LDAP и Все-Все-Все.

Барабанов Алексей, alekseybb@mail.ru , Москва
2004.01.11 черновик 1
2004.07.16 черновик 2

Примечание ко 2-му черновику: *Этот вариант представляет 3 первых главы и 3 приложения из плановых 8 глав. Если какие-то выводы кажутся недостаточно совершенными, то может ответ ожидается именно в еще не представленных здесь главах.*

В этой работе не ставится цель подробно, с нуля и всесторонне рассмотреть настройку LDAP в среде Linux, так как этот вопрос достаточно освящен в многочисленных публикациях на тему LDAP. И эта работа не заменит чтения документации. Здесь же будет уделено внимание тем сторонам функционирования LDAP, что на взгляд автора остались недосказанными в традиционном изложении темы. Главный акцент будет сделан на решении некоторых проблем, возникающих в процессе настройки типовой конфигурации, предлагаемой всевозможными руководствами, под нужды конкретной задачи. А также одной из целей будет изъятие из популярных шаблонных литературных настроек избыточных элементов.

Не следует думать, что автор в чем-то желает оппонировать написанному до него. Так же не надо принимать рекомендации не следовать вендорским рецептам, как указание на некомпетентность авторов этих рецептов. Не надо думать, что добавленные от SuSE предложения по настройке NSS LDAP и PAM LDAP служат для ослабления защищенности серверов и тотальная миграция системных баз с помощью скриптов с PADL Software Pty Ltd задумана с целью создания возможности атаки через подмену системных счетов в базе LDAP. Конечно же, нет. Следует воспринимать такие вещи, как негласный “заговор” профессионалов, предпочитающих скорее эксплуатировать проблемы, чем решать их кардинально. В Мире Open Source это обычное явление, встречающееся столь же часто, как неполная документированность проектов.

Последнее, о чем стоит договориться до прочтения столь “опасного” документа то, что автор не несет ответственность за результаты применения описанных здесь рекомендаций. И, тем не менее, желательно перед модификацией какого-либо файла предварительно сделать сохранение его исходного значения в одноименной копии с суффиксом *.orig . Или, как иногда рекомендуют, произвести полный бэкап рабочей системы. Это позволит, в случае неудачи, вернуть все настройки к первоначальному состоянию.

Оглавление

Глава 1. Введение.

1.1.Общие соглашения.

Определимся с некоторыми исходными понятиями. Будем считать, что этот опус предназначен для администраторов корпоративных сетей. Эта аудитория, как ожидается, владеет основными понятиями и даже углубленными аспектами использования Linux. Поэтому, что такое LDAP не требует дополнительного объяснения и для полноты картины достаточно договорится о том, что DN (Distinguished Name) это именно Уникальное Имя, а не что-то иное, как утверждают некоторые переводы. И тем более не станут проблемой иногда неформатно завернутые строки в скриншотах. Далее, примем, что работы выполняются на полностью настроенном сервере SuSE Linux v.9.0, на котором уже установлены все необходимые пакеты, и который содержит все необходимые сетевые службы, настроенные, как требуется. Точно также условимся, что все скриншоты и иные протоколы и выводы создавались и являются верными только в среде SuSE v.9.0.

Предположим, что внутренняя сеть 192.168.0.0/24. Сервер, он же шлюз – 192.168.0.1. Внутренний домен office.localnet . Соответственно полное доменное имя (FQDN) сервера будет server.office.localnet . Все остальные службы настроены из предположения работы именно в составе указанной внутренней среды.

При создании базы LDAP будем исходить из такого же предположения. Таким образом, корневым объектом нашей базы будет dc=office, dc=localnet, что дословно значит domain_component=office, domain_component=localnet.

Для создания базы изначально используется специальный бюджет, указываемый в директиве настройки LDAP как rootdn, то есть root distinguished name, который существует лишь в среде LDAP, а не в локальной системе. Этот привилегированный пользователь LDAP присутствует в пустой базе LDAP сразу после ее определения. И все первоначальные изменения производятся только от него. Хотя в дальнейшем, после создания новых пользователей базы, можно производить модификации от других пользователей, но пользователь, описанный в rootdn, остается всегда актуальным привилегированным пользователем с абсолютным доступом к любым записям и полям базы LDAP, независимо от назначенных прав в настройках сервера. Другими словами, упоминание пользователя из rootdn в ограничениях доступа к базе LDAP лишено всякого смысла. В нашем случае определим DN такого суперпользователя как cn=ldapadmin, dc=office, dc=localnet. В дальнейшем можно будет вообще изъять этого пользователя из конфигурации сервера. Это даже и рекомендовано. Но не будем этого делать до окончательной настройки.

Позднее для просмотра базы LDAP с локального клиента будет создан еще один пользователя LDAP специального только для внутреннего использования в локальном LDAP-клиенте. Этот пользователь, назовем его ldapbrowser, также не будет иметь системного счета, но будет записан в базе LDAP. Иногда такого пользователя называют посредником (proxyuser). Идея его использования состоит в том, чтобы по возможности урезать права такого пользователя и заменить им в настройках клиентов суперпользователя rootdn. Примем для начала необходимость пользователя посредника на веру, так как это почти классический ход, предлагаемый практически всеми руководствами.

Все использованные пароли специально упрощены. Реальные бюджеты должны иметь усиленные пароли, как минимум.

Все приведенные далее конфигурации сделаны в первую очередь, как иллюстрации к тексту, и верны лишь в контексте обсуждения. Если некоторое решение содержит явную ошибку, то возможно ее исправлению будут посвящены следующие главы.

Далее в тексте, атрибуты LDAP названы полями, как это принято в базах данных, чтобы не создавать путаницы с атрибутами файлов и проч.

1.2.Тривиальная настройка сервера LDAP.

Прежде всего, необходимо настроить и запустить LDAP сервер. Данному вопросу посвящено слишком много литературы. Процесс описан детально и даже противоречиво. Можно воспользоваться многочисленными руководствами по настройке Samba3 с LDAP. Для начала выберем самый простой вариант настройки. Далее приведено содержимое файла конфигурации LDAP с минимальным набором схем и не содержащее никаких ограничений доступа, с которым работает сервер, использованный для экспериментов с настройкой аутентификации.

```
server:~ # cat /etc/openldap/slapd.conf | grep -v ^# | grep -v ^$
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/nis.schema
include         /etc/openldap/schema/samba3.schema
pidfile         /var/run/slapd/slapd.pid
argsfile        /var/run/slapd/slapd.args
modulepath      /usr/lib/openldap/modules
replogfile      /var/lib/ldap/replica.log
database        ldbm
cachesize       40000
dbcachesize     60000000
suffix          "dc=office,dc=localnet"
rootdn          "cn=ldapadmin,dc=office,dc=localnet"
rootpw          {SSHA}K6n0nTsvOWxO1xPGBN5HoZCAsaO0wV7p
directory       /var/lib/ldap
index   objectClass                             eq
index   ou,cn,sn,displayName                    eq,pres,sub
index   uidNumber,gidNumber                     eq
index   sambaSID                                eq
index   memberUID,uid                           eq,pres,sub
index   sambaPrimaryGroupSID                    eq
index   sambaDomainName                         eq
index   default                                 sub
server:~ #
```

Для rootdn парольной фразой выбрано слово “secret”. В slapd.conf записан хеш парольной фразы. И LDAP в процессе проверки аутентификации сверяет с образцом тот хеш, который будет получен от пароля клиента. Функция хеширования обеспечивает однозначность соответствия “пароль” - “хеш”. Но в сервер LDAP клиентом передается именно пароль, а не хеш, хотя путешествие отрытых

парольных фраз от клиента к серверу вместо хешей подвергает пароли риску перехвата.

Индексация выбрана та, что предложена для работы с Samba3. В данной точке рассмотрения это непринципиальный вопрос. Индексы ускоряют, но не меняют ни информации, ни порядка доступа к ней.

