ID Mapping Re-Revisited
sambaXP 2009

Michael Adam

obnox@samba.org

2009-04-24

Contents

1 ID mapping - wtf?

ID mapping - what is it, and why?

- Windows users/groups: SID (S-1-5-21-12345679-987654321-512) - world unique
- Samba: needs unix users for file access
- Unix users UID/GID numbers: only unique to system
- Samba: Needs to associate UIDs/GIDs to SIDs
- Foreign domains: winbindd does this ID mapping
- `libnss_winbindd`
- idmap backends: tdb, ldap, ad, rid, tdb2, adex, hash, nss, passdb

2 ID mapping up tp 3.0.24

ID mapping up to Samba 3.0.24

`smb.conf` **idmap options**

```
idmap backend = BACKEND
idmap uid = 1000000-2000000
idmap gid = 1000000-2000000
```

- just one backend
- no support for configuring individual domains
- ⇒ too limited

3 ID mapping since 3.0.25

ID mapping since 3.0.25

- late 2006/early 2007: rewrite by Simo Sorce
- add support for configuring several domains differently
- greatly enhanced flexibility
- more complicated configuration (of course)

configuration - alloc

- NEW: id allocation appears in the configuration
- there is *one* allocator for all allocating backends / domains

```
idmap alloc backend = tdb
idmap alloc config : range = 1000000-2000000
```

```
idmap alloc backend = ldap
idmap alloc config : range = 1000000-2000000
idmap alloc config : ldap_url = ldap://server/
idmap alloc config : ldap_base_dn = ou=idmap,dc=sambaxp,dc=org
```

configuration - idmap

- `idmap backend` deprecated
- `idmap uid` and `idmap gid` change role:
  overwrite `idmap alloc config:range`
- explicit list of domains with config `idmap domains`
- placeholder for all other domains possible
- default domain flag for explicit setting possible

configuration - example

```
idmap domains = CATCHALL AD TRUSTED1

idmap config CATCHALL : default = yes
idmap config CATCHALL : backend = tdb
idmap config CATCHALL : range = 10000-19999

idmap config AD : backend = ad
idmap config AD : range = 20000-29999

idmap config TRUSTED1 : backend = rid
idmap config TRUSTED1 : base_rid = 0
idmap config TRUSTED1 : range = 30000-39999

idmap alloc config : backend = tdb
idmap alloc config : range = 10000-19999
```

criticism

- rather complicated configuration
- slight redundancies
- appearance of the alloc config on the surface somewhat irritating, seems artificial
- not possible to configure domains with different allocating backends and ranges

4 ID mapping since 3.3.0

ID mapping since 3.3.0

- summer 2008: rewrite by Volker Lendecke
- rather pragmatic simplification
- remove `idmap alloc config:range` (use `idmap uid/gid`)
- un-deprecate `idmap backend`
- remove `idmap domains`
- remove `default` flag for idmap configs
- domains with allocating backends in catch-all default config
- read-only backends like rid, ad, usually in explicit configs as before

configuration - simple

```
idmap backend = tdb
idmap uid = 10000-19999
idmap gid = 10000-19999

idmap config MYDOM : backend = ad
idmap config MYDOM : range = 20000-29999

idmap config TRUSTED1 : backend = rid
idmap config TRUSTED1 : range = 30000-39999
```

configuration - slightly less simple (for the fun of it)

```
idmap backend = tdb
idmap uid = 10000-19999
idmap gid = 10000-19999
idmap alloc backend = ldap
idmap alloc config : ldap_url = ldap://id-master/
idmap alloc config : ldap_base_dn = ou=idmap,dc=sambaxp,dc=org

idmap config MYDOM : backend = ad
idmap config MYDOM : range = 20000-29999

idmap config TRUSTED1 : backend = rid
idmap config TRUSTED1 : range = 30000-39999
```

criticism

- good: somewhat more simple, less redundancy
- trying to explicitly configure an allocating domain will fail
- only one allocating config (default)
- let's look a the API for more clues

