Embedding Samba 4

Alexander Bokovoy
ab@samba.org
Samba Team
More than 10 years of «Samba in the box»

• 1995 – Samba 1.9 release spurred number of embedded Linux and FreeBSD devices
• 1996/1997 – Whistle Communication's InterJet
• 1997/1998 – Cobalt Qube family of embedded GNU/Linux with Samba

With variety of operating systems:
GNU/Linux, FreeBSD, NetBSD, OpenBSD, Mac OS X, Solaris, ...
and architectures:
x86, PowerPC, MIPS, ARM, Sparc, ...
More than 10 years of «Samba in the box»

• Fast forward to 2006 – there are dozens of embedded Samba devices:
  ➢ Linksys NSLU2, Buffalo LinkStation, Maxtor SharedStorage, Lacie Ethernet Disk Mini, Intel SOHO NAS, and many more
  ➢ Now even Samba on Chip with Broadcom BCM4780P NaSoC solution

• Versions used:
  ➢ Samba 2.0, 2.2, 3.0
  ➢ With little or no modifications of Samba core code
  ➢ Extensions are done usually via external programs or VFS
«Let hundred flowers bloom»

• Embedded Samba usage (so far):
  ➢ Small NAS device for Home and Office
    • File sharing
    • Print server
    • Basic workgroup support
  ➢ Integrated gateway for external storage systems (USB-attached drives, wifi-enablement)
  ➢ Streaming and media applications support (e.g. ApplianceWare platform for Intel NAS)
  ➢ Integrated authentication for other applications (Proxy servers, SSO for specific applications)
  ➢ Automatic back-up solutions
«Let hundred flowers bloom», part II

• New devices on the market allow for more innovative approaches:
  ➢ Nokia 770 “Internet Tablet”, and others:
    • Pervasive access, pervasive administration
    • Rich client experience possible but requires exposure of client-side Samba interfaces
  ➢ Linksys NSLU2:
    • Micro-domain controller for masses:
      ▪ Take cheap home device and introduce fullFeatured ADS-compatible domain controller for home use
      ▪ Automatic workload scenario generator with Samba 4 CIFS proxy and nbench features
  ➢ PlayStation 3: ultimate home media center
Prerequisites for embedding Samba

• Device requirements
  ➢ Enough resources (RAM, disk space, CPU)

• Build system requirements
  ➢ Cross-compilation and toolchain support
  ➢ Easy way to include only needed functionality
    • No need to add printing core if there will be no printing support at all

• Integration requirements
  ➢ Easy way to manipulate configuration subsystem
  ➢ Performance monitoring facilities
Device requirements

• Anything with RAM $\geq 64$Mb is good for consideration
  ➢ Samba 4 has less memory consumption than Samba 3

• Anything with frequencies $\geq 200$MHz (ARM9, MIPS, PowerPC)
  ➢ DMA support is important
  ➢ Good context switching support is highly desired
  ➢ Decent peripherals, “higher LAN speed $\Rightarrow$ higher CPU”

• Disk space: $\geq 128$Mb (40Mb for full Samba 4 install)
  ➢ Occupied disk space could be lowered significantly
Build system requirements

• Any operating system with decent file systems and decent POSIX support
  ➢ GNU/Linux is the preferred choice today (Embedded Debian variations, Denx.de, etc.)
  ➢ FreeBSD is supported as well (ACLs, xattrs)
  ➢ mmap support is very recommended

• Toolchain support: GCC is preferred, 3.4 is better than 4.x for our goals
  ➢ Use vendor's compiler if the platform is better supported with it (i.e., Cell BE)
  ➢ CPU transparency support is very welcomed
  ➢ Scratchbox is supported and generally gives better results than a regular cross-compilation
Samba 4 build system

• Samba 4's build system:
  ➢ Modular design
  ➢ Support for private and public dependencies between modules
  ➢ Both shared and static builds are supported
  ➢ Out-of-tree builds are supported
  ➢ The build system uses perl, GNU make, and m4
  ➢ Perl-based PIDL generator
  ➢ Two C-based code generators in embedded Heimdal
Build system, part II

Red crosses of cross-compilation

- Macros for checking results of program's execution:
  - AC_TRY_RUN
- Macros for checking libraries which might include system ones:
  - AC_CHECK_LIB
- Binary code generators:
  - asn1_compile and compile_et
- Checks for run-time features (interfaces, /proc)

Samba 4 enjoys all of them
How to remove red crosses?