Сервер LDAP представлен парой демонов. Первый slapd, который собственно и есть LDAP, и второй slurpd, который является сервером репликатором. Сейчас будем настраивать только первый из них. Бинарные файлы серверов располагаются в нетрадиционном месте в /usr/lib/openldap. Файлы с настройками, используемыми при старте сервера, располагаются в /etc/openldap . Если сервер LDAP использует файловую базу, то она создается в /var/lib/ldap. Системный идентификатор запущенного сервера, параметры его запуска и некоторые дополнительные файлы размещаются в директории /var/run/slapd. Выбранные пути заданы при вендорской модификации и сборке пакета openldap2-2.1.22-65.

Еще раз напоминаю, что все далее сказанное относится к дистрибутиву SuSE v.9.0. Возможно, в другой дистрибуции надо будет что-то дополнительно поискать в ином месте и настроить иным путем. Счастливым обладателям slackware вообще придется немного потерпеть и поскучать. Но так как предполагаемая аудитория это системные администраторы, а добродетель сисадмина это лень, а значит и спокойный темперамент то, надеюсь, немного SuSE-специфичной информации будет встречено с пониманием.

И так, поскольку демоны запускаются от пользователя ldap, то и права на директорию с файлами базы и на /var/run/slapd должны быть ldap.ldap . Для такого запуска в системных базах созданы соответствующие группа и пользователь.

```
server:~ # cat /etc/group | grep ^ldap:
ldap:x:70:
server:~ # cat /etc/passwd | grep ^ldap:
ldap:x:76:70:User for OpenLDAP:/var/lib/ldap:/bin/bash
server:~ # cat /etc/shadow | grep ^ldap:
ldap:!:12097:0:99999:365:::
server:~ #
```

Как видно из приведенного скриншота, бюджет ldap имеет действующий шелл, но заблокированный пароль.

Непосредственно запуск демонов LDAP при старте сервера происходит из скрипта /etc/init.d/ldap. Скрипт включается в последовательность инициализации с единственным условием, согласно LSB, Required-Start: $local_fs. Но если база LDAP размещается, например в SQL , то вероятно для правильного подключения службы через insserv необходимо будет в строку Required-Start скрипта добавить название службы выбранного сервера SQL. Команда запуска, использованная внутри скрипта, выглядит буквально следующим образом:

```
server:~ # /sbin/startproc -p /var/run/slapd/slapd.pid \
           $SLAPD_BIN -h "$SLAPD_URLS" \
           $USER_CMD \
```

```
        $GROUP_CMD \
        $OPENLDAP_SLAPD_PARAMS
```

Параметры этой команды создаются динамически на основании настроечных опций из файла /etc/sysconfig/openldap. Например, дамп одного из вариантов интерпретации настроек, самых тривиальных, полученный вставкой отладочного эха в стартовый скрипт, выглядит так:

```
server:~ # /sbin/startproc -p /var/run/slapd/slapd.pid \
          /usr/lib/openldap/slapd -h " ldap://127.0.0.1:389/" \
                                  -u ldap \
                                  -g ldap
```

Здесь указано, что сервер будет слушать только внутренний адрес и соответственно будет доступен только с внутреннего адреса. Это предпочтительный режим для проведения работ, в успехе которых нет уверенности. Файл в /etc/sysconfig/openldap следует настраивать не путем редактирования во встроенном редакторе mc, а, как и все остальные параметрические файлы в этой директории, с помощью YaST2. Все настройки очевидны и не требуют специального комментария.

Важный здесь параметр, это способ запуска сервера, который будет определять его доступность. Самый скрытый режим, это т.н. “ldap over IPC”, при котором сервер будет доступен через Unix domain сокет /var/run/slapd/ldapi. Другие варианты представляют запуск slapd как службу, ждущую соединения через TCP на определенном порту и указанном адресе.

Суммарно все настройки для начальной фазы выглядят следующим образом.

```
server:~ # cat /etc/sysconfig/openldap | grep -v ^# | grep -v ^$
OPENLDAP_START_LDAPS="no"
OPENLDAP_USER="ldap"
OPENLDAP_GROUP="ldap"
OPENLDAP_CHOWN_DIRS="yes"
OPENLDAP_START_LDAPI="no"
OPENLDAP_SLAPD_PARAMS=""
OPENLDAP_RUN_DB_RECOVER="no"
OPENLDAP_LDAP_INTERFACES="127.0.0.1:389"
OPENLDAP_LDAPS_INTERFACES="192.168.0.1:636"
server:~ #
```

Запуск сервера производится очевидной командой.

```
server:~ # rcldap start
Starting ldap-server                                            done
server:~ #
```

Для проверки проверим, что сервер слушает только там, где ему и указано, то есть на локальном адресе.

```
server:~ # netstat -apn | grep "LISTEN.*slapd"
tcp    0   0 127.0.0.1:389      0.0.0.0:*        LISTEN      1582/slapd
server:~ #
```

Теперь все готово для настройки клиентов LDAP и проверки их взаимодействия.

Глава 2. Об NSS.

Одним из ключевых элементов механизма аутентификации пользователей, а так же одним из способов получения учетной информации в системе, является NSS или Диспетчер Службы Имен (Name Service Switch). Например, запрос о том, какому пользователю принадлежат некоторых файлов с определенным uid, обрабатывается этой службой. Таким образом, при решении вопроса о uid взаимодействие NSS с LDAP заключается в том, что если в системе присутствуют некоторые фалы с uid, который не содержится в локальных системных базах, /etc/passwd, то информация ищется в LDAP. Не исключен и другой порядок поиска. Естественно все вышесказанное справедливо и в отношении других запросов, обрабатываемых NSS, поскольку определение имени владельца файла по его uid не исчерпывает всего многообразия применения NSS в Linux.

Другими словами, NSS предоставляет всем программам и службам системы информацию из системных баз через запросы getpw*, getsh*, getgr*, gethost* и прочие. И использование NSS LDAP “прозрачным” образом расширяет круг поиска, добавляя к уже имеющимся базам еще и LDAP.

Поскольку NSS, в том числе поставляет в систему аутентификационные данные, то значит, тем самым участвует в формировании политики аутентификации пользователей. Именно настройке политики в NSS LDAP и будет посвящена эта глава.

2.1.Настройка NSS LDAP.

За работу с LDAP в NSS отвечает пакет nss_ldap, который в системе представлен, как объектная библиотека.

```
server:~ # ls -l `rpm -ql nss_ldap | grep lib`
-rwxr-xr-x    1 root     root        70779 Sep 24  2003 /lib/libnss_ldap.so.2
lrwxrwxrwx    1 root     root           21 Mar 16 18:04 /
usr/lib/libnss_ldap.so -> /lib/libnss_ldap.so.2
server:~ #
```

Расположение файла с настройками ldap.conf задается при компиляции этой библиотеки. Определить путь, где его ожидает найти библиотека, можно следующим образом.

```
server:~ # strings `rpm -ql nss_ldap | grep lib` | grep ldap.conf
/etc/ldap.conf
/etc/ldap.conf
server:~ #
```

Файл ldap.conf задает параметры клиента LDAP для связи с сервером LDAP и критерии поиска учетной информации в базе. Этот файл может быть весьма простым. Вот его минимальная, работоспособная версия.

```
server:~ # cat /etc/ldap.conf | grep -v ^# | grep -v ^$
host    127.0.0.1:389
base    dc=office,dc=localnet
nss_base_passwd         ou=People,dc=office,dc=localnet?one
nss_base_shadow         ou=People,dc=office,dc=localnet?one
nss_base_group          ou=Groups,dc=office,dc=localnet?one
binddn cn=ldapadmin,dc=office,dc=localnet
bindpw secret
server:~ #
```

Здесь записан адрес и порт сервера LDAP, указан корень поисковой базы, директивы для поиска учетной информации о бюджетах, паролях и группах, DN пользователя с которым происходит сеанс связи с сервером и, самое интересное, его пароль в открытом виде. Пока будем использовать rootdn, чтобы гарантированно не иметь проблем с доступом к полям базы LDAP.

Небольшой комментарий к записи “ou=People,dc=office,dc=localnet?one”. Дословно тем сказано искать от узла ou=People,dc=office,dc=localnet на дереве LDAP, просматривая только на один уровень вниз. Если не указывать “?one”, то по умолчанию поиск происходит в режиме “sub” (от subtree), что значит на всю глубину.

