Password Policy with OpenLDAP

From basics to production

Ondřej Kuzník, Shawn McKinney

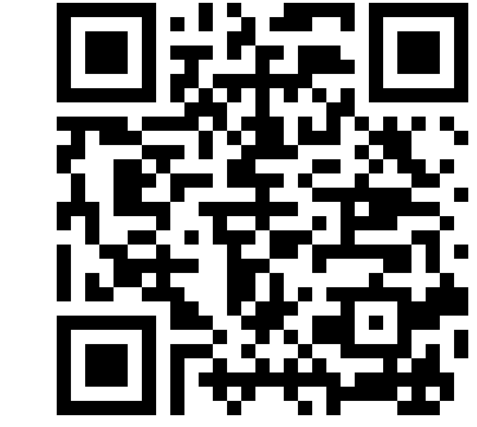
Symas

Our roadmap

We start with a common setup, one blanket password policy and evolve from there:

1. **First contact**: log in and look around
2. **Lockout**: crime and punishment
3. **Slowing attackers** or a slap on the wrist
4. **Quality** changed in 2.6: crafting with quality
5. **Targeted assignment** 2.7: making the policy (assignment) make sense
6. **Lifecycle and idleness** changed in 2.6: start, end, how to deal with unused accounts
7. **Hashing** 2.7: making insecure hashes secure
8. **…** bonus!
9. **Closing** closing remarks

Follow at your own pace



Your server

Log in to your `ldap1` and point the tools at it:

```
export LDAPURI=ldapi:/// LDAPBASE=dc=example,dc=com
```

As root on the socket you are the directory's `rootdn`. Everybody else's password is `<uid>-pw`: `alice` has `alice-pw`.

Decide now how you will change the configuration, and stay with it (all solutions are shown both ways):

- **`slapd.conf`**: edit it, restart slapd
- **`cn=config`**: `ldapmodify` as root, takes effect straight away

For `cn=config`, convert once, now (`slapd.d` takes precedence):

```
cd /opt/symas/etc/openldap && mkdir slapd.d
slaptest -f slapd.conf -F slapd.d && chown -R openldap: slapd.d
systemctl restart symas-openldap-server
```

The cast

One rule: experiment freely but on accounts you create yourself. The below are set up to illustrate concepts at key times. Entries below are of the form `uid=<uid>,ou=people,dc=example,dc=com`:

uid	State	Notes
`alice`	healthy	the control case
`bob`	changed 400 days ago	expired
`carol`	changed 345 days ago	about to expire
`dave`	not seen for 200 days	idle
`erin`	reset by an administrator	must change
`frank`	locked by an administrator	
`grace`	an administrator	
`heidi`	has no password at all	
`mallory`	expendable	lock her out as often as you like
`peggy`	fresh	for changing passwords

If anything goes wrong, the seed LDIF is stored at `/var/symas/openldap-data/bootstrap.ldif` on `ldap1`.

1. First contact

What is the server already doing?

slapo-ppolicy in one slide

- A **policy** is an ordinary entry with objectClass `pwdPolicy`
- The **overlay** applies one to every bind and password change
- Its **state** lives on the account: `pwdChangedTime`, `pwdFailureTime`, `pwdAccountLockedTime`, ...
- 2.5 A **password administrator** is an account that has `manage` (!= `write`) access to `userPassword`, that **includes** rootdn

What your server starts with:

```
overlay ppolicy
ppolicy_hash_cleartext
ppolicy_default "cn=default,ou=policies,dc=example,dc=com"
```

One policy, and everybody with a password is subject to it.

The policy

Attribute	Value	Note
`pwdMaxAge`	31536000	expires after a year
`pwdExpireWarning`	2592000	warned for the last month
`pwdMinLength`	8	
`pwdMaxFailure`	5	five wrong guesses...
`pwdLockout`	TRUE	...lock the account...
`pwdLockoutDuration`	300	...for five minutes
`pwdInHistory`	3	no reusing the last three
`pwdMustChange`	TRUE	an admin reset must be followed by a change

Task: Look around

- Which policy applies to `alice`? `pwdPolicySubentry` says; it is operational, so ask for it **by name**
- As `alice`, read that policy
- As `alice`, look at `bob`'s `pwdChangedTime`

Check: Look around

```
$ ldapsearch -x -D uid=alice,ou=people,dc=example,dc=com -w alice-pw \
     -LLL -s base -b uid=alice,ou=people,dc=example,dc=com pwdPolicySubentry
dn: uid=alice,ou=people,dc=example,dc=com
pwdPolicySubentry: cn=default,ou=policies,dc=example,dc=com
```

```
$ ldapsearch -x -D uid=alice,ou=people,dc=example,dc=com -w alice-pw -LLL \
     -b cn=default,ou=policies,dc=example,dc=com \
     pwdMaxAge pwdMaxFailure pwdLockout pwdLockoutDuration
dn: cn=default,ou=policies,dc=example,dc=com
pwdMaxAge: 31536000
pwdMaxFailure: 5
pwdLockout: TRUE
pwdLockoutDuration: 300
```

```
$ ldapsearch -x -D uid=alice,ou=people,dc=example,dc=com -w alice-pw \
     -LLL -s base -b uid=bob,ou=people,dc=example,dc=com pwdChangedTime
dn: uid=bob,ou=people,dc=example,dc=com
```

The rules are public; how close bob is to a lockout is not.