• Use native builds
  ➢ For example, ApplianceWare NAS toolkit for Intel IOP is built natively using RPM on GNU/Linux on XScale (ARM9)
    ➢ Takes ages to compile (glibc takes 9 hours, Samba takes 3 hours)
  ➢ For most of embedded x86 SBCs regular PC could be used with appropriate compiler's flags
  ➢ For ARM targets Scratchbox.org is the right answer
  ➢ For Cell BE/PowerPC there is Full-System simulator available for regular PCs
How to remove red crosses?

• If nothing helps, cross-compilation could be fixed
  ➢ For building Heimdal's code generators we need to configure Samba 4 for host system
    ➢ Heimdal uses Samba'4 config.h
    ➢ Out of tree build is required
    ➢ Use –srcdir option to specify original source tree
  ➢ After asn1_compile and compile_et are available, they can be copied to bin/ of the main tree to satisfy make rules
    ➢ Configure main tree for target system as usual
    ➢ Build it!
How to remove red crosses?

• What's about AC_TRY_RUN?
  ➢ AC_TRY_RUN has third argument “cross-compilation”
    ➢ If not specified, AC_TRY_RUN will stop during cross-compile
    ➢ All Samba 4 AC_TRY_RUN use cases safely continue configuration process during cross-compilation

• A developer should know target's platform better than autoconf:
  ➢ Use autoconf's cache to specify proper values for AC_TRY_RUN tests
  ➢ AC_CHECK_CACHE() wraps every AC_TRY_RUN so every test can be safely pre-configured
Fine-grained Samba 4 builds

• Samba 4 isn't a single program
  ➢ It isn't a monolith code block also
    ➢ Includes sub-systems, libraries, modules, and binaries
    ➢ Each component links only with required components

• Each binary can be made small as much as possible
  ➢ Libraries can be created easily out of sub-systems using definitions in config.mk files
    ➢ Hard facts: we have 106 libraries and 90 modules currently
  ➢ Binaries can be linked against only required libraries, including external ones
Demo

• Samba 4 registry editor running in scratchbox for Maemo platform (ARM architecture)
Demo running on real device

- The same Registry Editor running on a real device:
Demo running on real device

• The same Registry Editor running on a real device:
Demo running on real device 🍃🍂

- Sample session of accessing smbd from an smbclient running on the same device:

```
/opt/samba4/bin # smbclient -L localhost
dos charset 'CP850' unavailable - using ASCII
Password for [WORKGROUP\root]:
Failed to connect to ncacn_np:localhost - NT_STATUS_LOGON_FAILURE
REWRITE: list servers not implemented
/opt/samba4/bin # smbclient -L localhost -U%
dos charset 'CP850' unavailable - using ASCII

<table>
<thead>
<tr>
<th>Sharename</th>
<th>Type</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC$</td>
<td>IPC</td>
<td>IPC Service (Samba 4.0.0tp3-S</td>
</tr>
<tr>
<td>VN-build-UNKNOWN</td>
<td>Disk</td>
<td>DISK Service (Samba 4.0.0tp3-</td>
</tr>
<tr>
<td>ADMIN$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVN-build-UNKNOWN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

REWRITE: list servers not implemented
/opt/samba4/bin # uname -a
Linux Nokia770-13 2.6.12.3-cmap1 #1 Mon Mar 6 20:28:51 EET 2006
armv5tejl unknown
/opt/samba4/bin #
```
Questions?

Alexander Bokovoy
ab@samba.org
Samba Team