Системные настройки NSS содержатся в файле /etc/nsswitch.conf. Формат этого файла прост до чрезвычайности. Важными для нас являются следующие строки, отвечающие за поиск учетной информации бюджетов, паролей и групп.

```
server:~ # cat /etc/nsswitch.conf | grep "\(^passwd\|^shadow\|^group\)"
passwd: files ldap
shadow: files ldap
group:  files ldap
server:~ #
```

Тем самым указано производить поиск в локальной базе, а потом дополнительно и в базе LDAP. Так как речь идет о чтении данных пользовательских бюджетов, то это и есть настройка политики аутентификации с помощью NSS. Число вариантов комбинаторно – два. Или сначала LDAP затем локальные базы, или наоборот, сначала локальные базы, а потом LDAP. Соответственно таким путем устанавливается преимущество одних бюджетов над другими. Это и есть политика.

2.2.Формирование политики аутентификации.

Условимся, что служба LDAP содержит глобальные для всей сети бюджеты. И если речь идет об аутентификации на рабочей станции, то политика, при которой отдается предпочтение глобальным бюджетам перед локальными, возможно оправдана. Вероятно, имеет смысл делать такое, чтобы каждый раз, авторизуясь как root на локальной машине, использовать единый пароль для root из LDAP, вне зависимости от настроек root из локальной базы.

Например. Ставим поиск в LDAP впереди локального.

```
server:~ # cat /etc/nsswitch.conf | grep "^\(passwd\|shadow\|group\)"
```

```
passwd: ldap files
shadow: ldap files
group:  ldap files
server:~ #
```

Подготавливаем файл с учетными данными root.

```
server:~ # cat >root.ldif<<EOT
> dn: uid=root,ou=People,dc=office,dc=localnet
> uid: root
> cn: root
> sn: root
> objectClass: top
> objectClass: inetOrgPerson
> objectClass: posixAccount
> objectClass: shadowAccount
> shadowLastChange: 12089
> shadowMax: 10000
> loginShell: /bin/bash
> uidNumber: 0
> gidNumber: 0
> homeDirectory: /root
> gecos: root
> EOT
server:~ #
```

Устанавливаем простенький пароль “lroot”.

```
server:~ # echo "userPassword: `slappasswd -h {crypt} -s lroot`" >>root.ldif
server:~ #
```

Вот, что должно получиться.

```
server:~ # cat root.ldif
dn: uid=root,ou=People,dc=office,dc=localnet
uid: root
cn: root
sn: root
objectClass: top
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
shadowLastChange: 12089
shadowMax: 10000
loginShell: /bin/bash
uidNumber: 0
gidNumber: 0
homeDirectory: /root
gecos: root
userPassword: {CRYPT}.vSlVrIfg2SZ2
server:~ #
```

Проверяем, что пока все нормально. Реальный парольный хеш root на скриншоте замаркирован.

```
server:~ # getent passwd | grep ^root
root:x:0:0:root:/root:/bin/bash
server:~ # getent shadow | grep ^root
root:xxxxxxxxxxxxx:12089:0:10000::::
server:~ #
```

Добавляем пользователя root в LDAP.

```
server:~ # ldapmodify -a -v -D "cn=ldapadmin,dc=office,dc=localnet" -H
ldap://localhost -x -w secret -f root.ldif
ldap_initialize( ldap://localhost )
add uid:
        root
add cn:
        root
add sn:
        root
add objectClass:
        top
        inetOrgPerson
        posixAccount
        shadowAccount
add shadowLastChange:
        12089
add shadowMax:
        10000
add loginShell:
        /bin/bash
add uidNumber:
        0
add gidNumber:
        0
add homeDirectory:
        /root
add gecos:
        root
add userPassword:
        {CRYPT}.vSlVrIfg2SZ2
adding new entry "uid=root,ou=People,dc=office,dc=localnet"
modify complete

server:~ #
```

Проверяем результат – видим два бюджета, и LDAP-бюджет впереди.

```
server:~ # getent passwd | grep ^root
root:x:0:0:root:/root:/bin/bash
root:x:0:0:root:/root:/bin/bash
server:~ # getent shadow | grep ^root
root:.vSlVrIfg2SZ2:12089::10000::::0
root:xxxxxxxxxxxxx:12089:0:10000::::
server:~ #
```

И все! Теперь можно авторизоваться на локальной станции как root по паролю из root-бюджета LDAP.

```
alekseybb@server:~> su - root
Password:
server:~ # id
uid=0(root) gid=0(root) groups=0(root),501(cvs)
server:~ #
```

При этом локальный пароль перестает приниматься. Если в этом была цель, то вот оно – получено. Фактически это взлом локального бюджета. При такой настройке безопасность рабочих станций всецело зависит от надежности сервера, содержащего базу LDAP, и не только от его сетевой защищенности, но также и от защиты файлов LDAP из /var/lib/ldap, которые допускают как чтение, так и модификацию в режиме offline.

Кстати сказать, размещение бюджетов в LDAP в MS Windows лишено подобного недостатка. Так как, во-первых, выбор базы аутентификации указывается явно, локальная машина или домен. И, во-вторых, названия бюджетов содержат префикс, имя локальной машины или домена, что позволяет различать их и в стадии пользовательской сессии. То есть перекрытие бюджетов есть местная юниксовая особенность. Но вернемся к теме.

Если же настройка LDAP ставит своей целью не подмену локальных бюджетов, а лишь их дополнение, то поиск по локальным базам следует указывать перед поиском в LDAP. Тогда, как это ожидается, локальные бюджеты будут превалировать перед сетевыми, что и будет являться той аутентификационной политикой, достижение которой поставлено целью. Итак, далее используется только настройка аналогичная следующей.

```
server:~ # cat /etc/nsswitch.conf | grep "\(^passwd\|^shadow\|^group\)"
passwd: files ldap
shadow: files ldap
group:  files ldap
server:~ #
```

Важно отметить, что в таком случае поиск в LDAP производится вне зависимости от успеха или неудачи локального. То есть информация в LDAP дополняет локальную. И какой именно информации отдать предпочтение остается на усмотрение клиентскому ПО, запросившему её. Если для получения справочной информации такой формат общения полезен, то в процессе аутентификации неоднозначности недопустимы. Далее именно процедуру решения, какой информации будет отдано предпочтение в каждом случае, попытаемся уточнить опытным путем. Для этого искусственно создадим конфликт бюджетов. И кроме этого проверке подвергнем такие ситуации, когда локальный бюджет конкурирует с бюджетом из LDAP частично.

Следует заметить, что изменения, вносимые в указанные файлы с настройками, ldap.conf и nsswitch.conf, должны отрабатываться немедленно. Но особо мнительные могут и перегрузиться для очистки среды и кэшей, связанных с постоянно задействованными библиотеками. Хотя обычно достаточно проследить, чтобы во время проведения настроек был отключен Кэш Сервиса Имен (Name Service Cache Daemon – NCSD). В SuSE остановка этого демона производится следующей командой.

```
server:~ # rcnscd stop
Shutting down Name Service Cache Daemon                                    done
server:~ #
```

Проверку работы системной аутентификации будем проводить в режиме отключенного кеширования nscd.

2.3.Создание тестовых бюджетов.

Чтобы определить, как будет отданы предпочтения в разных стадиях работы NSS с локальной базой и с LDAP, создадим два одинаковых бюджета и там и там. Естественно, использовать для таких экспериментов бюджет root недопустимо. Но в оценке важности полученных выводов следует не забывать о том, что они распространяются и на root тоже.