Asking the server why

A failed bind has to say "invalid credentials" and nothing else. Attach the **password policy control** (`-e ppolicy`) to get more information:

```
$ ldapwhoami -x -e ppolicy -D uid=alice,ou=people,dc=example,dc=com -w alice-pw
dn:uid=alice,ou=people,dc=example,dc=com
$ ldapwhoami -x -e ppolicy -D uid=bob,ou=people,dc=example,dc=com -w bob-pw
ldap_bind: Invalid credentials (49); Password expired
$ ldapwhoami -x -e ppolicy -D uid=carol,ou=people,dc=example,dc=com -w carol-pw
ldap_bind: Success (0) (Password expires in [...] seconds)
dn:uid=carol,ou=people,dc=example,dc=com
```

Task: erin can log in, but...

...nothing else works for her. Find out why, and get her going again: change her password, as erin. Try a short one first.

Check: erin can log in, but…

```
$ ldapwhoami -x -e ppolicy -D uid=erin,ou=people,dc=example,dc=com -w erin-pw
ldap_bind: Success (0); Password must be changed
dn:uid=erin,ou=people,dc=example,dc=com
$ ldapsearch -x -e ppolicy -D uid=erin,ou=people,dc=example,dc=com -w erin-pw -LLL -s base 1.1
Insufficient access (50)
Additional information: Operations are restricted to bind/unbind/abandon/StartTLS/modify password
```

```
$ ldappasswd -x -e ppolicy -D uid=erin,ou=people,dc=example,dc=com -w erin-pw -s short
Result: Constraint violation (19)
ppolicy: error=6 (Password is too short for policy)
$ ldappasswd -x -e ppolicy -D uid=erin,ou=people,dc=example,dc=com -w erin-pw -s Trombone-Marmalade-71
$ ldapwhoami -x -e ppolicy -D uid=erin,ou=people,dc=example,dc=com -w Trombone-Marmalade-71
dn:uid=erin,ou=people,dc=example,dc=com
```

An administrator reset her password (`pwdReset`) and the policy has `pwdMustChange`. The change clears it, and stamps the `pwdChangedTime` that `pwdMaxAge` counts from.

2. Lockout

Five strikes, and then what?

When a Bind fails, things happen

Every wrong password adds a `pwdFailureTime` to the account.

When there are `pwdMaxFailure` of them (5), and the policy has `pwdLockout: TRUE`, the overlay adds `pwdAccountLockedTime` and the account is locked: **even the right password is refused.**

A successful bind clears the failures. Nobody else can: they are `NO-USER-MODIFICATION`, even for an administrator except via a password reset.

Task: Lock mallory out

They're flaky and forgetful. Five wrong passwords, look at their entry as an administrator, then try the right password.

Check: Lock mallory out

```
$ for i in 1 2 3 4 5; do ldapwhoami -x -D uid=mallory,ou=people,dc=example,dc=com -w guess; done
ldap_bind: Invalid credentials (49)
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=mallory,ou=people,dc=example,dc=com \
     pwdFailureTime pwdAccountLockedTime
dn: uid=mallory,ou=people,dc=example,dc=com
pwdFailureTime: [...]
pwdAccountLockedTime: [...]
```

```
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw
ldap_bind: Invalid credentials (49)
```

The right password gets exactly what a wrong one gets. The safe default: "account locked" also says "this account exists".

Task: Tell them

`ppolicy_use_lockout` makes a locked account say so. Worth it when your users are real people and "invalid credentials" sends them to the help desk.

Turn it on, and bind as mallory again.

Solution: Tell them

In `slapd.conf`, in the ppolicy section, then restart slapd:

```
+ppolicy_use_lockout
```

Or live, in `cn=config`:

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
changetype: modify
replace: olcPPolicyUseLockout
olcPPolicyUseLockout: TRUE
EOF
modifying entry "olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config"
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw
ldap_bind: Invalid credentials (49); Account locked
```

Task: Unlocking

Account unlocks when `pwdAccountLockedTime` lapses or by an administrator deleting `pwdAccountLockedTime`.

Unlock mallory. Then try `frank`: an administrator locked him.

Solution: Unlocking

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: uid=mallory,ou=people,dc=example,dc=com
changetype: modify
delete: pwdAccountLockedTime
EOF
modifying entry "uid=mallory,ou=people,dc=example,dc=com"
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw
dn:uid=mallory,ou=people,dc=example,dc=com
$ ldapwhoami -x -e ppolicy -D uid=frank,ou=people,dc=example,dc=com -w frank-pw
ldap_bind: Invalid credentials (49); Account locked
```

frank's lock is `000001010000Z`: a special value for "locked by an administrator". This one never expires.

Task: Bounding it

- **Writes**: every failed bind is a write, and anyone can bind. `pwdMaxRecordedFailure` caps the failures kept (but at least `pwdMaxFailure`). Set it to 5 on `cn=default`.
- **Time**: `pwdLockoutDuration` (300) ends a lock on its own. Make it 10 seconds, lock mallory, check `pwdAccountLockedTime`, wait, and put it back.

