The New GDM Turns You On

William Jon McCann
<jmccann@redhat.com>

GUADEC
11 July 2008
Cadillac
General Motors Brands

- Cadillac - luxury
- Buick - affordable luxury?
- Oldsmobile - (defunct)
- Pontiac - progressive
- Chevrolet - value
- GMC - professional

(Saab, Vauxhall, Opel, Daewoo, Holden, HUMMER)
General Motors Brands

Cadillac  - old folks, 70’s pimps
Buick    - doctors, big in China
Oldsmobile - (defunct)
Pontiac  - crap
Chevrolet - cheap
GMC      - trucks

(Saab, Vauxhall, Opel, Daewoo, Holden, HUMMER)
Designed to use interchangeable parts
DeVille
DeVille (customized)
GM eventually understood
Desire to change
Focus on the user experience
Kate Walsh:
“In today's luxury game, the question isn't whether or not your car has available features like a 40-gig hard drive. It isn't about sun roofs or Sapelli wood accents, popup nav screens or any of that. No, the real question is:

When you turn your car on, does it return the favor?”
A Lesson for Free Software
GDM
Some facts

- Complete rewrite
- 2007 Oct 15 – Replaced trunk with the branch
- 2008 May 1 – Released 2.22.0 (stable)
- Shipped in Fedora 9
- Design driven by user experience, security, accessibility
What's new?
What's new

- Power management support
- User list by default
- Deeper integration of fast user switching
- Uses ConsoleKit for shutdown and reboot
- Accessibility features enabled by default
- “Factory” greeter support
Power Management

- Make the login window user interface run in a “real” session.
- Run gnome-power-manager
- Done!
What is a “real” session?

- X Server
- D-Bus session bus
- Window manager
- GConf daemon
- Notification area
- Session manager
- Settings daemon
Side-effect: gnome-settings-daemon

- Needed a settings daemon for xsettings etc.
- Some features of 2.20 g-s-d were not desired: screensaver, typing break, etc
- Redesigned g-s-d to load modules as plugins
- Plugins loaded if enabled in Gconf
- Uses a different Gconf prefix when run in the greeter
User list by default

- Show “local” users initially
- Wanted to show users that have logged in to a seat before
- Display users in login frequency order
wtmp

- Doesn't store enough information
- Can't be “followed” well
- Has undesirable log rotation behavior
- Is not seat aware
Really want a user session history per seat
ConsoleKit history log

- Made ConsoleKit write all events to a log file
- Provide a simple tool to generate reports from the log

% ck-history --seat=Seat1 --session-type="" --frequent
jmccann  284
mccann   16
scoyote  4
bbob     3
Integrated FUS

- Register greeter “LoginWindow” session with ConsoleKit
- Switch to existing greeter when available
- Absorb fast-user-switch-applet into GDM
  - Rewrite to support new interfaces
  - Share code with GDM
ConsoleKit Shutdown

- org.freedesktop.ConsoleKit.Manager.Restart()
  - org.freedesktop.consolekit.system.restart
  - org.freedesktop.consolekit.system.restart-multiple-users

- org.freedesktop.ConsoleKit.Manager.Stop()
  - org.freedesktop.consolekit.system.stop
  - org.freedesktop.consolekit.system.stop-multiple-users
Factory Greeter

- LoginWindow always running
- Produces all other sessions
- Avoids strange behavior due to the X Server switching to the VT it was started from when it exits
Accessibility Support

Enable features that make your computer easier to use:
- Hear text read aloud (Reader)
- Make items larger (Magnifier)
- Type without the keyboard (On-screen Keyboard)
- See more contrast in colors (High Contrast)
- Press keyboard shortcuts one key at a time (Sticky Keys)
- Ignore duplicate keypresses (Bounce Keys)
- Only accept long held keypresses (Slow Keys)
- Use a larger font size (Large Print)
What is happening now?
Common Accessibility

- Use same accessibility interfaces in all sessions
- Rely on session manager to start / stop ATs
  - Requires sharing more session management
- “New” gnome-session is a good start
  - Uses autostart framework
  - Can bind start/stop to Gconf key
  - Similar to the Greeter session manager
  - EXCEPT it relies on XSMP
Side effect: gnome-session

- Redesigned the “new” gnome-session to support a D-Bus based API
- Use an Inhibit style API to inform the user that programs are running at logout time
  - Also supports inhibiting Suspend
  - We can consolidate interfaces
  - Action can proceed when inhibit is released
Inhibit log out

CD/DVD Creator is busy
A CD burn is in progress.

File Browser is busy
A file transfer is in progress.

Waiting for programs to finish. Interrupting these programs may cause you to lose work.

