Using and Developing Samba Modules

Jelmer Vernooij
Samba Team
jelmer@samba.org

April 8, 2004
Who am I?

- Samba Team member

- author of several modules and the modules system in 3.0 and parts of it in 4.0
Agenda

- What are modules?
- Using modules in Samba 2.2 and 3.0
- Available modules
- Developing modules
- Modules in Samba 4.0 and future developments

©2004 Jelmer Vernooij (jelmer@samba.org)
What are modules?

```c
void *lib = dlopen("mymodule.so", 0);
void (*myfunc)(void) = dlsym(lib, "myfunction");
myfunc();
```

- Sometimes called 'plug-ins' or DSO's (Dynamically loaded Shared Objects)
- Basically libraries loaded at run-time
Why use modules?

- Separates core from backends

- Dependencies
  - no need for recompiling the core
  - packages with fewer dependencies

- Lowers barrier for people unfamiliar with the samba source to develop modules
Other places modules are used

- Linux Kernel, glibc (NSS), PAM
- GUI Applications: Gimp, xmms, ...
- Servers: apache, php, nessus
Samba 3.0 Module System

General Plugins

General plugins
- Alarm 0.3.5 [libalarm.so] (enabled)
- IRman Control 1.2.10 [libir.so]
- Joystick Control 1.2.10 [libjoy.so]
- Song Change 1.2.10 [libsong_change.so]
- Status Docklet Plugin 1.0 [libstatusdocklet.so]

©2004 Jelmer Vernooij (jelmer@samba.org)
Systems without DSO support

- Still a lot of them out there
- Need ability to compile static
- Similar mechanism to linux kernel
Samba 2.2 Modules

- Only VFS modules could be loaded dynamically
- No support for statically linked VFS modules
- No common system for loading modules
Samba 2.2 Modules — Sample configuration

[programs]
  vfs object = /home/jelmer/oldvfsmodule.so
Samba 3 Modules

- Can be compiled statically
- Loaded automatically from /usr/lib/samba or
- Single filenames specified explicitly
Samba 3 Modules — Building

• Sane defaults — most modules built by default

• Experimental modules can be specified explicitly

./configure \
--with-static-modules=module1,module2 \ 
--with-shared-modules=module1,module2
Samba 3 Modules — Configuring

- No need to specify modules explicitly if they’re located in /usr/lib/samba/SUBSYSTEM

- `identifier:name = value` syntax used for configuration

```
[global]
passdb backend = mysql:db1
db1:host = localhost
db1:user = abmas

[myshare]
    vfs objects = recycle
    recycle:keeptree = True
```

©2004 Jelmer Vernooij (jelmer@samba.org)
This will automatically load /usr/local/samba/lib/pdb/mysql.so and /usr/local/samba/lib/vfs/recycle.so.
Available modules — Subsystems using modules

- VFS: Virtual File System
- Passdb: User Account Database
- Charset: Conversion to and from different character sets
- Idmap: Mapping SIDs to and from UID’s and GID’s
- Auth: Authentication
- RPC: Remote Procedure Call pipes
Available modules — Passdb

Maintains the Samba user database. Similar to NSS.

Specified by the passdb backend parameter. Multiple backends can be loaded at a time.

- tdb: Samba’s internal database format. Used by default.
- smbpasswd: The old plaintext /etc/samba/smbpasswd file.
- ldap: LDAP
- postgresql: PostgreSQL
• xml: XML files

• mysql: MySQL
Available modules — VFS

Provides a 'Virtual File System'.
Specified by the *vfs objects* parameter.

- **audit**: Log what files users are using
- **default_quota**
- **extd_audit**
- **fake_perms**
- **netatalk**

©2004 Jelmer Vernooij (jelmer@samba.org)
- readonly

- recycle

- databasefs

- antivirus: Automatically check for viruses

Multiple modules can be used at the same time.
I would like to open `\FOO\BAR\bla.zip`

This doesn't look like a deleted file.. ignoring it.

Scanning the file showed no viruses, allowing...

```
open("/my/location/bla.zip", O_RDONLY)
```
Available modules — CharSet

Provides conversion between UCS-2 (used internally in Samba) and another charset.

