Possibilities for Samba 3.0 / Samba4 Integration

Andrew Bartlett <abartlet@samba.org>

13th January 2005

Abstract

This paper explores some of the options, both practical and otherwise, for close integration between the production Samba 3.0 release and the Samba4 development code. It examines integration attempts that have been successful, and those that have failed, and provides recommendations to Samba implementors on how best to proceed.

Introduction

As developers, vendors and large-scale implementors look for increasing features in their Samba installations, many are looking with interest at the Samba4 development project. While still unfinished, many have suggested that Samba4 would be developed faster, better or released sooner if there was close integration possible between Samba4 and the existing Samba 3.0 release. Likewise, Samba 3.0 is showing its age, and could potentially benefit quite substantially from some of Samba4's impressive new technologies.

Samba 3.0

Samba 3.0 is the result of the long, evolutionary development of the Samba suite of CIFS software. Samba 3.0's design has allowed it to gain a reputation as a stable, enterprise-ready product, but has shown growing pains as the Samba community added support for many of the modern technologies.

Samba 3.0 is almost entirely built by 'hand'. That is, there is little if any auto-generated code, and few high level abstractions. From a development perspective, it is a heavy platform, with a lot of code required to perform tasks that could be handled much more easily with a different design.

Samba 3.0's advantage comes from it's long history, and in particular its long history of suitability for 'enterprise' use. Samba 3.0 is not comprehensive, and Samba4's testing has shown how much further there really is to go. For the features it provides, however, it is a very solid product that clearly will be deployed and supported for a number of years yet.

Samba4

Samba4 is the development branch of Samba, built out of an initial frustration with the complex layer that Samba uses to communicate with the POSIX file-systems it operates on, and the lack of flexibility in providing new and better ways to store files for Samba.

Now a complete rewrite of Samba, Samba4 implements a comprehensive DCE-RPC subsystem based on IDL and an asynchronous RPC runtime. With the IDL basis, Samba4's RPC layer is far easier to work with than Samba 3.0. New calls can be added in a matter of hours, not days or weeks. This is assisted by a comprehensive database backend, based on an LDAP-like database known as 'ldb'. This design has made it much easier to code than Samba 3.0's passdb interface, which covered much, but not all, of this space.

Samba4 also provides new subsystems for authentication and security, as well as an LDAP server and modules for a Kerberos server. Together, these bring Active Directory support vastly closer to Samba.

Compared to Samba 3.0, Samba4 development is faster, and less error prone, particularly due to the extensive use of code generation techniques and test-driven development.

A Question of Interfaces

For any two products to communicate and cooperate, there must be a commonly agreed interface, a notion with which Samba, being a network interface implementation at it's heart, is intimately familiar. Internally, Samba has traditionally had many different points of interface, and many others proposed but never implemented.

The utility of some of these interfaces has been shown by vendors such as PADL (with their XAD product), Novell (with eDirectory support) and Apple (with their Open Directory interfaces).

Samba 3.0 Existing Interfaces

Samba 3.0 ships with a number of plug-in and similar interfaces:

passdb

passdb is the single most replaced interface in Samba 3.0. It allows administrators to run their Samba installation against a local TDB, an LDAP server, or the traditional flat-file smbpasswd. The passdb interface is used to share user, partial group, and limited domain information between the database and Samba. In the Samba 3.1 development code, trusted domains are also available via the passdb.

POSIX VFS modules

Perhaps the single most extensible interface in Samba 3.0 is the POSIX VFS interface. This interface has allowed modules such as virus scanners and access auditors to override file-system features in terms of simple POSIX-like operations such as `open()` and `read()`.

Winbind

Winbind provides another interface, with a clearly described structure. While never replaced in practice (except in some limited situations where only part of Samba has been upgraded on a host), the Winbind interface definition is shared between the Winbind client and server libraries. Currently, the client portion has been ported to Samba4.

auth modules

The Samba authentication subsystem provides for authentication modules which may be installed to control or redirect the user login process. As an example, Samba 3.0 used this interface to direct logins to Winbind for processing. This interface has also been successfully used by XAD, Apple and others to integrate Samba authentication into their own authentication infrastructure.

Kerberos Keytab and Secret State

Samba 3.0 has interfaces to export Kerberos keytab information to outside users. Samba4 is one of these "outside users", and can be considered a client of this interface. Samba 3.0 stores this information in the secrets.tdb, which while natural to the Samba 3.0 code, is an awkward but readable format for Samba4.