Check: Bounding it

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=default,ou=policies,dc=example,dc=com
changetype: modify
add: pwdMaxRecordedFailure
pwdMaxRecordedFailure: 5
EOF
modifying entry "cn=default,ou=policies,dc=example,dc=com"
```

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=default,ou=policies,dc=example,dc=com
changetype: modify
replace: pwdLockoutDuration
pwdLockoutDuration: 10
EOF
modifying entry "cn=default,ou=policies,dc=example,dc=com"
$ for i in 1 2 3 4 5; do ldapwhoami -x -D uid=mallory,ou=people,dc=example,dc=com -w guess; done
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw
ldap_bind: Invalid credentials (49); Account locked
$ sleep 11
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw
dn:uid=mallory,ou=people,dc=example,dc=com
```

Solution: Put it back

The rest of the day depends on it:

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=default,ou=policies,dc=example,dc=com
changetype: modify
replace: pwdLockoutDuration
pwdLockoutDuration: 300
EOF
modifying entry "cn=default,ou=policies,dc=example,dc=com"
```

3. Slowing attackers down

instead of locking people out

The trouble with lockout

Anyone who knows a username can lock its owner out, at will.

The alternative: make every wrong guess **cost time**.

- `pwdMinDelay`: the wait after a failure, doubling with each one...
- `pwdMaxDelay`: ...up to this
- the wait is stamped as `pwdAccountTmpLockoutEnd`, and until then **even the right password is refused**

1, 2, 4, 4, 4 seconds: nothing to a user who mistyped, ruinous for a guessing attacker.

Note: This "return immediately and force a temporary lockout" is OpenLDAP's implementation. Others might actually delay the Bind response. Check before you rely on this.

Task: A policy that delays

Create `cn=hardened,ou=policies`: one-second minimum delay, four-second maximum, and never a permanent lock.

Try it on mallory: `pwdPolicySubentry` written on an entry names its policy, overriding everything else.

Solution: A policy that delays

```
$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=hardened,ou=policies,dc=example,dc=com
objectClass: namedPolicy
objectClass: pwdPolicy
cn: hardened
description: Delay, never lock. See stage 3.
pwdAttribute: userPassword
pwdMaxAge: 31536000
pwdMinLength: 8
pwdLockout: TRUE
pwdMaxFailure: 0
pwdMinDelay: 1
pwdMaxDelay: 4
EOF
adding new entry "cn=hardened,ou=policies,dc=example,dc=com"
```

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: uid=mallory,ou=people,dc=example,dc=com
changetype: modify
replace: pwdPolicySubentry
pwdPolicySubentry: cn=hardened,ou=policies,dc=example,dc=com
EOF
modifying entry "uid=mallory,ou=people,dc=example,dc=com"
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=mallory,ou=people,dc=example,dc=com pwdPolicySubentry
dn: uid=mallory,ou=people,dc=example,dc=com
pwdPolicySubentry: cn=hardened,ou=policies,dc=example,dc=com
```

Task: Guess, and wait

One wrong guess as mallory, then the right password straight away. Look at her entry; try deleting the stamp as root. Wait two seconds and try again.

Check: Guess, and wait

```
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w guess
ldap_bind: Invalid credentials (49)
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw
ldap_bind: Invalid credentials (49); Account locked
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=mallory,ou=people,dc=example,dc=com pwdAccountTmpLockoutEnd
pwdAccountTmpLockoutEnd: [...]
$ sleep 2
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw
dn:uid=mallory,ou=people,dc=example,dc=com
```

```
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w guess
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: uid=mallory,ou=people,dc=example,dc=com
changetype: modify
delete: pwdAccountTmpLockoutEnd
EOF
ldap_modify: Constraint violation (19)
$ sleep 2
$ ldapwhoami -x -e ppolicy -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw
dn:uid=mallory,ou=people,dc=example,dc=com
```

Nobody can cut a backoff short, not even root: you wait.

Solution: Delays and lockout together

On a policy that also locks, failures below `pwdMaxFailure` are delayed (rate-limiting attacks) and then it locks out properly.

mallory goes back to the default; `cn=hardened` stays:

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: uid=mallory,ou=people,dc=example,dc=com
changetype: modify
delete: pwdPolicySubentry
EOF
modifying entry "uid=mallory,ou=people,dc=example,dc=com"
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=mallory,ou=people,dc=example,dc=com pwdPolicySubentry
pwdPolicySubentry: cn=default,ou=policies,dc=example,dc=com
```

4. Quality

What counts as a good password?

What the server can check

Only what it can see: `ppolicy_hash_cleartext` lets a client send the new password in the clear, to be hashed on the way in.

Attribute	
`pwdMinLength`	8 in the seed
`pwdMaxLength`	keeps a megabyte away from the hash; not for quality
`pwdInHistory`	password reuse mitigation, the last 3 cannot be reused
`pwdMinAge`	makes the reuse mitigation actually work
`pwdSafeModify`	the old password must accompany the new
`pwdAllowUserChange`	FALSE when a provisioning system owns it, e.g. for services
`pwdCheckQuality`	`2`: refuse a password that cannot be checked (pre-hashed?)

Task: Try them

As `peggy`: set a seven-character password; change it properly, and try to change it back.

As `uid=admin`, write a ready-made hash into her `userPassword`.

Check: Try them

```
$ ldappasswd -x -e ppolicy -D uid=peggy,ou=people,dc=example,dc=com -w peggy-pw -s Ab1@efg
ppolicy: error=6 (Password is too short for policy)
$ ldappasswd -x -e ppolicy -D uid=peggy,ou=people,dc=example,dc=com -w peggy-pw -s Interlude-Falcon-33
$ ldappasswd -x -e ppolicy -D uid=peggy,ou=people,dc=example,dc=com -w Interlude-Falcon-33 -s peggy-pw
Additional info: Password is in history of old passwords
```