5 Current API

current idmap API

```
idmap_methods {
        init
        unixids_to_sids
        sids_to_unixids
        set_mapping
        remove_mapping
        dump_data
        close_fn
}
```

current idmap alloc API

```
idmap_alloc_methods {
        init
        allocate_id
        get_id_hwm
        set_id_hwm
        close_fn
}
```

in the winbind protocol

```
WINBINDD_SID_TO_UID
WINBINDD_SID_TO_GID
WINBINDD_UID_TO_SID
WINBINDD_GID_TO_SID

WINBINDD_SET_MAPPING
WINBINDD_REMOVE_MAPPING

WINBINDD_ALLOCATE_UID
WINBINDD_ALLOCATE_GID
WINBINDD_SET_HWM
```

criticism

- appearance of the alloc methods on the surface seems artificial and wrong (to me)
- restriction to have only one (default) allocating config
- appearance of the `set/remove_mapping` in the idmap methods seems utterly wrong
- users of id mapping should just ask for an ID for a SID and get on or not. should not need to take care of allocation and setting ids themselves.
- exposure of the HWM in the allocator seems wrong
- difference between idmap methods and winbind protocol seems wrong

6 Vision - another rewrite?

new rewrite started...

- January 2009: new rewrite started by `/me` to get rid of restrictions
- hide the allocator completely inside the idmap backend modules
- each explicitly configured domain can thus have its own allocator
- this removes the configuration difference between allocating and R/O backends from the user
- it allows for having R/O-backend as default and R/W backends for explicit domains
- make idmap methods and winbind protocol more similar

idmap API

```
idmap_methods {
        init
        idmap_sids_to_unixids
        idmap_unixids_to_sids
        close
}
```

winbind protocol

```
WINBINDD_SIDS_TO_UNIXIDS
WINBINDD_UNIXIDS_TO_SIDS
```

configuration

```
idmap backend = tdb
idmap range = 10000-19999

idmap config MYDOM : backend = ad
idmap config MYDOM : range = 20000-29999

idmap config TRUST1 : backend = rid
idmap config TRUST1 : range = 30000-39999

idmap config TRUST2 : backend = tdb
idmap config TRUST2 : range = 40000-49999

idmap config TRUST3 : backend = ldap
idmap config TRUST3 : range = 50000-59999
idmap config TRUST3 : ldap_url = ldap://map-master/
idmap config TRUST3 : ldap_base_dn = ou=idmap,dc=sambaxp,dc=org
idmap config TRUST3 : ldap_alloc_url = ldap://alloc-master/
idmap config TRUST3 : ldap_alloc_base_dn = ou=idalloc,dc=sambaxp,dc=org
```

Full Stop

- It does not work like this! `:-(`
- the idmap allocator is not only an idmap allocator but an overall unix ID allocator to Samba:
- passdb backend ldap with `ldapsam:editposix` creates UIDs/GIDs with the idmap allocator and stores them in the passdb (user/group LDAP objects)

How to solve this?

- create a separate passdb id allocator? (...no)
- use one overall master id allocater that idmap and passdb allocators use? (...no)
- don't use an allocator in passdb but use winbind/idmap instead. move all of passdb functionality into winbindd. remove group mapping, too. similar to what passdb backend `wbc_sam` ist currently already doing. (...yes!)

7 Further Plans

further plans/ideas

- incorporate `nss_info` with the idmapping configuration
- consolidate `idmap_tdb` and `idmap_tdb2`
- and move the `idmap:script` feature of `idmap_tdb2` to a proper idmap module (`idmap_script`)
- create a `idmap_unixinfo` to talk to a samba domain controller
- rework winbindd idmap process model

  (idmap domain children, async!)
- consolidate winbindd of Samba 3 and Samba 4 ? (libwbclient)

Wake up - time for lunch!