Сначала в локальной системной базе создадим тестового пользователя atest традиционным способом, и укажем для него парольную фразу “sates”.

```
server:~ # useradd -m -p `mkpasswd satest q1` atest
server:~ # cat /etc/passwd | grep atest
atest:x:1007:100::/home/atest:/bin/bash
server:~ #
```

Его uid, как видно выше, равен 1007. Проверим, что возможна аутентификация созданного пользователя именно с заданным паролем.

```
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
atest@server:~> exit
logout
alekseybb@server:~>
```

Теперь создадим аналогичного пользователя в LDAP, но в диалоге, назначающем пароль, укажем “latest”. Для создания используем утилиты из пакета Samba3. Настройка этих утилит подробно описана в руководствах по Samba3.

```
server:~ # /var/lib/samba/sbin/smbldap-useradd.pl -s /bin/bash atest
server:~ # /var/lib/samba/sbin/smbldap-passwd.pl atest
Changing password for atest
New password :
Retype new password :
server:~ #
```

Проверим, что два бюджета существуют одновременно. Сначала учетная информация, потом пароли.

```
server:~ # getent passwd | grep atest
atest:x:1007:100::/home/atest:/bin/bash
```

```
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ #
server:~ # getent shadow | grep atest
atest:q1MWt83DAs2IY:12615:1:99999:14:::
atest:x:::::::0
server:~ #
```

Как видно выше, второй бюджет имеет uid равный 1002, что позволит в дальнейшем отличать эти бюджеты не только по паролям.

Тут обнаруживается мелкая неувязочка, заключающаяся в том, что в режиме прозрачной настройки “только NSS с LDAP” возможно лишь применение парольных хешей, созданных по алгоритму {crypt}, а для Samba3 + LDAP стандартными являются хеши {ssha}. Поэтому парольные хеши в иной кодировке, чем {crypt}, не видны при чтении базы shadow, и в приведенном выше скриншоте вместо второго пароля выведен знак “x”. Не будем ничего править в настройках утилит из Samba3, а исправление одного единственного парольного хеша проведем в ручном режиме, как указано далее.

```
server:~ # cat >change.pass.ldif<<EOT
> dn: uid=atest,ou=People,dc=office,dc=localnet
> changetype: modify
> replace: userPassword
> EOT
server:~ # echo "userPassword: `slappasswd -h {crypt} -s latest`"
>>change.pass.ldif
server:~ # cat change.pass.ldif
dn: uid=atest,ou=People,dc=office,dc=localnet
changetype: modify
replace: userPassword
userPassword: {CRYPT}HGqB6dm4QjL5Y
server:~ #
server:~ # ldapmodify -v -D "cn=ldapadmin,dc=office,dc=localnet" -H
ldap://localhost -x -w secret -f change.pass.ldif
ldap_initialize( ldap://localhost )
replace userPassword:
        {CRYPT}HGqB6dm4QjL5Y
modifying entry "uid=atest,ou=People,dc=office,dc=localnet"
modify complete

server:~ #
```

Снова проверим, как два пароля видятся системой.

```
server:~ # getent shadow | grep atest
atest:q1MWt83DAs2IY:12615:1:99999:14:::
atest:HGqB6dm4QjL5Y:::::::0
server:~ #
```

Теперь вся учетная информация о синонимичных бюджетах доступна через NSS и их парольные хеши тоже. То есть все готово для проверки аутентификации.

2.4.Проверка NSS LDAP.

Далее будем по очереди пробовать переключение с текущего пользователя, в данном случае alekseybb, на atest с помощью утилиты su, с использованием парольной фразы «satest» и затем «latest». Взаимодействие su с PAM описывается следующим образом.

```
server:~ # cat /etc/pam.d/su | grep -v ^# | grep -v ^$
auth     sufficient     pam_rootok.so
auth     required       pam_unix2.so    nullok #set_secrpc
account  required       pam_unix2.so
password required       pam_pwcheck.so  nullok
password required       pam_unix2.so    nullok use_first_pass use_authtok
session  required       pam_unix2.so    debug # none or trace
server:~ #
```

Это стандартная дистрибутивная настройка PAM для утилиты su в SuSE Linux. Здесь главное то, что подсистема PAM не настроена на использование LDAP и всю информацию о доступных бюджетах и их аутентификационных данных PAM получает через NSS.

2.4.1. Проверка nss 1.

Вся информация об альтернативных бюджетах присутствует в полной мере. Каждый описан в соответствующей базе. Проверяем переключение пользователей.

```
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
atest@server:~> exit
logout
alekseybb@server:~> su - atest
Password:
su: неправильный пароль
alekseybb@server:~>
```

Существование локального бюджета в данном случае полностью перекрывает бюджет в LDAP. Аутентификация бюджета из LDAP невозможна. Тем самым не представляется возможным путем создания синонимов для локальных пользователей в LDAP взломать локальный компьютер, воспользовавшись тем, что бюджеты в LDAP создаются с известными паролями.

2.4.2. Проверка nss 2.

Удаляем запись о локальном бюджете из passwd.

```
server:~ # cp /etc/passwd /etc/passwd.atest
server:~ # cat /etc/passwd.atest | grep -v ^atest >/etc/passwd
server:~ #
server:~ # getent passwd | grep atest
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:q1MWt83DAs2IY:12615:1:99999:14:::
atest:HGqB6dm4QjL5Y:::::::0
server:~ #
```

И снова проверяем.

```
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
atest@server:~>
atest@server:~> exit
logout
alekseybb@server:~> su - atest
Password:
su: неправильный пароль
alekseybb@server:~>
```

Утилита приняла локальный пароль, но учетную информацию новый сеанс взял из базы LDAP.

Здесь в полной мере проявляется то, что политика предпочтения всецело находится в ведении клиентской части. И, хотя смоделированная ситуация маловероятна с практической точки зрения, но это дает основания для внесения правок в тексты NSS.

2.4.3. Проверка nss 3.
Удалим запись о парольном хеше из shadow.

```
server:~ # cp /etc/shadow /etc/shadow.atest
server:~ # cat /etc/shadow.atest | grep -v ^atest >/etc/shadow
server:~ # getent passwd | grep atest
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:HGqB6dm4QjL5Y:::::::0
server:~ #
```

Теперь фактически локальный пользователь отсутствует для системы аутентификации.

```
alekseybb@server:~> su - atest
Password:
su: неправильный пароль
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
atest@server:~> exit
logout
alekseybb@server:~>
```

Результат следует логике. Авторизация полностью была проведена с учетом базы LDAP.

2.4.4. Проверка nss 4.

Восстановим запись о бюджете в passwd.

```
server:~ # cat /etc/passwd.atest >/etc/passwd
server:~ # getent passwd | grep atest
atest:x:1007:100::/home/atest:/bin/bash
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:HGqB6dm4QjL5Y:::::::0
server:~ #
```

Случай маловероятный, но тоже проверим.

```
alekseybb@server:~> su - atest
Password:
su: неправильный пароль
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
atest@server:~> exit
logout
alekseybb@server:~>
```

Теперь был принят пароль LDAP, но учетная информация взята из системной базы. В сравнении с проверкой nss 3, не знаю, что из этого хуже. Но выводы в отношении неразборчивости алгоритмов NSS аналогичны.

2.4.5. Проверка nss 5.
Вернем всю информацию снова в локальную базу, но заблокируем локальный логин.

```
server:~ # echo "atest:*:::::::0" >>/etc/shadow
server:~ # getent passwd | grep atest
atest:x:1007:100::/home/atest:/bin/bash
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:*:::::::0
atest:HGqB6dm4QjL5Y:::::::0
server:~ #
```

И проверим.

```
alekseybb@server:~> su - atest
Password:
su: неправильный пароль
alekseybb@server:~> su - atest
Password:
su: неправильный пароль
alekseybb@server:~>
```

Как видно по результату, авторизация пользователя atest заблокирована, как в локальной базе, так и через LDAP.

Полученные результаты сведем в единую таблицу.

	NSS 1	NSS 2	NSS 3	NSS 4	NSS 5
passwd	есть			есть	есть
shadow	есть	есть			блокирован
pass	shadow	shadow	LDAP	LDAP	нет
id	passwd	LDAP	LDAP	passwd	нет
1 из 5	OK	-LDAP	OK	???	OK

Теперь можно сделать предварительный вывод. Политика, реализованная в NSS LDAP, полностью отвечает требованиям безопасности локальных бюджетов в первом приближении. Никакое создание альтернативных пользователей в LDAP не позволит компрометировать систему локальной аутентификационной базы. Однако, путем создания подставного, локального пароля можно произвести аутентификацию с его помощью бюджета из LDAP. Оценка “1 из 5”.