- CP850: Support for 850 Codepage on systems without iconv
- CP437: Support for 437 Codepage on systems without iconv
- weird: Test module for developers, has a different ‘Q’ symbol.

A Japanese Samba user group has developed two modules that convert to HEX and LE.
Available modules — Idmap

Maps between Unix and Windows User and Group IDs. SID \(<\) UID/GID

Can be set using *idmap backend*.

- tdb: default storage of uid/rid mappings
- ldap: store mappings in LDAP database
Available modules — Auth

Authenticates connections.

Samba automatically determines which modules need to be loaded. Can be overriden using the auth methods parameter.

- rhhosts: Trust all connections from certain remote hosts
- sam: Lookup user in Samba user database and check password
- unix: Lookup user in Unix password database and check password
- winbind

©2004 Jelmer Vernooij (jelmer@samba.org)
• server

• domain:
Available modules — RPC

Support for Windows Remote Procedure Calls.
Loaded automatically when a pipe is used.
lsa, reg, lsa_ds, wks, net, dfs, srv, spoolss, samr, echo
Developing your own modules — How does it work?

• Load plugin file or initialise statically linked module

• Module registers backend for one or more subsystems

• Subsystem keeps list of available backends and uses one whenever it needs it
Developing your own modules — Simple module

```c
#include "includes.h"

NTSTATUS init_module(void) {
    printf("Hi there!\n");
    return NT_STATUS_OK;
}
```

©2004 Jelmer Vernooij (jelmer@samba.org)
Developing your own modules – Example: VFS module

— Presentation of Andrew Bartlett’s fake_perms module —
Developing your own modules — Tips

• Don’t get lost in the API

• Start with one of the examples from examples/ from the tarball.

• Use the helper utilities (pdbedit, vfstest)
Developing your own modules — What happens underneath

- *passdb backend* is set to “mysql”
- Passdb subsystem looks up ’mysql’ in its list
- Passdb subsystem doesn’t find ’mysql’
- */usr/local/samba/lib/mysql.so* is loaded
- *init_pdb_mysql()* registers passdb backend ’mysql’
- Passdb subsystem looks for ’mysql’, finds it and uses it
• mysql is used
Developing your own modules — Useful references/tools

- Samba Developers Guide
- "Implementing CIFS", by Chris Hertel
- ethereal, www.ethereal.com
Developing your own modules — License

- Should be GPL or GPL-compatible
- May not be linked against proprietary libraries

©2004 Jelmer Vernooij (jelmer@samba.org)
Future — Samba 4 Modules

- Specified, but wildcards can be used
  (e.g. modules = /usr/lib/samba/* .so)

- Used for a lot more

- Can still be compiled statically

- Pre-load and post-load
Future — Samba 4 Preload and Postload

Pre-load

- Loaded at smbd start-up
- Only loaded once, less overhead for new connections

Post-load

- Loaded after client connects
- Not always available (multi-threaded, single-process, nmbd)
• No need to restart smbd after module upgrade

• Ideal for module programmers (for every new client, the modules are ‘freshly’ loaded)
Future — New modular subsystems

- ntvfs: VFS that uses NT semantics
- registry: Access to the Windows Registry
- gums: Groups and Users Management System
- process models

©2004 Jelmer Vernooij (jelmer@samba.org)
Future — New modules

- passdb: ads
- rpc pipes: exchange/MAPI (?)
- registry: RPC (remote registry, NTUSER.dat)
Future — Changes in the module system

• Ability to unload

• Automatically unload modules that crash (setjmp, longjmp)

• Provide higher stability (might be important for high availability)

• Somewhat more restricted API

• 'Toolkit' for creating modules, without need to have Samba sources around.

• Separation between subsystems
• Better config system
Future — Separation between subsystems

Fewer interdependencies

Samba 3:
Samba 4:
Future — Separation between subsystems

- Fewer knowledge required to extend Samba (in most cases, no need to know about protocol semantics)
Future — Libraries

- Expose subsystems to the rest of the system, e.g. install them as .so’s
- Useful for integrating with projects like KDE, GNOME
Config changes

• Validation checks (does that parameter really exist? Is it valid?)

• Register config variables for use in the web manager?
Thanks

http://jelmer.vernstok.nl/slides/

Questions?