```
$ ldapmodify -x -D uid=admin,ou=administrators,dc=example,dc=com -w admin-pw <<EOF
dn: uid=peggy,ou=people,dc=example,dc=com
changetype: modify
replace: userPassword
userPassword: {SSHA}By44dgO7UjY2IXJqEq0nC/1Z1TxS/SnJ
EOF
ldap_modify: Constraint violation (19)
```

The hash is refused under `pwdCheckQuality: 2`. With `1` it goes through, and a provisioning system can push anything past every rule.

Note: Any password rules are always bypassed by a password administrator!

A check module: ppm

Anything cleverer than length is a loadable module. OpenLDAP ships **ppm**: character classes, forbidden characters, no username in the password.

- `ppolicy_check_module`: load it, for the whole overlay
- `pwdPolicyChecker`: an auxiliary objectClass a policy must have, to...
- `pwdUseCheckModule: TRUE` 2.6: ...use it, and...
- `pwdCheckModuleArg`: ...configure it

```
minQuality 3           at least 3 of the 4 classes below
checkRDN 1             the uid may not appear in the password
class-upperCase ABCDEFGHIJKLMNOPQRSTUVWXYZ 0 1 0
class-lowerCase abcdefghijklmnopqrstuvwxyz 0 1 0
class-digit 0123456789 0 1 0
class-special <>,?;.:/!%*^$&~"#'{([-|`_@)]=}+ 0 1 0
```

Task: Load ppm

Load `/opt/symas/lib/openldap/ppm.so`, and have `cn=default` use it with the configuration on the last slide. Then try some passwords as peggy.

Solution: Load ppm changed in 2.6

`slapd.conf` (and restart), or `cn=config`:

```
+ppolicy_check_module /opt/symas/lib/openldap/ppm.so
```

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
changetype: modify
add: olcPPolicyCheckModule
olcPPolicyCheckModule: /opt/symas/lib/openldap/ppm.so
EOF
modifying entry "olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config"
```

Solution: ...and use it

Several lines of configuration, so base64 (`base64 -w0 ppm.conf`):

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=default,ou=policies,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: pwdPolicyChecker
-
add: pwdUseCheckModule
pwdUseCheckModule: TRUE
-
add: pwdCheckModuleArg
pwdCheckModuleArg:: bWluUXVhbGl0eSAzCmNoZWNrUkROIDEKZm9yYmlkZGVuQ2hhcnMKbWF4Q29uc2VjdXRpdmVQZXJDbGFzcyAwCnVzZUNyYWNrbGliIDAKY2xhc3MtdXBwZXJDY
EOF
modifying entry "cn=default,ou=policies,dc=example,dc=com"
```

Check: ppm at work

```
$ ldappasswd -x -e ppolicy -D uid=peggy,ou=people,dc=example,dc=com -w Interlude-Falcon-33 -s onlylowercaselett
Additional info: [...] does not pass required number of strength checks (1 of 3)
$ ldappasswd -x -e ppolicy -D uid=peggy,ou=people,dc=example,dc=com -w Interlude-Falcon-33 -s Peggy-Is-Here-42
Additional info: [...] contains tokens from the RDN
$ ldappasswd -x -e ppolicy -D uid=peggy,ou=people,dc=example,dc=com -w Interlude-Falcon-33 -s Thunder-Wallaby-4
$ ldapwhoami -x -D uid=peggy,ou=people,dc=example,dc=com -w Thunder-Wallaby-42
dn:uid=peggy,ou=people,dc=example,dc=com
```

Too few classes, and her own name: both refused.

5. Targeted assignment

The right policy for each account 2.7

Check: Something is already broken

The service account `backup` cannot log in. Why?

```
$ ldapwhoami -x -e ppolicy -D uid=backup,ou=applications,dc=example,dc=com -w backup-pw
ldap_bind: Invalid credentials (49); Password expired
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=backup,ou=applications,dc=example,dc=com pwdPolicySubentry
pwdPolicySubentry: cn=default,ou=policies,dc=example,dc=com
```

One policy for everybody: the backup agent's password expired like a person's, an outage at 3am on a Sunday. `web01` and `db01` too.

ppolicy_rules 2.7

Rules in the configuration pick a policy for each entry:

```
ppolicy_rules dn.subtree="ou=applications,dc=example,dc=com"
              policy_dn="cn=service,ou=policies,dc=example,dc=com"
```

- **where**: `dn.base`, `dn.one`, `dn.subtree`, `dn.regex`
- **what else**: `filter=`, `group/<objectClass>/<attr>=`
- `require_password=` (default yes): skip entries without a password
- **tried in order; the first match wins**, unless its action is `continue`
- `pwdPolicySubentry` on the entry still beats them all but this is now discouraged

In `cn=config` each rule is an entry below the overlay's, objectClasses **`olcPPolicyScopedRule`**/**`olcPPolicyRegexRule`**.

Task: A policy for services

No expiry and **no lockout** since a service account lockout is a denial of service. Twenty characters minimum, no history.

Create `cn=service`, and send `ou=applications` and `ou=hosts` to it.

Solution: A policy for services

```
$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=service,ou=policies,dc=example,dc=com
objectClass: namedPolicy
objectClass: pwdPolicy
cn: service
description: Service accounts. Never expires, never locks.
pwdAttribute: userPassword
pwdLockout: FALSE
pwdMaxFailure: 0
pwdMinLength: 20
pwdInHistory: 0
EOF
adding new entry "cn=service,ou=policies,dc=example,dc=com"
```

In `slapd.conf`, at the end of the ppolicy section:

```
+ppolicy_rules dn.subtree="ou=applications,dc=example,dc=com" policy_dn="cn=service,ou=policies,dc=example,dc=com"
+ppolicy_rules dn.subtree="ou=hosts,dc=example,dc=com" policy_dn="cn=service,ou=policies,dc=example,dc=com"
```

Solution: ...or the rules in cn=config