2.5.Повышение безопасности NSS LDAP.

Вернемся к опциям сервера LDAP. До сих пор все запросы от клиента к LDAP проводились от привилегированного пользователя rootdn. Для этого пришлось указать пароль этого пользователя в /etc/ldap.conf в открытом виде. Попробуем изменить эту ситуацию, так как rootdn имеет абсолютные права в отношении LDAP, и указание его пароля хоть где-нибудь в открытом виде явно снижает безопасность системы аутентификации на основе LDAP.

Для проверки будем использовать su в режиме, когда локальный бюджет удален из системной базы.

```
server:~ # cat /etc/shadow.atest | grep -v ^atest >/etc/shadow
server:~ # cat /etc/passwd.atest | grep -v ^atest >/etc/passwd
server:~ # getent passwd | grep atest
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:HGqB6dm4QjL5Y:::::::0
server:~ #
```

Исходный результат.

```
alekseybb@server:~> su - atest
Password:
su: неправильный пароль
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
atest@server:~> exit
logout
alekseybb@server:~>
```

Далее проверяем только нелокальный логин с паролем latest, так как будет менятся лишь настройка клиента LDAP и права служебного пользователя в базе LDAP.

Добавим специальный служебный LDAP бюджет. Ранее применявшийся rootdn не требовал его фактического занесения в базу LDAP, так как это был предопределенный системный бюджет, а новый бюджет ldapbrowser надо будет явно занести в базу вместе с его учетной информацией. Парольной фразой выберем “browser”. Поскольку авторизоваться с этим бюджетом будет клиент LDAP, то ограничение на использования только хешей {crypt} здесь не имеет силы.

```
server:~ # cat >ldapbrowser.ldiff<<EOT
> dn: cn=ldapbrowser,dc=office,dc=localnet
> cn: ldapbrowser
> sn: ldapbrowser
> objectClass: person
> objectClass: top
> EOT
server:~ # echo "userPassword:: `slappasswd -h {ssha} -s browser`"
>>ldapbrowser.ldiff
server:~ # cat ldapbrowser.ldiff
dn: cn=ldapbrowser,dc=office,dc=localnet
cn: ldapbrowser
sn: ldapbrowser
objectClass: person
objectClass: top
userPassword:: {SSHA}stfV4nrgwlBg3K4dReyUFo8KSAHqOjNo
server:~ # ldapmodify -v -a -D "cn=ldapadmin,dc=office,dc=localnet" -H
ldap://localhost -x -w secret -f ldapbrowser.ldiff
ldap_initialize( ldap://localhost )
add cn:
        ldapbrowser
add sn:
        ldapbrowser
add objectClass:
        person
        top
add userPassword:
        {SSHA}stfV4nrgwlBg3K4dReyUFo8KSAHqOjNo
adding new entry "cn=ldapbrowser,dc=office,dc=localnet"
modify complete

server:~ #
```

Для начала определим полномочия ldapbrowser как “все читать”. Права на запись оставим только владельцам записей. Для этого в файл с настройками LDAP допишем указанный ниже фрагмент и перезапустим сервер LDAP.

```
server:~ # cat /etc/openldap/slapd.conf | grep -v ^# | grep -v ^$ | sed -n
"/^access/,$ p"
access  to dn=".*,dc=office,dc=localnet"
        by self                                 write
        by *                                    read
server:~ # rcldap restart
Shutting down ldap-server                                           done
Starting ldap-server                                                done
```

```
server:~ #
```

И поправим настройки клиента NSS LDAP так, чтобы для доступа к LDAP использовался этот специальный пользователь ldapbrowser.

```
server:~ # cat /etc/ldap.conf | grep -v ^# | grep -v ^$
host    127.0.0.1:389
base    dc=office,dc=localnet
nss_base_passwd         ou=People,dc=office,dc=localnet?one
nss_base_shadow         ou=People,dc=office,dc=localnet?one
nss_base_group          ou=Groups,dc=office,dc=localnet?one
binddn cn=ldapbrowser,dc=office,dc=localnet
bindpw browser
server:~ #
```

Тут же проверим видимость бюджета LDAP в системе.

```
server:~ # getent passwd | grep atest
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:HGqB6dm4QjL5Y:::::::0
server:~ #
```

И далее, как и ожидалось, успешный тестовый логин.

```
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
atest@server:~> exit
logout
alekseybb@server:~>
```

Теперь будем последовательно снижать права доступа пользователя ldapbrowser от read до тех пор, пока аутентификация atest не прекратится. В LDAP принята следующая эскалация прав: none – ничего нельзя, auth – возможна аутентификация по значению этого поля, compare – возможно выполнения операций сравнения (равно, нет, больше, меньше), search – возможен поиск (есть ли это значение в базе), read – можно читать значение, write – можно изменять.

После замены на search получаем.

```
alekseybb@server:~> su - atest
su: пользователь atest не существует
alekseybb@server:~>
```

Вполне логичный результат. Тогда конкретизируем доступ по полям. В локальной базе парольные хеши отделены от описаний бюджетов и соответственно недоступны для чтения никому за пределами группы shadow кроме root.

```
server:~ # ls -als /etc/passwd
   4 -rw-r--r--    1 root     root         2184 Jul 16 20:05 /etc/passwd
server:~ # ls -als /etc/shadow
```

```
   4 -rw-r-----    1 root     shadow       1107 Jul 16 20:05 /etc/shadow
server:~ #
```

Примем такую политику разделения доступа за эталон, и запретим всем кроме ldapbrowser доступ на чтение к атрибуту userPassword, содержащему значение хеша. Для этого снова поменяем настройка сервера LDAP.

```
server:~ # cat /etc/openldap/slapd.conf | grep -v ^# | grep -v ^$ | sed -n
"/^access/,$ p"
access  to dn=".*,dc=office,dc=localnet" attr=userPassword
        by anonymous                                auth
        by self                                     write
        by dn="cn=ldapbrowser,dc=office,dc=localnet" read
        by *                                        none
access  to dn=".*,dc=office,dc=localnet"
        by self                                     write
        by *                                        read
server:~ #
```

Первое правило устанавливает ограничения для атрибута userPassword, а второе на все остальное. Кроме DN ldapbrowser фигурирует определение self, то есть владелец, и “*”, что обозначает всех остальных. Первый может изменять пароли, а остальные могут читать все кроме парольных хешей. Важно указать доступ на аутентификацию для anonymous, так как прежде, чем владелец может быть аутентифицирован как таковой, он является анонимом, что в настройках доступа обозначается как anonymous. Осталось перегрузить LDAP и проверить, что информация из базы LDAP доступна.

```
server:~ # rcldap restart
Shutting down ldap-server                                              done
Starting ldap-server                                                   done
server:~ # getent passwd | grep atest
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:HGqB6dm4QjL5Y:::::::0
server:~ #
```

Проверим, что авторизация нелокального пользователя при таком изменении происходит также как и ранее.

```
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
atest@server:~> exit
logout
alekseybb@server:~>
```

Сделаем выводы. Использование NSS в процедуре аутентификации производится прозрачным образом. То есть, NSS LDAP предоставляет лишь транспорт для доступа к полям базы LDAP. И поэтому процедуры аутентификации требуют от NSS LDAP лишь значения полей, содержащих парольные хеши. Таким образом, бюджет LDAP, от имени которого клиент NSS LDAP производит доступ к базе LDAP, должен

иметь права на чтение парольных хешей, так как он их читает и передает для обработки в вышестоящую процедуру.

Но, тем не менее, произведенная выше настройка позволила избавиться от указания пароля привилегированного пользователя rootdn в настройках клиента. Таким образом, взлом локального компьютера не угрожает взломом также и всех бюджетов LDAP базы локальной сети.

