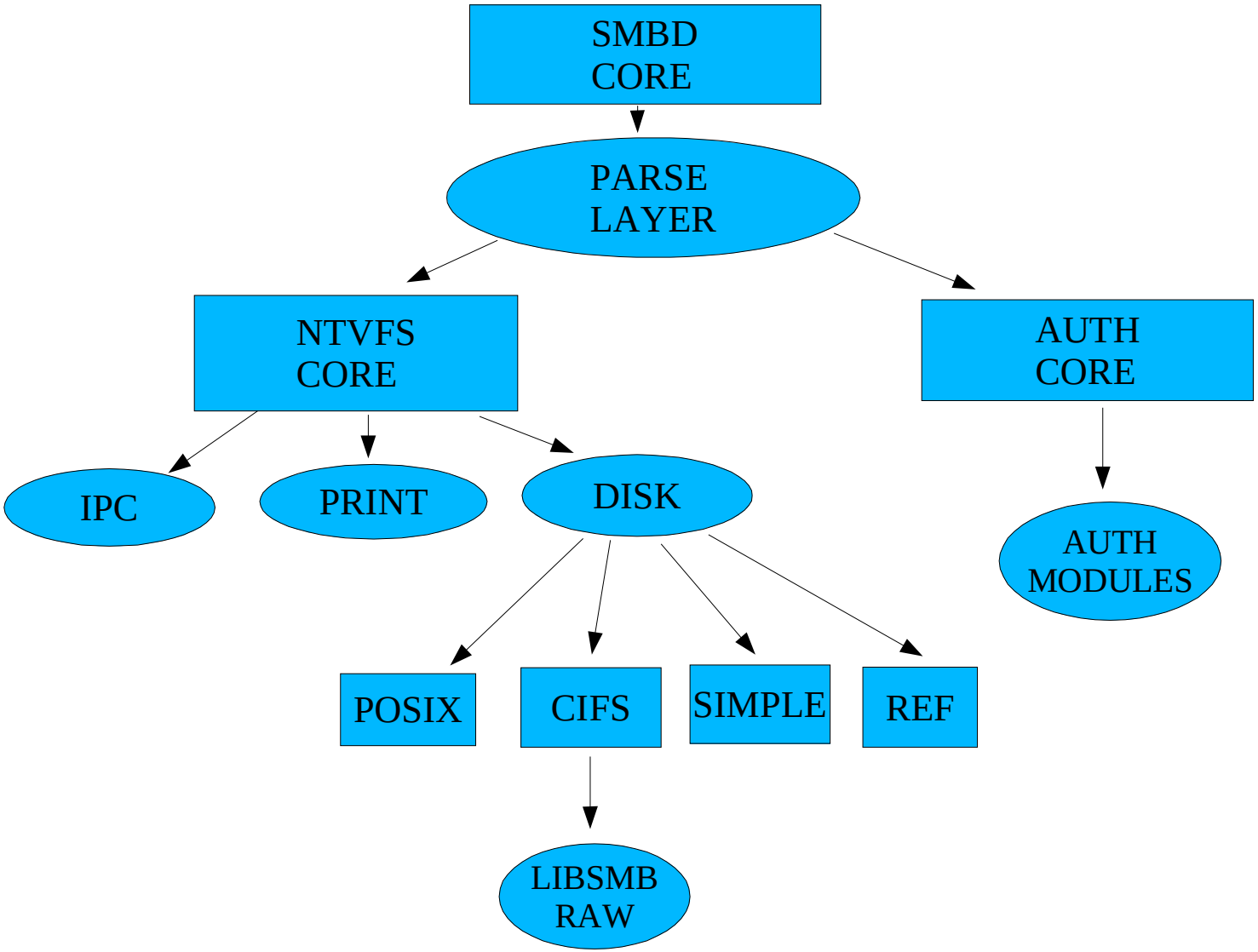
Samba 4

(was exotic filesystem backends)

Andrew Tridgell
tridge@samba.org

Samba 4

- Started out as 'NTVFS rewrite', now expanded to much broader restructure. I hope this will be the basis of Samba version 4.
- Core architectural ideas are:
  - heavily context oriented
  - not tied to POSIX
  - much more modular
  - separation of parse and logic layers
  - much more complete CIFS feature support


SMBD
CORE
PARSE
LAYER
NTVFS
CORE
AUTH
CORE
IPC
PRINT
DISK
AUTH
MODULES
POSIX
CIFS
SIMPLE
REF
LIBSMB
RAW

Samba 3 structure

- Samba 3 still uses the same basic structure as the 1992 code
  - SMB parsing and logic mixed together
  - tightly tied to POSIX
  - static string handling
  - tied to single SMB socket per process
  - no unifying structure

Contexts

- Samba 4 is heavily context oriented
  - server_context replaces globals
  - tcon_context per tree connect
  - request_context per packet
- All functions take a context of some sort
- Contexts contain back-pointers to server_context

Sub Contexts

- Context structures also contain sub-contexts
- For example, server_context contains:
  - negotiate_context
  - substitute_context
  - socket_context
  - tree_context
  - users_context
  - printing_context
  - timers_context

Request context

- replaces inbuf/outbuf and many global variables
- separation of header, command word and data packet sections. Makes chaining clean.
- reduces ties to NBT encapsulation
- buffers allocated to right size, not maximum size
- talloc context for all request related allocation
- requests can be easily deferred, replacing several packet queue mechanisms
- unified bounds checking

Parse Layer

- Samba 4 has a separate SMB parse layer, producing structures describing each request
- unlike Samba 3, all SMB parameters are parsed
- unions used to combine varients on common concepts, such as the many open and read varients
- shares structures with new raw client interface
- shares structures with NTVFS backend API