```
$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=service,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: service
olcPPolicyRuleObject: ou=applications,dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRulePolicy: cn=service,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop

dn: cn=hosts,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: hosts
olcPPolicyRuleObject: ou=hosts,dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRulePolicy: cn=service,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop
EOF
adding new entry "cn=service,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config"
adding new entry "cn=hosts,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config"
$ ldapwhoami -x -e ppolicy -D uid=backup,ou=applications,dc=example,dc=com -w backup-pw
dn:uid=backup,ou=applications,dc=example,dc=com
$ ldapwhoami -x -e ppolicy -D cn=web01,ou=hosts,dc=example,dc=com -w web01-pw
dn:cn=web01,ou=hosts,dc=example,dc=com
```

Task: Partners, and the pager

- `ou=external` subtree: 90-day expiry, three strikes and 15 minutes out, twelve characters
- members of `cn=oncall` (rfc2307bis `groupOfMembers`, not the default `groupOfNames`): 90 days, three strikes, sixteen characters

Solution: Partners

```
$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=external,ou=policies,dc=example,dc=com
objectClass: namedPolicy
objectClass: pwdPolicy
cn: external
description: Partner accounts. Short leash.
pwdAttribute: userPassword
pwdMaxAge: 7776000
pwdLockout: TRUE
pwdMaxFailure: 3
pwdLockoutDuration: 900
pwdMinLength: 12
EOF
adding new entry "cn=external,ou=policies,dc=example,dc=com"
```

Solution: The pager

```
$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=privileged,ou=policies,dc=example,dc=com
objectClass: namedPolicy
objectClass: pwdPolicy
cn: privileged
description: On-call staff. Tighter than ordinary users.
pwdAttribute: userPassword
pwdMaxAge: 7776000
pwdLockout: TRUE
pwdMaxFailure: 3
pwdMinLength: 16
EOF
adding new entry "cn=privileged,ou=policies,dc=example,dc=com"
```

Solution: Their rules

```
+ppolicy_rules dn.subtree="ou=external,dc=example,dc=com" policy_dn="cn=external,ou=policies,dc=example,dc=com"
+ppolicy_rules dn.subtree="ou=people,dc=example,dc=com" group/groupOfMembers/member="cn=oncall,ou=groups,dc=example,dc=com" policy_dn="
```

```
$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=external,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: external
olcPPolicyRuleObject: ou=external,dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRulePolicy: cn=external,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop

dn: cn=oncall,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: oncall
olcPPolicyRuleObject: ou=people,dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRuleGroup: cn=oncall,ou=groups,dc=example,dc=com
olcPPolicyRuleGroupOC: groupOfMembers
olcPPolicyRuleGroupAttr: member
olcPPolicyRulePolicy: cn=privileged,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop
EOF
adding new entry "cn=external,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config"
adding new entry "cn=oncall,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config"
```

Check: Who gets what

```
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=victor,ou=external,dc=example,dc=com pwdPolicySubentry
pwdPolicySubentry: cn=external,ou=policies,dc=example,dc=com
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=dave,ou=people,dc=example,dc=com pwdPolicySubentry
pwdPolicySubentry: cn=privileged,ou=policies,dc=example,dc=com
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=alice,ou=people,dc=example,dc=com pwdPolicySubentry
pwdPolicySubentry: cn=default,ou=policies,dc=example,dc=com
```

Also worth knowing:

- `filter="(employeeType=contractor)"` matches entries accordingly, only stored attributes
- `continue` action type records a match and keeps looking: broad first, narrow later
- `dn.regex` + `policy_dn.expand` (or `group.expand`) derives the policy (group) from the DN

Task: Retire ppolicy_default

It is deprecated. Unless you use action `continue`, best replaced with a rule at the **end** that catches everything with a password. Nobody's policy should change, and heidi, who has no password, should still have none.

From then on, new rules go *before* the catch-all: after it they are never reached.

Solution: Retire ppolicy_default

```
-ppolicy_default "cn=default,ou=policies,dc=example,dc=com"
+ppolicy_rules dn.subtree="dc=example,dc=com" require_password=true policy_dn="cn=default,ou=policies,dc=example,dc=com"
```

```
$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=default,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: default
olcPPolicyRuleObject: dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRuleRequirePassword: TRUE
olcPPolicyRulePolicy: cn=default,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop
EOF
adding new entry "cn=default,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config"
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
changetype: modify
delete: olcPPolicyDefault
EOF
modifying entry "olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config"
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=alice,ou=people,dc=example,dc=com pwdPolicySubentry
pwdPolicySubentry: cn=default,ou=policies,dc=example,dc=com
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=heidi,ou=people,dc=example,dc=com pwdPolicySubentry
dn: uid=heidi,ou=people,dc=example,dc=com
```

Check: The result

```
$ ldapsearch -Q -Y EXTERNAL -LLL -s one -b olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config \
     "(objectClass=*)" olcPPolicyRulePolicy
dn: cn={0}service,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
olcPPolicyRulePolicy: cn=service,ou=policies,dc=example,dc=com
dn: cn={1}hosts,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
olcPPolicyRulePolicy: cn=service,ou=policies,dc=example,dc=com
dn: cn={2}external,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
olcPPolicyRulePolicy: cn=external,ou=policies,dc=example,dc=com
dn: cn={3}oncall,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
olcPPolicyRulePolicy: cn=privileged,ou=policies,dc=example,dc=com
dn: cn={4}default,olcOverlay={2}ppolicy,olcDatabase={2}mdb,cn=config
olcPPolicyRulePolicy: cn=default,ou=policies,dc=example,dc=com
```

6. Lifecycle and idleness

Accounts that should not be used, yet or any more

Three ways to say "not this account"

Attribute	Use case	Effect
`pwdAccountLockedTime`	a punishment	cleared by an administrator (stage 2)
`pwdMaxIdle`	disuse	judged at every bind
`pwdStartTime`, `pwdEndTime`	a schedule	known in advance

They are not interchangeable, a lock/idleness is reset by a password reset, start/end specify absolute validity (someone left, ...) that sticks.

Task: Why can't they log in?

`ivan` and `judy` both have the right password. Find out what is stopping them, then let ivan start early.

Check: Why can't they log in?