В /etc/ldap.conf пароль пользователя ldapbrowser до сих пор записан в открытом виде и в общедоступном месте. Есть способ немного изменить и это.

```
server:~ # cat /etc/ldap.conf | grep -v ^# | grep -v ^$
host    127.0.0.1:389
base    dc=office,dc=localnet
nss_base_passwd         ou=People,dc=office,dc=localnet?one
nss_base_shadow         ou=People,dc=office,dc=localnet?one
nss_base_group          ou=Groups,dc=office,dc=localnet?one
rootbinddn cn=ldapbrowser,dc=office,dc=localnet
server:~ # echo browser >/etc/ldap.secret
server:~ # chmod 600 /etc/ldap.secret
server:~ # ls -als /etc/ldap.secret
   4 -rw-------    1 root     root            8 Jul 16 21:52 /etc/ldap.secret
server:~ #
```

Это максимум защищенности аутентификационной системы с использованием NSS LDAP.

Последняя нерассмотренная функция управления бюджетами это passwd, которая позволяет администратору и пользователям менять пароли и управлять другими данными в бюджетных базах. В настройке аутентификации через NSS LDAP управление локальными базами производится через утилиты семейства passwd, а управление записями в LDAP через утилиту ldapmodify, smbldap-passwd.pl и проч. И эти группы утилит никак между собой не связаны. Попытки как-то синхронизировать эти процедуры можно делать лишь, используя PAM, о чем речь пойдет в следующей главе.

Как инструмент формирования политики NSS весьма прост. Но в такой простоте содержится вырождение. Во-первых, как уже упоминалось, возможно лишь последовательное подключение новых механизмов поиска системной информации и предпочтение данных остается всецело на приложении. Во-вторых, NSS “не может” в силу дизайна соблюсти тайну парольных хешей, а значит, использование его в аутентификации ослабляет защищенность этой процедуры. Но, тем не менее, в-третьих, настройка NSS LDAP является необходимой для работы очень многих необходимых утилит и подсистем вместе с LDAP, то есть надо каждый раз определять, какой уровень доступа к данным LDAP для NSS является минимально необходимым.

Глава 3. О PAM.

PAM или Подгружаемые Модули Аутентификации (Pluggable Autentication Modules) представляют собой вторую систему, определяющую пути и режимы аутентификации пользователей в Linux, доступ к данным пользовательских бюджетов и управление ими. С точки зрения именно аутентификации это основная система. В описанной выше настройке NSS LDAP, предполагалось использование PAM в обычной режиме, когда информация из LDAP поставлялась прозрачным образом через библиотеки NSS. Но PAM может быть настроен на использование LDAP самостоятельно и независимо от настроек NSS. Проделаем это, начиная с самого простого.

3.1.Настройка клиента PAM LDAP.

В системе PAM представлен, как набор объектных библиотек. И соответственно, за взаимодействие их с LDAP отвечает библиотека, а точнее динамический модуль pam_ldap из одноименного пакета.

```
server:~ # ls -l `rpm -ql pam_ldap | grep lib`
-rwxr-xr-x  1 root  root    41273 Sep 24  2003 /lib/security/pam_ldap.so
server:~ #
```

Для начала надо настроить клиент LDAP, который будет далее использован PAM. В PAM LDAP настройки клиента размещаются в том же файле, что и для NSS.

```
server:~ # strings `rpm -ql pam_ldap | grep lib` | grep ldap.conf
/etc/ldap.conf
server:~ #
```

Ничего там менять не будем. Воспользуемся настройками, что были сделаны ранее.

```
server:~ # cat /etc/ldap.conf | grep -v ^# | grep -v ^$
host    127.0.0.1:389
base    dc=office,dc=localnet
nss_base_passwd         ou=People,dc=office,dc=localnet?one
nss_base_shadow         ou=People,dc=office,dc=localnet?one
nss_base_group          ou=Groups,dc=office,dc=localnet?one
rootbinddn cn=ldapbrowser,dc=office,dc=localnet
server:~ # cat /etc/ldap.secret
browser
server:~ # ls -als /etc/ldap.secret
   4 -rw-------    1 root     root            8 Jul 16 21:52 /etc/ldap.secret
server:~ #
```

Но в настройках NSS отключим поиск по базе LDAP, чтобы случайно не допустить “утечки” информации через NSS.

```
server:~ # cat /etc/nsswitch.conf >/etc/nsswitch.conf.ldap
server:~ # cat /etc/nsswitch.conf.ldap | grep -v "^\(passwd\|shadow\|group\)"
>/etc/nsswitch.conf
server:~ # cat >>/etc/nsswitch.conf<<EOT
```

```
passwd: files
shadow: files
group:  files
EOT
server:~ #
server:~ # cat /etc/nsswitch.conf | grep "^\(passwd\|shadow\|group\)"
passwd: files
shadow: files
group:  files
server:~ # cat /etc/nsswitch.conf >/etc/nsswitch.conf.noldap
server:~ #
```

И сразу же проверим, что это привело к нужному результату.

```
server:~ # getent passwd | grep atest
atest:x:1007:100::/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:v3uly2DUUdlNo:12615:1:99999:14:::
server:~ #
```

Теперь NSS не обращается к LDAP, соответственно не видит бюджетов LDAP, не передает информацию о них в систему, и для проведения независимых настроек PAM LDAP нет более препятствий.

Взаимодействие PAM с LDAP осуществляется с помощью соответствующего модуля, названного, как очевидно, pam_ldap. Режимы взаимодействия всевозможных служб и утилит Linux с PAM определяются в SuSE, как и во многих других дистрибуциях, в специальных файлах-сценариях, размещенных в /etc/pam.d, каждый из которых назван по имени соответствующей программы.

Немного о формате файла-сценария. Модули PAM вызываются для выполнения 4-х сервисных задач (в терминологии PAM - facility): auth - аутентификации , account - получения привилегий доступа, password - управления паролями и session - сопровождение сессий. В каждой задаче может быть перечислено несколько модулей, которые будут вызываться последовательно, образуя стек PAM для данной задачи. Каждый вызываемый модуль возвращает в стек результат своей работы, или успешный (PAM_SUCCESS), или негативный (PAM_AUTH_ERR), или игнорирующий (PAM_IGNORE) или иной. Для каждого вызова может быть указан набор управляющих флагов в виде соответствия кода возврата и того, как результат работы модуля скажется на обработке всей сервисной задачи, например ignore, ok, die, что не требует расшифровки, так как интуитивно ясно. Принято не расписывать все соответствия кодов возврата и типовых реакций, а применять макросы, соединяющие определенные наборы таких флагов. Наиболее употребимые из них:

requisite – немедленное прекращение дальнейшего выполнения сервисной задачи с общим негативным результатом в случае негативного результата выполнения данного модуля;

required – требование удачного выполнения этого модуля одновременно с выполнением всех остальных, перечисленных в данной сервисной задаче;

sufficient – немедленное прекращение дальнейшего выполнения сервисной задачи с общим позитивным результатом, в случае позитивного результата выполнения данного модуля и всех, предыдущих с флагом required в стеке задачи, если же модуль вернул негативный результат, то его значение игнорируется;

optional – выполнение данного модуля никак не сказывается на результате всей задачи, но играет дополнительную информационную роль.

Итак, исходная настройка PAM для su.

```
server:~ # cat /etc/nsswitch.conf.noldap >/etc/nsswitch.conf
server:~ # cat /etc/shadow.atest | grep -v ^atest >/etc/shadow
server:~ # cat /etc/passwd.atest | grep -v ^atest >/etc/passwd
server:~ # cat /etc/security/pam_unix2.conf | grep -v ^# | grep -v ^$
auth:   nullok debug
account:        debug
password:       nullok debug
session:        debug none
server:~ # cat /etc/pam.d/su | grep -v ^# | grep -v ^$
auth     sufficient     pam_rootok.so
auth     required       pam_unix2.so    nullok #set_secrpc
account  required       pam_unix2.so
password required       pam_pwcheck.so  nullok
password required       pam_unix2.so    nullok use_first_pass use_authtok
session  required       pam_unix2.so    debug # none or trace
server:~ # getent passwd | grep atest
server:~ # getent shadow | grep atest
server:~ #
```

Так как NSS LDAP отключен, то локальный бюджет atest не виден, и единственная возможность аутентификации с использованием atest из LDAP заключается в настройке PAM LDAP. В исходном состоянии это невозможно.

```
alekseybb@server:~> su -c "id" atest
su: пользователь atest не существует
alekseybb@server:~>
```

