site.d/my-config.pkla`
  - [Allow users to shutdown the system even when someone else is logged on]
    - Identity=*  
    - Action=org.freedesktop.consolekit.system.stop-multiple-users  
    - ResultAny=no  
    - ResultInactive=no  
    - ResultActive=yes
  
  - [Let some users change the CPU frequency by hand]
    - Identity=unix-group:benchmarks  
    - Action=org.gnome.CPUFreqSelector  
    - ResultAny=no  
    - ResultInactive=no  
    - ResultActive=yes
  
  - [Let a user install any package from the repository using software-center]
    - Identity=unix-user:joss  
    - Action=org.debian.apt.install-packages  
    - ResultAny=no  
    - ResultInactive=no  
    - ResultActive=auth_self
  
  - In jessie, you will be able to set more complex rules using JavaScript
User settings in GNOME 2.x: GConf

- Still used by a few applications, but not the core of GNOME in wheezy
- Stack of stores implementing defaults, user settings, mandatory (readonly) settings
- Debian-specific paths:
  - `/usr/share/gconf/schemas` → schemas (+ upstream defaults)
  - `/usr/share/gconf/{defaults,mandatory}` → overrides and mandatory settings
  - `/var/lib/gconf/*` → default stores (where schemas/defaults are applied)
  - `/etc/gconf/2/path` → the stores list

- Changing a user setting: `gconftool --type type --set key value`

- Changing a system setting:
  `gconftool --direct --config-source xml:readwrite:/etc/gconf/gconf.xml.defaults --type type --set key value`

- Changing a setting in a Debian package:
  `debian/package.gconf-defaults` or `package.gconf-mandatory
  /path/to/key value`
  `dh_gconf --priority 90`

- Which settings are available?
  `gconf-editor` or `gconftool -R`
Schemas, defaults and overrides are managed by the client

The daemon uses binary databases for speed

Changing a user setting:
- `gsettings set org.gnome.desktop.sound event-sounds false`

Listing all settings:
- `gsettings list-recursively org.gnome.nautilus`

There is also the (buggy) dconf-editor

I don’t like those beeps
Tuning GSettings in a package

- Ship an override file in `debian/package.gsettings-override`
  - `dh_installgsettings --priority=90`

  - # Custom background
    - `[org.gnome.desktop.background]`
    - `picture-options='zoom'`
    - `picture-uri='file:///my/nice/picture.svg'`

  - # Squeeze-like icons on the desktop
    - `[org.gnome.desktop.background]`
    - `show-desktop-icons=true`

  - # I haz a theme
    - `[org.gnome.desktop.interface]`
    - `gtk-theme='FabulousTheme'`
    - `icon-theme='Wonderfullcons'`
    - `[org.gnome.desktop.wm.preferences]`
    - `theme='CoolBorders'`

  - # Default applications and extensions in the shell
    - `[org.gnome.shell]`
    - `favorite-apps=['evolution.desktop', 'libreoffice-impress.desktop', .....]`
    - `enabled-extensions=['apps-menu@gnome-shell-extensions.gcampax.github.com']`

You can also use XML files for evolving backgrounds

The GTK theme needs to have the same name for GTK+ 2.0 and 3.0
D-Conf: default and mandatory system settings

- Configure a system database: `/etc/dconf/profile`
  - user-db:user
  - system-db:local

- Default settings then go in `/etc/dconf/db/local.d/00_my_defaults`
  - # Those users are too dumb, don’t let them do anything
    - [org/gnome/desktop/lockdown]
      - disable-applications-handlers=true
      - disable-log-out=true
      - disable-print-setup=true
      - ...

- Make those defaults mandatory with `locks`: `/etc/dconf/db/local.d/locks/my_locks`
  - /org/gnome/desktop/lockdown/disable-applications-handlers
  - /org/gnome/desktop/lockdown/disable-log-out
  - /org/gnome/desktop/lockdown/disable-print-setup
  - ...

- To `update the database`:
  - dconf update

Separator for defaults is `/` (instead of . for schemas)
All communication goes through D-Bus

Tight integration with ConsoleKit (manages user/VT/display relations)

Displays are started and closed dynamically

Minimal login session launched to manage login (with full a11y support)
Configuring GDM

- Daemon configuration: `/etc/gdm3/daemon.conf` (Debian-specific)
  - Enabling autologin, debugging, VT configuration...
  - XDMCP

- The real configuration for the minimal session (Debian-specific)
  - GNOME 2.30: `/etc/gdm3/greeter.gconf-defaults`
    In a package: `/usr/share/gdm/greeter-config/90_my_config`
    + `invoke-rc.d gdm3 reload`
  - GNOME 3.x: `/etc/gdm3/greeter.gsettings` (GSettings format)
    In a package: `/usr/share/gdm/dconf/90-my-settings` (DConf format)
    + `invoke-rc.d gdm3 reload`