Samba4 Existing Interfaces

Samba4 provides a large number of interfaces into which plug-in modules may be inserted. Indeed, there seems to be very few parts of Samba4 that cannot be plugged into. However, the following are the interfaces most relevant to Samba 3.0 integration:

ldb

ldb is a core interface in Samba4, since a serious attempt has been made to direct all 'database like' queries via this API. This includes all queries for user and group information, as well as all other long-term state. The utility of this interface has been demonstrated by the construction of hdb-ldb, a plug-in for the Heimdal Kerberos suite that allows Heimdal to use ldb as its data storage back end. As this database may be extracted or imported with standard LDIF, this is a very useful interface for data mining or migration.

NTVFS and the CIFS back-end

The NTVFS interfaces is core to Samba4's architecture, and was created to ensure that multiple, 'rich' virtual file-systems could be created and plugged into Samba.

The CIFS NTVFS back end was created to exercise the entire NTVFS infrastructure, by translating incoming CIFS requests back into CIFS requests to another server.

DCE-RPC proxy

The Samba4 RPC server is capable of relaying DCE-RPC requests, after authentication, to another RPC server. Each endpoint on the target is registered and available on the local Samba4 server. This functionality was used to prove that the issues Samba experienced as side-effects from Microsoft's MS04-11 security patch was in fact related to the authentication system in Samba 3.0. Using Samba4 to handle decryption and authentication and then passing off the request to Samba 3.0, Andrew Bartlett showed that the issues surrounding MS04-11 could be solved with existing knowledge, and were not new cryptographic challenges.

auth Modules

Samba4 has an almost identical auth subsystem to Samba 3.0, and already has a Samba 3.0 compatible Winbind module. This has been used to demonstrate the NTLMSSP and SPNEGO code available from Samba4's version of `ntlm_auth` against a remote Windows 2003 domain (Samba4 does not yet have any domain member capability without Samba 3.0's winbind).

Kerberos Keytab and Secrets State

By virtue of sharing the same Kerberos code, Samba4 can read a system keytab that may or may not have been exported by Samba 3.0. Samba4 can also currently read the old Samba 3.0 style secrets.tdb, however this support is deprecated and will be removed very soon.

Proposed interfaces

Wholesale Named Pipe Outsourcing

For many years, it has been proposed that Samba 3.0 provide the ability to 'outsource' DCE-RPC traffic, as it occurs over 'Named Pipes' on the CIFS protocol. In particular this proposal was to outsource any and all Named Pipe traffic from Samba to an external program. These proposals have never been fully implemented in Samba, however the Samba-TNG team implemented it in Samba-TNG, as has PADL with a patched version of Samba for their XAD product[1].

Past Samba 3.0 / Samba4 integration efforts

There have been a number of past efforts to integrate Samba 3.0 and Samba4, not all of which where successful. All, however, do provide lessons in how to handle these two code-bases.

Failed Efforts

Samba 3.0 back-end for CIFS VFS

Volker Lendecke proposed a patch to Samba4 which caused an instance of Samba 3.0 to be launched to handle the file storage requirements of a Samba4 share connection. The purpose of this patch was to demonstrate the capabilities of Samba4, but also to allow file serving using the same user contexts as Samba 3.0 since Samba4, at that early stage, performed all operations as root and lacked a full mapping of expected CIFS features. At the time, this seemed to be an issue that would be a long time in the fixing. It was also seen that the lack of a 'real' file back end for Samba4 was holding up other development efforts. A patch to avoid authentication in Samba 3.0 was proposed at the same time.

This effort was never integrated as it was beautiful mostly in its 'hack value,' and the root access and correctness issues have since been addressed. By it's very nature, this patch could only be as good as Samba 3.0.

Samba 3.2

Two different attempts, spearheaded by Volker Lendecke and Jerry Carter, were made at merging Samba 3.0 and the new Samba4 development branch, with proposals put forward to merge the Samba4 client library with Samba 3.0, and to

[1] http://lists.samba.org/archive/samba-technical/2002-October/024614.html

merge the Samba4's RPC encoding and transport layers with Samba4. Both of these efforts were targeted as Samba 3.2, and both failed due to the pace at which Samba4 moved, and an unwillingness on the part of Samba4 developers to 'slow down' or compromise the Samba4 development to accommodate the half-merge.[2]

Merging code-bases is perhaps one of the hardest tasks in software development, particularly when they have diverged in the way that Samba 3.0 and Samba4 have. Samba4 branched off the same code-base in early 2003 and has been radically rewritten since. Perhaps more troubling in the merge process were the less radical rewrites, with lots of simple changes to function arguments proving a frustration to those attempting the merge.

Since the two abortive efforts, it has become clear that Samba4 is moving very rapidly to it's own release, perhaps even faster than the 'shortcut' Samba 3.2 effort would be able to go.