Добавим перед вызовом pam_unix2 обращение к pam_ldap с флагом sufficient, чтобы можно было произвести настройку и одновременно ничего не испортить в шатной работе. Например, так.

```
server:~ # cat /etc/pam.d/su | grep -v ^# | grep -v ^$
auth     sufficient     pam_rootok.so
auth     sufficient     pam_ldap.so     nullok
auth     required       pam_unix2.so    nullok #set_secrpc
account  sufficient     pam_ldap.so
account  required       pam_unix2.so
password required       pam_pwcheck.so  nullok
password sufficient     pam_ldap.so     nullok use_first_pass use_authtok
password required       pam_unix2.so    nullok use_first_pass use_authtok
account  sufficient     pam_ldap.so
session  required       pam_unix2.so    debug # none or trace
server:~ #
```

Попробуем авторизоваться как atest.

```
alekseybb@server:~> su -c "id" atest
su: пользователь atest не существует
alekseybb@server:~>
```

Таким образом, без настройки NSS пользовательский бюджет из LDAP не виден большинству служб. Хотя не исключено, что есть программы и сервисы, обходящиеся без просмотра базы passwd, но su к таким не относится. И чтобы продвинуться хоть как-то в решении наших вопросов обманем эту утилиту, исправив пути поиска информации в nsswitch.conf для passwd следующим образом.

```
server:~ # cat /etc/nsswitch.conf | grep ^passwd
passwd: files ldap
server:~ #
```

Теперь авторизация проходит успешно.

```
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513 группы=513,14(uucp),16(dialout),17(audio),33(video)
alekseybb@server:~>
```

И поскольку база shadow, содержащая парольные хеши, через NSS не видна, можно быть уверенными, что аутентификация была произведена с использованием именно pam_ldap.

Настало время определить, что дает использование PAM LDAP, кроме того, что лишает просмотра базы passwd без привлечения NSS LDAP.

3.2.Повышение безопасности PAM LDAP.

В исходных настройках ограничений доступа по полям LDAP было установлено, что некий пользователь-посредник ldapbrowser должен иметь возможность читать парольные хеши, так как NSS доставляет их в вызывающую программу прозрачным образом. Но теперь NSS для аутентификации не используется. Тогда запретим доступ к парольным хешам всем кроме владельцев. Полностью установки доступа в LDAP будут выглядеть так.

```
server:~ # cat /etc/openldap/slapd.conf | grep -v ^# | grep -v ^$ | sed -n
"/^access/,$ p"
access  to dn=".*,dc=office,dc=localnet" attr=userPassword
        by anonymous                            auth
        by self                                 write
        by *                                    none
access  to dn=".*,dc=office,dc=localnet"
        by self                                 write
        by *                                    read
server:~ #
```

Теперь пользователь-посредник более не упоминается. Проверим.

```
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513 группы=513,14(uucp),16(dialout),17(audio),33(video)
alekseybb@server:~>
```

Все работает. Так как использование хешей {crypt} было условием использования

NSS, то сейчас и это можно исправить. Поменяем парольный хеш снова на {ssha} так, как это было сделано вначале с использованием smbldap-passwd.pl и снова проверим.

```
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513 группы=513,14(uucp),16(dialout),17(audio),33(video)
alekseybb@server:~>
```

Все работает. Тем самым достигнут максимум ограничения доступа для клиента PAM LDAP: пользователь посредник не имеет доступа к парольным хешам и мощность хеширующего алгоритма полностью определяется приложением. Например, для Samba3 стандартными являются хеши {ssha}.

3.3.Настройка pam_unix2.

Штатный способ подключения LDAP в PAM, который рекомендован в SuSE, заключается в использовании параметра use_ldap в аргументах модуля pam_unix2. В этом случае модуль pam_ldap будет вызван автоматически так, будто в файле-скрипте его вызов предшествует обращению к pam_unix2 с макросом sufficient.

Файл, который задает режимы работы pam_unix2, размещен в /etc/security.

```
server:~ # cat /etc/security/pam_unix2.conf | grep -v ^# | grep -v ^$
auth:   nullok
account:
password:       nullok
session:        none
server:~ #
```

В рекомендациях SuSE к pam_ldap указано для включения использования LDAP добавить в настройки pam_unix2 параметр use_ldap для задач auth, account, password. Это можно сделать вручную или эти изменения будут произведены автоматически через YaST2 при настройке аутентификации как нелокальной. Поступим, как указано. Употребление параметра use_ldap не для всех PAM-задач, вероятно вызвано тем, что pam_unix2, по версии SuSE, в задаче session не используется. Но если воспользоваться опцией debug, то по отметкам в системных логах можно проследить какие задачи PAM на самом деле используются в su. Это auth и session. Password будет задействована, если бюджет, в который происходит аутентификация, потребует изменения пароля, например из-за его устаревания.

```
server:~ # cat /etc/security/pam_unix2.conf | grep -v ^# | grep -v ^$
auth:   nullok use_ldap
account:        use_ldap
password:       nullok use_ldap
session:        none
server:~ #
```

Вернем /etc/pam.d/su в исходное состояние.

```
server:~ # cat /etc/pam.d/su | grep -v ^# | grep -v ^$
```

```
auth     sufficient     pam_rootok.so
auth     required       pam_unix2.so    nullok #set_secrpc
account  required       pam_unix2.so
password required       pam_pwcheck.so  nullok
password required       pam_unix2.so    nullok use_first_pass use_authtok
session  required       pam_unix2.so    debug # none or trace
server:~ #
```

И проверим аутентификацию с использованием настроек pam_unix2.

```
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513 группы=513,14(uucp),16(dialout),17(audio),33(video)
alekseybb@server:~>
```

Теперь все готово для проверки полученной политики аутентификации и того, как она обрабатывает бюджеты-двойники.

3.4.Проверка pam_unix2.

Далее проверку проведем аналогично тому, как это было сделано для NSS, но немного сократим, как консольные команды, так и комментарии к ним.

3.4.1. Проверка unix2 1.

Оба бюджета присутствуют в полной мере.

```
server:~ # cat /etc/passwd.atest >/etc/passwd
server:~ # cat /etc/shadow.atest >/etc/shadow
server:~ # getent passwd | grep atest
atest:x:1007:100::/home/atest:/bin/bash
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:q1MWt83DAs2IY:12615:1:99999:14:::
server:~ #
```

Пробуем.

```
alekseybb@server:~> su -c "id" atest
Password:
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
alekseybb@server:~>
alekseybb@server:~> su -c "id" atest
Password:
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
alekseybb@server:~>
```

О, чудо! С помощью синонима из LDAP был взломан локальный пользователь.

3.4.2. Проверка unix2 2.

Удалим бюджет из passwd.

```
server:~ # cat /etc/passwd.atest | grep -v atest >/etc/passwd
server:~ # getent passwd | grep atest
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:q1MWt83DAs2IY:12615:1:99999:14:::
server:~ #
```

Проверим.

```
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
alekseybb@server:~>
```

Теперь происходит взлом бюджета LDAP через локальный пароль. Не уверен, что такое имеет ценность, но в среде с перемещаемыми профилями это позволит одному пользователю прочесть данные другого из закешированного профиля из-за ошибки администратора.

3.4.3. Проверка unix2 3.

Произведем удаление пароля из shadow.

```
server:~ # cat /etc/shadow.atest | grep -v ^atest >/etc/shadow
server:~ # getent passwd | grep atest
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
server:~ #
```

Проверим аутентификацию.

```
alekseybb@server:~> su -c "id" atest
Password:
su: неправильный пароль
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
alekseybb@server:~>
```

Все логично, так как локальный бюджет фактически отсутствует, то работает лишь бюджет из LDAP.

3.4.4. Проверка unix2 4.

Снова восстановим запись о бюджете в passwd.

```
server:~ # cat /etc/passwd.atest >/etc/passwd
```

```
server:~ # getent passwd | grep atest
atest:x:1007:100::/home/atest:/bin/bash
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
server:~ #
```

Вот результат проверки.

```
alekseybb@server:~> su -c "id" atest
Password:
su: неправильный пароль
alekseybb@server:~> su -c "id" atest
Password:
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
alekseybb@server:~>
```

Был принят единственный пароль, хеш которого содержится в базах, но бюджет выбран локальный.