Parse structures

```
enum fsinfo_level {SMB_FSINFO_GENERIC, SMB_FSINFO_DSKATTR};

union smb_fsinfo {
        /* generic interface */
        struct {
                enum fsinfo_level level;
                struct {
                        uint32 block_size;
                        SMB_BIG_UINT blocks_total;
                        SMB_BIG_UINT blocks_free;
                } out;
        } generic;

        /* SMBdskattr interface */
        struct {
                enum fsinfo_level level;
                struct {
                        uint16 units_total;
                        uint16 blocks_per_unit;
                        uint16 block_size;
                        uint16 units_free;
                } out;
        } dskattr;
};
```

NTVFS layer

- separate interfaces for IPC, DISK and PRINT backends
- receives fully parsed SMB requests with all parameters
- provides mapping for specific to generic backend functions
- modular replacement of backends

Level Mapping

- Allows most backends to only implement 'sane' generic functions
- Allows some backends to implement specific level handlers if need be
- Confines most esoteric SMB knowledge to one place

Backend registration

- Backends can be dynamically registered via modules
- Registration specifies type of backend
- Backends can ask for other backend functions

```
BOOL ntvfs_register(const char *name, enum ntvfs_type type, struct ntvfs_ops *ops);

struct ntvfs_ops *ntvfs_backend_byname(const char *name, enum ntvfs_type type);
```

POSIX backend

- much of the old smbd code will move to the new POSIX NTVFS backend
- consolidation into backend code should make POSIX semantic compromises much clearer
- should we keep the old POSIX VFS system?

CIFS backend

- a NTVFS disk backend that connects to a remote CIFS server
- provides a 'perfect' backend with all protocol features
- allows for testing of core code with MS backend
- not intended to be used in real systems
- ideal testbed for distributed Samba
- will integrate with server level security

Simple Backend

- a NTVFS disk backend mapping to a local filesystem
- no attempt at accurate POSIX mapping
- useful template for new backends
- allows performance cost of POSIX backend to be measured

New server models

- separation of contexts in smbd allows for new modes of operation
  - can break 1-1 socket to process model
  - possibility of threaded architecture
  - interactive mode could handle multiple sockets, useful for debugging
  - allows identification of components needed for distributed operation

Raw client interface

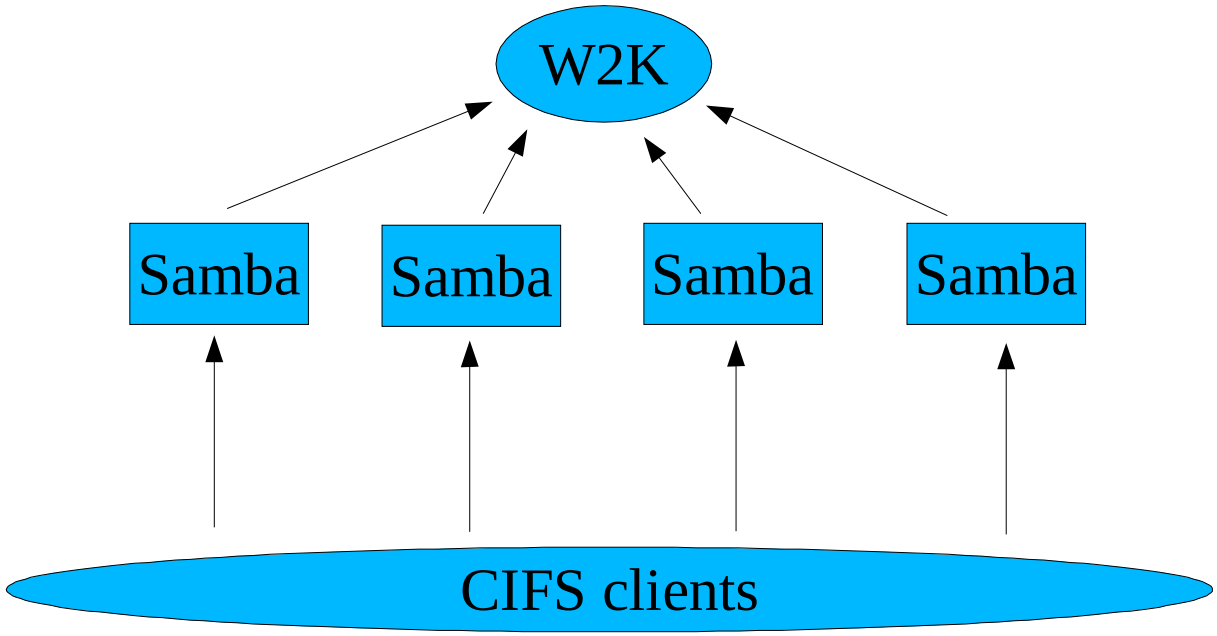
- a new raw client library
- aims to be much more complete than previous libs
- shares parse structures with smbd and NTVFS
- core component of new broad coverage testsuite

Storage Tank

- A distributed SAN filesystem developed by IBM
- NTVFS layer will talk directly to filesystem, not via the kernel
- Has rich non-POSIX semantic
    - case insensitive
    - DOS attributes
    - Extended ACLs

Clustered Samba

- Context structures will allow for several possible clustering methods
- CIFS backend provides ideal clustered testbed



Code status

- Samba 4 codebase at very early stages of development
- can send and receive a few packets
- removed much of core functionality, will need to be replaced
- parse layer well defined, NTVFS interface defined, main contexts defined