Successful Efforts

Winbind Client in Samba4

The integration of the Samba 3.0 Winbind client into Samba4, with its associated authentication module has been a success. This is partly because the Winbind interface is relatively stable and partly because the addition of an authentication module is very unobtrusive. The module allowed the demonstration and testing of Samba4's `ntlm_auth` utility (which was otherwise very hard to test without backing to a Windows 2003 domain). The Winbind module also enabled development of parts of the code required for domain membership. However, by providing a solution to immediate issues of domain membership, it made the construction of a real solution less urgent, and therefore less of a priority.

smbtorture Application to Samba4

In rather a different way to the other approaches described in this document, the Samba4 smbtorture utility has become 'intergrated' as the de-facto testing tool for Samba 3.0. In this role it has been a great success, prompting fixes for short-term issues of correctness to Samba 3.0. While Samba4 is progressing very rapidly, everybody agrees that Samba 3.0 will still be in production sites for some time to come, meaning Samba4's smbtorture has and will continue to improve the quality of Samba 3.0.

Likewise, the IDL generated by the Samba4 product has been used to correct defects in the hand-generated Samba 3.0 code. While this practice can be continued long-term, it is very human-resource intensive because of the manual translation process.

Kerberos Code Merge

In October 2004, work was undertaken in the Samba 3.0 code branch that drastically improved the reliability of the Kerberos code, in particular when faced with salted encryption types. This work was done originally by RedHat and merged by Jeremy Allison into Samba 3.0. Because Samba 3.0's Kerberos code was copied and kept largely intact in Samba4, Andrew Bartlett was able to successfully merged this code into Samba4 in December 2004 / January 2005.

Samba 3.0's nmbd

While a new nmbd is proposed for Samba4, currently Samba 3.0's nmbd is sufficient for use in developing Samba4. For particular situations, some patches are available from the Samba4 source tree.

Ideas for Samba 3.0 / Samba4 Integration

Named Pipe Redirection

This is perhaps the oldest suggestion regarding Samba and RPC services. It has long been suggested that Samba should 'outsource' named pipes to alternate programs. This, it is argued, would allow each named pipe to be developed separately and in parallel, with a faster overall result. Unfortunately, for core RPC services this soon becomes an 'all or nothing' proposition; all of the

[2] Had a compromise been reached with certain 'no go' zones established, internal interfaces fixed and significant effort put into the merge, it was feared that Samba4 development could stall out entirely, or be limited in order to force API compatibility with Samba 3.x.

core RPC pipes must be handled together, and a matching authentication module written. Also, issues surrounding SID/UID mappings must usually also be handled by the same integrated back end, and the LANMAN pipe (used by OS/2 and Windows 9x clients) must be explicitly handled.

Authentication

DCE-RPC connections are often authenticated, in one way or another, and Samba4 implements a particularly complete authentication layer for it's RPC services, and RPC proxy, while Samba 3.0 has a very basic DCE-RPC authentication layer.

Session State

Even for unauthenticated connection it should be noted that redirection of incoming packets on named pipes is not as simple as simply forwarding the data stream, as there is a significant amount of state that is inherited from the CIFS level connection. Correctly handling this state transfer has, for the XAD and Samba-TNG cases, been done by an 'out of band' mechanism, or by prefixing it to the first message. In either case, details such as user identity, groups, and session keys must be communicated and accepted.

Samba 3.0 to Samba4

All this is possible, and it should indeed be possible to hand off named pipes from Samba 3.0 to Samba4 in the same way that a patched Samba 3.0 hands off pipes to XAD. However, there seems to be little benefit: Samba4 already includes a mature file-server, and as such a Samba 3.0 integration project of this type seems to add little of value.

Samba 3.0 is best placed to handle this at the raw named pipe layer, before any DCE-RPC parsing is performed.

Samba4 to Samba 3.0

Likewise, it has been proposed that Samba4 hand it's RPC services off to Samba 3.0 - providing a more functional file-server with the backing of a known RPC server. This is more interesting, until Samba4 surpasses Samba 3.0 in RPC function, but will require some effort to correctly handle UID mappings (which are tightly integrated with ldb in Samba4).

Samba4 is best placed to handle this RPC handoff post-authentication in the existing DCE-RPC proxy.

LDB integration efforts

Perhaps the most interesting possibility for the integration of Samba 3.0 and Samba4 lies in the new LDB subsystem. Being such an open format, and with an LDAP server in development, the possibilities are that separate Samba 3.0 and Samba4 components could update or at the very least read the same integrated data source.

Samba4 SamSync

Perhaps the most interesting LDB related integration idea is that Samba4 would re-implement the SamSync 'vampire' code, and place the results in an structured LDB. This could then be extracted to LDIF and munged into a format compatible with the Samba 3.0 LDAP schema, or one of the other compatible passdb back-ends.

pdb_ldb

The proposal here is that Samba 3.0 will use its standard LDAP libraries to talk to a ldapi:// LDAP server, which is in fact Samba4, running the Samba4 ldb schema. This will allow Samba4 to update the database in its native format and Samba 3.0 to read it.