Lock Screen  Cancel  Log Out Anyway
Common Accessibility

- Add notification area status icon when a11y hotkey or gesture support is enabled
- Notify the user when a11y features are enabled
  - Allows features to be quickly turned off when enabled by accident (e.g., Sticky keys)
  - Provide convenient access to tools
- Use new session framework to start/stop a11y tools in response to Gconf keys
Enforcing session boundaries

- Currently we allow processes to “escape” the session
- Exiting with the session is voluntary
- Applications that don't exit are broken
- Enforcement is required in security sensitive environments
ConsoleKit Session.Kill()

- May use /proc/self/sessionid to find all processes in a login session
- May need to use cgroups to avoid races
- Signal all processes
- Will need to fix applications (eg. screen)
- Need to provide an API to run a program outside of the current security context / session
User Account Information

- Proved to be a major pain in the ass to dig preferred language, session, keyboard, picture out of user's home directory
  - Remote filesystem, encryption, etc.
- Home is not a public data store
- Name service switch information is way too limited
- Should be able to manage centrally
- Need a good API
Directory Services

- Want user experience to drive design and API
- Focus on:
  - Login experience
  - System management and policy restriction
  - Parental controls
  - User creation and management tools
Managing Identity

- User vs. system administrator audience for tools
- User-controlled vs. System-controlled identity
- All current tools seem to get it wrong:
  - gnome-about-me
  - System-config-users
  - Gnome-system-tools
User Account Tool

- Focus on “managed” / system identity
- Targeted for inclusion in GNOME Control Center
- User audience focus
- Use fine-grained privilege escalation (PolicyKit)
- Use a backend directory service that allows for creating more advanced front ends
Multi-Seat

- A seat is a collection of sessions and a set of devices
- Multi-seat is the hosting of multiple seats per system
- Potential to reach millions of new users quickly
  - Large deployments already in Brazil etc
- Multi-seat is a general case
  - Remote login
  - Terminal services
- Need to associate devices with groups of sessions
Multi-Seat Design
What comes next?
Our authentication framework is broken
Authentication

- PAM modules were designed to be run in the process that becomes the session
- And must also request, gather, and verify credentials
- **But** the modules are running in an inactive proto-session that may not have access to devices
Authentication

- PAM modules run in a stack – serially
- No way for module to advertise credentials
- Modules must implement some out of process service to reset the PAM conversation
- And still have support built into any non-trivial authentication user interface
- Or else do bad things like simulating return keys
Not so pluggable
Authentication

- For login, PAM modules must run in proto-session process
- In GDM, we proxy the entire PAM conversation over a private D-Bus connection
- Using conversation design precludes network separation of login prompt and remote session
- Need to know what to ask for
  - “Enter password or swipe finger or insert smart card or ...”
Authorization: Authentication
Authorization

- Authorization **determines** how you will need to authenticate
- Different claims may be required in a security realm
  - Auto-login / No password
  - Simple password login
  - Fingerprint
  - Smart card
  - Smart card + PIN
Authorization

- Whatever starts the session may request authorization for the action:
  - org.freedesktop.system.login-graphical
  - org.freedesktop.system.login-textual

- Need to lookup policy restrictions in directory
  - Machine
  - Login hours
  - Account status
  - Required credentials
New Authentication System?

• Authentication UI would know up front what should be offered
  - Hardware capabilities
  - Available credentials
  - Policy restrictions

• Providers of these credentials would run close to UI
  - Provide interface
  - Run in parallel
  - Signal the UI when status changes
New Authentication System?

• Solve the multiple sufficient credentials (ie. The “or”)
  problem by having each provider display its own UI

• Serialize results to submit to backend for evaluation
  (could be remote)

• What about multi-factor (ie. “and”) ?
  – Not common on personal systems
  – Probably need to customize UI – not just sequence
  – Can provide a legacy PAM provider
  – Provide tools to create new hybrid providers
Inconsistent and arbitrary authentication interfaces
Secure Desktop

- Uniform presentation of authentication requests
- At a known and verifiable location
- With access to secrets (security image?)
- Not prone to eavesdropping
- Only “special” programs can access it
- Card Space?
Session Lock

- Use the Secure Desktop to lock the session
- Current screensavers don't support accessibility well or at all
- Current screen locks can't be trusted
  - Rely on grabs
  - Run as the user
  - May be “trojan”
Secure Attention Key

- Can be used to verify authenticity of authentication prompts
- Switch to Secure Desktop when used in the session
- Requires knowing what session was active at the time and which Secure Desktop to switch to
- Must ensure that keys do not propagate to user session
Switching Sessions

- Currently relies on the VT subsystem
- VT sucks
- We can do better
To Do (soon)

- Move redesigned gnome-session to trunk
- Commit accessibility notification area stuff in g-s-d
- Finish ConsoleKit device support
- Finish ConsoleKit session kill support
- Enable GDM factory greeter
- Create D-Bus DirectoryService
- Add user account management tool to control center
To Do (later)

- Use PolicyKit to authorize login
- Create new authentication framework
- Push session creation down to ConsoleKit
- Use ConsoleKit instead of VT for session switching
- Create “graphical” text logins
- Add support for disconnected sessions
- Add secure attention key support to ConsoleKit