```
$ ldapwhoami -x -e ppolicy -D uid=ivan,ou=people,dc=example,dc=com -w ivan-pw
ldap_bind: Invalid credentials (49); Account locked
$ ldapwhoami -x -e ppolicy -D uid=judy,ou=people,dc=example,dc=com -w judy-pw
ldap_bind: Invalid credentials (49); Account locked
$ ldapsearch -Q -Y EXTERNAL -LLL "(|(uid=ivan)(uid=judy))" pwdStartTime pwdEndTime
dn: uid=ivan,ou=people,dc=example,dc=com
pwdStartTime: [...]
dn: uid=judy,ou=people,dc=example,dc=com
pwdEndTime: [...]
```

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: uid=ivan,ou=people,dc=example,dc=com
changetype: modify
delete: pwdStartTime
EOF
modifying entry "uid=ivan,ou=people,dc=example,dc=com"
$ ldapwhoami -x -e ppolicy -D uid=ivan,ou=people,dc=example,dc=com -w ivan-pw
dn:uid=ivan,ou=people,dc=example,dc=com
```

ivan starts next month; judy left last month, her password untouched and the reason on record: the honest way to disable an account.

Task: Idle accounts 2.6

`pwdMaxIdle` is judged from `pwdLastSuccess`, and nothing writes that until **lastbind** is on, on the database. That makes every login a write, so `lastbind-precision` says how stale it may get.

Turn lastbind on (not the overlay, please don't use the overlay anymore), with a precision of an hour. Then expire the on-call staff (`cn=privileged`) after 90 days without a login: `dave` last logged in 200 days ago.

Solution: Idle accounts

`slapd.conf`, in the database section (and restart), or `cn=config`:

```
+lastbind on
+lastbind-precision 3600
```

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: olcDatabase={2}mdb,cn=config
changetype: modify
replace: olcLastBind
olcLastBind: TRUE
-
replace: olcLastBindPrecision
olcLastBindPrecision: 3600
EOF
modifying entry "olcDatabase={2}mdb,cn=config"
$ ldapwhoami -x -e ppolicy -D uid=alice,ou=people,dc=example,dc=com -w alice-pw
dn:uid=alice,ou=people,dc=example,dc=com
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=alice,ou=people,dc=example,dc=com pwdLastSuccess
pwdLastSuccess: [...]
```

Check: Idle accounts

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=privileged,ou=policies,dc=example,dc=com
changetype: modify
add: pwdMaxIdle
pwdMaxIdle: 7776000
EOF
modifying entry "cn=privileged,ou=policies,dc=example,dc=com"
$ ldapwhoami -x -e ppolicy -D uid=dave,ou=people,dc=example,dc=com -w dave-pw
ldap_bind: Invalid credentials (49); Account locked
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=dave,ou=people,dc=example,dc=com pwdLastSuccess
pwdLastSuccess: [...]
```

7. Hashing

Better hashes, upgraded as people log in 2.7

What is stored today

```
$ ldapsearch -Q -Y EXTERNAL -LLL -o ldif_wrap=no -s base -b uid=alice,ou=people,dc=example,dc=com userPassword
userPassword:: e1NTSEF9[...]
$ ldapsearch -Q -Y EXTERNAL -LLL -o ldif_wrap=no -s base -b uid=oscar,ou=people,dc=example,dc=com userPassword
userPassword:: b3NjYXItcHc=
```

`userPassword` is "not printable", so it comes back in base64: `e1NTSEF9` (= `{SSHA}`), respectable in 2005. And oscar's `b3NjYXItcHc=` (= literally `oscar-pw` in the clear), there will be one in every directory that has run long enough unless enforced otherwise.

Two new attributes 2.7

On the `pwdHashingPolicy` objectClass, which a policy has to opt in to:

- **`pwdDefaultHash`**: the scheme for passwords set under this policy. Per policy, so people can be expensive to check and service accounts, which bind thousands of times a minute, stay cheap
- **`pwdRehashOnBind`**: upgrade a password on its next successful simple bind, the only moment the server sees it in the clear

Tip: this works best when coupled with **`pwdMaxIdle`**, by the end of the idle period you can (almost) guarantee that everyone is either locked out or migrated.