Winbind replacement

The winbind interface is very interesting because of it's relative stability and the fact that it is largely an interface to external programs. As such, it is possible to conceive that either Samba 3.0 or Samba4 could provide this interface.

Samba 3.0 Winbindd for Samba4

As Samba4 matures, it is reaching the stage where it would be quite practical to treat it as an external domain controller (even if on the same machine), and have winbindd provide accounts to

POSIX, while performing the other duties typically assigned to it. This may bridge the gap between Samba4's ldb implementation of user details, and the POSIX world's expectation of user and group behaviour.

Samba4 Winbindd for Samba 3.0

Samba4 has many advantages in the construction of a Winbind daemon particularly in its asynchronous nature. While not all calls are available asynchronously at the remote end, it is very attractive to consider that the local system should not block waiting for each and every one of them. A winbindd compatible with Samba 3.0's file-serving and domain-control logic could be constructed, and 'dropped in' to the otherwise existing Samba 3.0 infrastructure.

Perhaps No Integration at All?

The final option that must be explored is that of no integration; simply allow Samba 3.0 and Samba4 to follow along their current development paths. While it is true that Samba4 has a long way to go and it is a big change, the risks associated with creating products based on a hybrid are real and do need to be quantified. In Samba 3.0 development, many vendors of products using Samba wondered if it would be 'safer' or 'easier' to simply merge the aspects of Samba 3.0 that were interesting into Samba 2.2, rather than use the newer code-base. Indeed, during Samba 3.0 development, many things were back-merged, particularly by Jeremy Allison.

Samba 3.0, and in particular early alpha releases of the software, provided a big advantage to those who took it up and provided for it's continued development. These vendors were able to take advantage of their position with new and better products, using the new functionality. These vendors also helped particularly on quality assurance of the new code, and were able to drive development in that way.

Samba4 promises to be the same; while Jeremy and others (particularly those contracted to 'enterprise' customers) will almost certainly continue to fix Samba 3.0 as best as they can, the Samba4 development program will provide new and significant functionality, that will simply not be available on the old platform. While it may be tempting to try and back-port functionality, we have already shown that this is not viable for any significant functionality.

Conclusion

It seems clear that three different approaches will be taken for three different parts of the Samba community:

Existing Enterprise deployments

Those with existing enterprise deployments of Samba 3.0 will wish to avoid change, and will use Samba4 as a testing tool for their existing deployments, and as a diagnostic tool for issues that arise in production.

These customers will fund, occasionally at great expense, small changes to be made on the basis of Samba4 testing and IDL knowledge.

Deployments needing some new functionality

Where a new deployment is proposed, it seems natural to place the efforts into Samba4. Not only is this development path much faster, the new functionality will be available long-term, rather than being lost with Samba 3.0.

The challenge is to use this functionality in an environment that meets time-lines and stability requirements. By choosing interfaces (such as ldb) with Samba 3.0 carefully, successful hybrid development should be possible.

While certain customer and product demands may well appear to demand that Samba 3.0 alone 'grow' these new features, the cost-benefit balance is in Samba4's favor, simply due to it's greater ease of development.

Vendors developing new products

Vendors able to plan their product development time-lines should be putting significant efforts into an exclusively Samba4 solution, due to the increased protocol coverage, and test-driven framework. For those with an ability to plan product development, effort spent on a hybrid development would be better spent on the final Samba4 release.

Glossary

DCE-RPC DCE-RPC is a standard for the implementation of RPC from the Open Group, and is used extensively by Microsoft for remote administration and other tasks.

Kerberos A trusted third party authentication system, based on strong cryptography and tightly built into Active Directory

ldb ldb is an LDAP-like light-wight database, on which much of Samba is built.

NTVFS NTVFS is the VFS interface in Samba4, designed to expose the full richness of the CIFS protocol, as backed by NTFS on Microsoft Windows NT.

POSIX VFS In Samba 3.0, the VFS interface is defined in terms of the basic operations found on POSIX systems, such as `read()`, `write()`, and `open()`, rather than the richer NTVFS interfaces.

RPC Remote Procedure Call, a communication method between two systems, described in terms of functions and parameters.

SID Security Identifier, the globally unique structured numeric identifier for every user on a Windows NT compatible system.

UID User ID, in this case as reflected by the locally unique numeric identifier of users on a Unix-like system.

VFS Virtual File System, an interface that abstracts file system operation details from the application programmer, providing a common interface across multiple possible implementations.

Credits

Thanks to St Bernard Software for funding the production of this white paper, and to the Samba Team, and Vance Lankhaar in particular for providing feedback and corrections.

The source and history for this document are available from Lorikeet SVN `http://websvn.samba.org/cgi-bin/viewcvs.cgi/trunk/white-papers/?root=lorikeet`.