- User defaults (language, session, user icon):
  - In GNOME 2.30: `~/.dmrc` and `~/.face`
  - In GNOME 3.x: AccountsService → `/var/lib/accountsservice`
Storing secrets: the GNOME keyring

- Keeps user secrets in AES-encrypted files
  - Several *keyrings*, each with its own password
  - Also acts as GnuPG and SSH agent
  - Special case: the *login keyring* uses the login password

**User interface:** *seahorse*

- Access user keys and passwords
- *pam_gnome_keyring* also acts when *changing the password*

- Infrastructure constraint: password change is on the same machine
The Network-Manager infrastructure

- **System connections**: started at boot time
  - Controlled by users with appropriate permissions (PolicyKit)
  - Preconfigured by the sysadmin
- **User connections**: started at login time / on-the-fly
  - Secrets stored securely in the keyring
  - Fast user switching: drops the connection (either wanted or buggy behavior).
    - NM 0.9 now defaults to system connections but supports user connections
- System connections with user secrets: 802.1x
Configuring system connections

- Let’s say your DHCP server returns incorrect information, Windows-only
- But you need working DHCP + IPv6 in the outside world

In `/etc/network-manager/system-connections/eth0-internal`

- [connection]
  
  id=eth0-internal
  uuid=deadbeef-1234-1234-1234-deadbeef1234
  type=802-3-ethernet
  autoconnect=false

  [ipv4]
  method=auto

  [802-3-ethernet]
  duplex=full
  mac-address=13:37:15:de:ad:11

  [ipv6]
  method=auto

- In `/etc/network-manager/system-connections/eth0-external`

- [connection]
  
  id=eth0-external
  uuid=deadbeef-1234-1234-1234-deadbeef1234
  type=802-3-ethernet
  autoconnect=false

  [ipv4]
  method=auto

  [802-3-ethernet]
  duplex=full
  mac-address=13:37:15:de:ad:11

  [ipv6]
  method=auto

Other use cases

- Pre-configuring Wi-Fi with a shared key the user doesn’t see (not very secure though)
- 802.1x with a per-machine certificate the user doesn’t see
- Pre-configured 802.1x with per-user credentials

→ All still with access to other networks for users with **PolicyKit permissions**
Networked and local filesystems: the VFS layers

- All communications go through D-Bus
  - All mount actions are explicit from the application
    → Done by gnome-settings-daemon, nautilus or gnome-shell

- Command-line:
  - See all mounted filesystems: `gvfs-mount -l`
  - Mount a CIFS mount: `gvfs-mount smb://server/share/path`

- Gvfs-fuse: nautilus redirects applications not supporting GIO to `~/.gvfs`
  - Needs `fuse` group membership
The palimpsest interface (GNOME disk utility)
Other useful things to know & configure

- Available applications (menus and MIME associations):
  /usr/share/applications and ~/.local/share/applications

- Adding new sub-menus:
  /etc/xdg/menus/applications-merged/my-menu.menu

- CUPS PolicyKit interface: cups-pk-helper
  - Squeeze: system-config-printer{,-applet}
  - Wheezy: directly in g-control-center & g-settings-daemon
  - Query / configure printers, notifications for print operations

- Power management interface: upower
  - g-power-manager (squeeze) / g-settings-daemon (wheezy) defines the policy

- Sound server / mixer: PulseAudio (wheezy only)
  - All mixing now done through it
  - Can be configured to mute sound when switching users
GNOME is easily scriptable

- **In Python:**
  
  ```python
  from gi.repository import Gtk, GnomeKeyring, ...
  ```

  - Formerly in squeeze: autogenerated Python modules
  
  *The conversion script does most of the job*

- **In JavaScript:**
  
  ```javascript
  #!/usr/bin/seed
  Gtk = imports.gi.Gtk;
  ```

- **Some real-world-examples:**

  - A daemon / applet to bypass an IE-only enterprise proxy
    
    - Notification area / libnotify: display status
    - Autostart with the session
    - Store the password in the keyring

  - A script to create CIFS shortcuts accessible from “Places” menu
    
    - Store the password for GVFS
    - `~/.gtk-bookmarks` → “Places” and the shortcuts for GtkFileChooser

  - A script to wrap a RDP / Citrix client
    
    - Extract the same password as for CIFS
An infrastructure for GNOME machines

- The infrastructure is more work than the desktop
- Most of the time: a Debian mirror and a custom APT repository
  → rsync / debmirror and reprepro / mini-dinstall / …
- A custom installation CD: FAI or d-i
- Authentication: OpenLDAP or Fedora directory server
- Printing is tricky
  - CUPS can hold thousands of printers but the UI becomes unusable
  - J. Blache's solution: filtering printers by location with LDAP
    → Welcome to the wonderful world of copyright assignment.
- Network file systems: don’t forget about NTP!
- Administrating a large bunch of machines: forget about simplistic solutions
  - 2 good tools in Debian: Puppet and BCFG2
  - Can be linked to inventory: GLPI + FusionInventory
- Root password management anyone?
- You encrypt partitions? Don’t forget about key escrow
Thank you.