Task: Better hashes

- people (`cn=default`) get `{ARGON2}`, services (`cn=service`) `{SSHA512}`
- mallory changes her password: what is stored now?
- alice has no reason to change hers: upgrade her the next time she logs in

Solution: Better hashes

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=default,ou=policies,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: pwdHashingPolicy
-
add: pwdDefaultHash
pwdDefaultHash: {ARGON2}
EOF
modifying entry "cn=default,ou=policies,dc=example,dc=com"
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=service,ou=policies,dc=example,dc=com
changetype: modify
add: objectClass
objectClass: pwdHashingPolicy
-
add: pwdDefaultHash
pwdDefaultHash: {SSHA512}
EOF
modifying entry "cn=service,ou=policies,dc=example,dc=com"
```

Check: Better hashes

```
$ ldappasswd -x -D uid=mallory,ou=people,dc=example,dc=com -w mallory-pw -s Grapnel-Windlass-42
$ ldapsearch -Q -Y EXTERNAL -LLL -o ldif_wrap=no -s base -b uid=mallory,ou=people,dc=example,dc=com userPassword
userPassword:: e0FSR09OMn0[...]
```

```
$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=default,ou=policies,dc=example,dc=com
changetype: modify
add: pwdRehashOnBind
pwdRehashOnBind: TRUE
EOF
modifying entry "cn=default,ou=policies,dc=example,dc=com"
$ ldapwhoami -x -D uid=alice,ou=people,dc=example,dc=com -w alice-pw
dn:uid=alice,ou=people,dc=example,dc=com
$ ldapsearch -Q -Y EXTERNAL -LLL -o ldif_wrap=no -s base -b uid=alice,ou=people,dc=example,dc=com userPassword
userPassword:: e0FSR09OMn0[...]
```

`e0FSR09OMn0` is `{ARGON2}`. A rehash happens once, and only when possible: hash schemes don't match, a successful plain bind.

Gotcha: A rehash currently counts as a password change as of 2.7.1

It moves `pwdChangedTime`: carol, weeks from expiry, logs in and has a whole new year without choosing anything. Migrate a directory like this and everybody's password age is reset.

And oscar's password has expired, so he cannot log in to be rehashed: a migration suspends expiry first.

8. Replication

A bind is a write

There has been a second server all along

Open a second terminal on your **ldap2**, set the same LDAPURI and LDAPBASE. (Commands for ldap2 have an ldap2$ prompt.)

```
ldap2$ export LDAPURI=ldapi:/// LDAPBASE=dc=example,dc=com
```

```
ldap2$ ldapsearch -Q -Y EXTERNAL -LLL -b cn=config "(olcUpdateRef=*)" olcUpdateRef
olcUpdateRef: [...]
ldap2$ ldapsearch -x -D uid=admin,ou=administrators,dc=example,dc=com -w admin-pw -LLL -s base -b uid=mallory,ou=people,dc=example,dc=com u
dn: uid=mallory,ou=people,dc=example,dc=com
```

A read-only replica of ldap1; anything that would write here is referred to that URL. You want to keep it:

```
export PROVIDER=ldaps://ldap1.example.net/    # your olcUpdateRef, exactly
```

Again for cn=config, convert first:

```
ldap2$ cd /opt/symas/etc/openldap && mkdir slapd.d
ldap2$ slaptest -f slapd.conf -F slapd.d && chown -R openldap: slapd.d
ldap2$ systemctl restart symas-openldap-server
```

Check: Guess against the replica

A failed bind is a write, and a replica is read-only. Guess mallory's password on ldap2: what does ldap1 know about it?

```
ldap2$ for i in 1 2; do ldapwhoami -x -D uid=mallory,ou=people,dc=example,dc=com -w guess; done
ldap_bind: Invalid credentials (49)
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=mallory,ou=people,dc=example,dc=com pwdFailureTime
dn: uid=mallory,ou=people,dc=example,dc=com
```

Nothing. Spread the guesses over the replicas and you get free guesses.

Task: Fix it

In this set up configuration does not replicate. The policy entries replicated from ldap1 but ldap2's ppolicy configuration is still the same as it was 2 hours ago. Adjust these to match ldap1: rules, lockout disclosure, ppm, lastbind.

Then send the writes upstream. Three settings that only work together:

- **`updateref`**: where to find the writable copy is (already set up)
- **`overlay chain`**: follows that referral instead of returning it, as `uid=updater` (ldap1's ACLs already let it write policy state)
- **`ppolicy_forward_updates`**: let ppolicy use it for its own writes

Solution: Fix it, in slapd.conf

On ldap2 (`lastbind` in the database section). `slapd.conf` does not expand `$PROVIDER`: type your ldap1's URL.

```
-ppolicy_default "cn=default,ou=policies,dc=example,dc=com"
+ppolicy_use_lockout
+ppolicy_check_module /opt/symas/lib/openldap/ppm.so
+ppolicy_rules dn.subtree="ou=applications,dc=example,dc=com" policy_dn="cn=service,ou=policies,dc=example,dc=com"
+ppolicy_rules dn.subtree="ou=hosts,dc=example,dc=com" policy_dn="cn=service,ou=policies,dc=example,dc=com"
+ppolicy_rules dn.subtree="ou=external,dc=example,dc=com" policy_dn="cn=external,ou=policies,dc=example,dc=com"
+ppolicy_rules dn.subtree="ou=people,dc=example,dc=com" group/groupOfMembers/member="cn=oncall,ou=groups,dc=example,dc=com" policy_dn="
+ppolicy_rules dn.subtree="dc=example,dc=com" require_password=true policy_dn="cn=default,ou=policies,dc=example,dc=com"
+lastbind on
+lastbind-precision 3600
```

```
@@ before moduleload
+moduleload back_ldap
@@ before database
+overlay chain
+chain-uri "$PROVIDER"
+chain-idassert-bind bindmethod=simple binddn="uid=updater,ou=administrators,dc=example,dc=com" credentials="updater-pw" mode=none tls_
+chain-return-error TRUE
@@ before ppolicy_rules
+ppolicy_forward_updates
```

Solution: Fix it, in cn=config (#1/5)

On ldap2 the overlay is `olcOverlay={0}ppolicy,olcDatabase={1}mdb`: different numbers from ldap1's.

```
ldap2$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config
changetype: modify
replace: olcPPolicyUseLockout
olcPPolicyUseLockout: TRUE
-
add: olcPPolicyCheckModule
olcPPolicyCheckModule: /opt/symas/lib/openldap/ppm.so
-
delete: olcPPolicyDefault

dn: olcDatabase={1}mdb,cn=config
changetype: modify
replace: olcLastBind
olcLastBind: TRUE
-
replace: olcLastBindPrecision
olcLastBindPrecision: 3600
EOF
modifying entry "olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config"
modifying entry "olcDatabase={1}mdb,cn=config"
```

Solution: Fix it, in cn=config: the rules (#2/5)

```
ldap2$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=service,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: service
olcPPolicyRuleObject: ou=applications,dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRulePolicy: cn=service,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop

dn: cn=hosts,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: hosts
olcPPolicyRuleObject: ou=hosts,dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRulePolicy: cn=service,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop

dn: cn=external,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: external
olcPPolicyRuleObject: ou=external,dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRulePolicy: cn=external,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop
EOF
adding new entry "cn=service,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config"
adding new entry "cn=hosts,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config"
adding new entry "cn=external,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config"
```

Solution: Fix it, in cn=config: the rules (#3/5)

```
ldap2$ ldapadd -Q -Y EXTERNAL <<EOF
dn: cn=oncall,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: oncall
olcPPolicyRuleObject: ou=people,dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRuleGroup: cn=oncall,ou=groups,dc=example,dc=com
olcPPolicyRuleGroupOC: groupOfMembers
olcPPolicyRuleGroupAttr: member
olcPPolicyRulePolicy: cn=privileged,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop

dn: cn=default,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config
objectClass: olcPPolicyScopedRule
cn: default
olcPPolicyRuleObject: dc=example,dc=com
olcPPolicyRuleScope: sub
olcPPolicyRuleRequirePassword: TRUE
olcPPolicyRulePolicy: cn=default,ou=policies,dc=example,dc=com
olcPPolicyRuleAction: stop
EOF
adding new entry "cn=oncall,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config"
adding new entry "cn=default,olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config"
ldap2$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=backup,ou=applications,dc=example,dc=com pwdPolicySubentry
pwdPolicySubentry: cn=service,ou=policies,dc=example,dc=com
```

Solution: Fix it, in cn=config: the chain (#4/5)

```
ldap2$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: cn=module{0},cn=config
changetype: modify
add: olcModuleLoad
olcModuleLoad: back_ldap
EOF
modifying entry "cn=module{0},cn=config"
```

```
ldap2$ ldapadd -Q -Y EXTERNAL <<EOF
dn: olcOverlay=chain,olcDatabase={-1}frontend,cn=config
objectClass: olcOverlayConfig
objectClass: olcChainConfig
olcOverlay: chain
olcChainReturnError: TRUE

dn: olcDatabase=ldap,olcOverlay={0}chain,olcDatabase={-1}frontend,cn=config
objectClass: olcLDAPConfig
objectClass: olcChainDatabase
olcDatabase: ldap
EOF
adding new entry "olcOverlay=chain,olcDatabase={-1}frontend,cn=config"
adding new entry "olcDatabase=ldap,olcOverlay={0}chain,olcDatabase={-1}frontend,cn=config"
```

Solution: Fix it, in cn=config: the chain (#5/5)

```
ldap2$ ldapadd -Q -Y EXTERNAL <<EOF
dn: olcDatabase=ldap,olcOverlay={0}chain,olcDatabase={-1}frontend,cn=config
objectClass: olcLDAPConfig
objectClass: olcChainDatabase
olcDatabase: ldap
olcDbURI: "$PROVIDER"
olcDbIDAssertBind: bindmethod=simple
  binddn="uid=updater,ou=administrators,dc=example,dc=com"
  credentials="updater-pw" mode=none
  tls_reqcert=demand tls_cacertdir=/etc/ssl/certs/
EOF
adding new entry "olcDatabase=ldap,olcOverlay={0}chain,olcDatabase={-1}frontend,cn=config"
```

```
ldap2$ ldapmodify -Q -Y EXTERNAL <<EOF
dn: olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config
changetype: modify
replace: olcPPolicyForwardUpdates
olcPPolicyForwardUpdates: TRUE
EOF
modifying entry "olcOverlay={0}ppolicy,olcDatabase={1}mdb,cn=config"
```

Check: Relevant information is flowing again

```
ldap2$ ldapwhoami -x -D uid=mallory,ou=people,dc=example,dc=com -w guess
ldap_bind: Invalid credentials (49)
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=mallory,ou=people,dc=example,dc=com pwdFailureTime
pwdFailureTime: [...]
ldap2$ ldapwhoami -x -D uid=alice,ou=people,dc=example,dc=com -w alice-pw
dn:uid=alice,ou=people,dc=example,dc=com
$ ldapsearch -Q -Y EXTERNAL -LLL -s base -b uid=alice,ou=people,dc=example,dc=com pwdLastSuccess
pwdLastSuccess: [...]
```

A failure on ldap2 is recorded on ldap1, and a login on ldap2 keeps an account from looking idle (stage 6).

That's it

You've learned how to turn most requirements you'll see into a workable setup

In principle (there is no HA in this example), this is close to what you would see in a production directory environment that is easy to manage and audit while allowing replicas at each site.

Thank you for attending!

Useful links:

- `man slapo-ppolicy`: https://openldap.org/software/man.cgi?query=slapo-ppolicy
- Relevant UK NCSC guidance: https://www.ncsc.gov.uk/collection/passwords

Also note that sometimes passwords are discouraged, talk to us about OpenLDAP and OAuth2/OIDC.