3.4.5. Проверка unix2 5.

Вернем всю информацию локального бюджета на место, но заблокируем его логин.

```
server:~ # cat /etc/shadow.atest | grep -v ^atest >/etc/shadow
server:~ # echo "atest:*:::::::" >>/etc/shadow
server:~ # getent passwd | grep atest
atest:x:1007:100::/home/atest:/bin/bash
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep atest
atest:*:::::::
server:~ #
```

Проверяем.

```
alekseybb@server:~> su -c "id" atest
Password:
su: неправильный пароль
alekseybb@server:~> su -c "id" atest
Password:
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
alekseybb@server:~>
```

Несмотря на блокировку логин возможен. Опять локальный бюджет был взломан.

Подведем итоги. Представим полученные результаты в виде таблицы аналогично тому, как это было сделано для NSS LDAP.

	unix2 1	unix2 2	unix2 3	unix2 4	unix2 5
passwd	есть			есть	есть
shadow	есть	есть			блокирован
pass	любой	любой	LDAP	LDAP	LDAP
id	passwd	LDAP	LDAP	passwd	passwd
4 из 5	-passwd	-LDAP	OK	-passwd	-passwd

Выводы. Использование use_ldap с модулем pam_unix2, как это рекомендовано SuSE, и так, как это настраивает автоматически YaST2, недопустимо. Такая настройка создает условия, как для взлома локальных бюджетов через LDAP, так и обратно, для взлома LDAP бюджетов через локальные. Оценка “4 из 5”.

3.5.Настройка pam_ldap.

Более распространен способ подключения LDAP к PAM, заключающийся в использовании модуля pam_ldap. Если в дистрибуции используется файлы групповых политик, например system-auth, то добавление вызова модуля pam_ldap надо производить именно туда, иначе во все файлы-сценарии тех сервисов, которые предполагается использовать с LDAP. В SuSE нет файла групповой политики. Так как для проверки выбран su, то настройки будут проводиться с файлом-сценарием для утилиты su.

На первый взгляд очевидное включение pam_ldap в контексте макросов sufficient даже после просмотра локальных баз даст, скорее всего, такой же результат, как и при использовании pam_unix2 с use_ldap в предыдущем разделе. К сожалению, макрос для флагов sufficient не создает нужной политики. Но он максимально близок. Детальное соответствие флагов в sufficient следующее: success=done, new_authtok_reqd=done, default=ignore. В создаваемой же политике единственный случай, когда допустимо игнорирование лишь тот, когда пользователь не найден в локальной базе. Все остальные негативные ответы должны передаваться вызывающей программе. Поэтому создадим нужный файл-сценарий для su на основе следующей схемы.

```
<task>     sufficient pam_unix2.so
<task>     sufficient      pam_ldap.so
<task>     required   pam_deny.so
```

Или на основе упрощенного ее варианта.

```
<task>     sufficient pam_unix2.so
<task>     required   pam_ldap.so
```

То есть сначала поиск информации в локальной базе, потом в LDAP, и если нет ни там, ни там, то завершение с фатальным итогом. Но макрос sufficient для локального поиска немного модифицируем. Игнорировать укажем лишь отсутствие пользователя в локальной базе, а дефолтное значение заменим на неудачу, как в макросе required. Должно получиться так:

[success=done new_authtok_reqd=done user_unknown=ignore default=bad]

Но ведь ранее для использования su пришлось обмануть эту утилиту и теперь номинально пользователь должен находится хоть где-то даже, несмотря на то, что бюджет из LDAP будет недоступен для pam_unix2. Поэтому скорректируем набор флагов окончательным образом:

[success=done new_authtok_reqd=done user_unknown=ignore authinfo_unavail=ignore default=bad]

Теперь, указанные флаги должны заставить отрабатывать модуль pam_unix2 так, как нужно в создаваемой политике. Но не тут то было! Чудо дизайнерской мысли SuSE, модуль pam_unix2, во всех интересных случаях сообщает код ошибки PAM_AUTH_ERR, что не позволяет никак его далее интерпретировать и предположить источник ошибки. Поэтому придется вообще отказаться от использования этого модуля. Но и здесь не все просто. Вероятно именно с той целью, чтобы заставить даже несогласных пользователей обращаться только через pam_unix2, поскольку YaST настраивает политику безопасности лишь для pam_unix2, в SuSE традиционный альтернативный модуль pam_unix переименован, а его имя использовано как хардлинка на pam_unix2.

```
server:~ # ls -il /lib/security/pam_unix*.so
 112340 -rwxr-xr-x 2 root root 47468 Sep 23  2003 /lib/security/pam_unix.so
 112340 -rwxr-xr-x 2 root root 47468 Sep 23  2003 /lib/security/pam_unix2.so
 112329 -rwxr-xr-x 4 root root 51566 Sep 23  2003 /
lib/security/pam_unix_acct.so
 112329 -rwxr-xr-x 4 root root 51566 Sep 23  2003 /
lib/security/pam_unix_auth.so
 112329 -rwxr-xr-x 4 root root 51566 Sep 23  2003 /
lib/security/pam_unix_passwd.so
 112329 -rwxr-xr-x 4 root root 51566 Sep 23  2003 /
lib/security/pam_unix_session.so
server:~ #
```

Таким образом, надо будет использовать pam_unix_* в одноименных задачах PAM. Фактически достаточно подменить pam_unix2 только в задаче auth и там же указать нужный перечень флагов для создания правильной политики.

Окончательное решение будет таким.

```
server:~ # cat /etc/pam.d/su | grep -v ^# | grep -v ^$
auth     sufficient     pam_rootok.so
auth [success=done new_authtok_reqd=done user_unknown=ignore
authinfo_unavail=ignore default=bad] pam_unix_auth.so nullok
auth     required       pam_ldap.so     use_first_pass
account  sufficient     pam_unix2.so
account  required       pam_ldap.so
password required       pam_pwcheck.so  nullok
password sufficient     pam_unix2.so    nullok use_first_pass use_authtok
password required       pam_ldap.so     nullok use_first_pass use_authtok
session  sufficient     pam_unix2.so
session  required       pam_ldap.so
```

```
server:~ #
```

Опция use_first_pass позволяет модулю pam_ldap не запрашивать пароль для LDAP вторично, а воспользоваться тем, что уже был принят предыдущим модулем. Опция use_authtok действует аналогично, подавляя вывод предложения пользователю ввести новый пароль.

Используя такие опции, мы заставляем альтернативные модули pam_unix2 и pam_ldap работать с единожды введенным паролем. Тем самым процесс аутентификации происходит до первого совпадения и не позволяет иметь различные парольные фразы для разных баз. Вернемся к этому вопросу в разделе настройки процесса обновления паролей.

3.6.Проверка pam_ldap.

Проверку проведем по той же схеме, что и pam_unix2.

3.6.1. Проверка pam 1.

В случае, когда оба конкурентных бюджета представлены полностью, получаем следующий результат.

```
alekseybb@server:~> su -c "id" atest
Password:
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
alekseybb@server:~> su -c "id" atest
Password:
su: неправильный пароль
alekseybb@server:~>
```

То есть локальный бюджет перекрывает тот, что в LDAP.

3.5.2. Проверка pam 2.

Запись о бюджете в passwd уничтожена.

```
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
alekseybb@server:~>
alekseybb@server:~> su -c "id" atest
Password:
su: неправильный пароль
alekseybb@server:~>
```

Принимается пароль из shadow, но переключение происходит на бюджет из LDAP.

3.6.3. Проверка pam 3.

Локальный бюджет уничтожен полностью.

```
alekseybb@server:~> su -c "id" atest
Password:
su: неправильный пароль
alekseybb@server:~> su -c "id" atest
Password:
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
alekseybb@server:~>
```

Аутентификация происходит с использованием данных из LDAP.

3.6.4. Проверка pam 4.

Ситуация, когда есть запись в passwd, но отсутствует соответствующая в shadow.

```
alekseybb@server:~> su -c "id" atest
Password:
su: неправильный пароль
alekseybb@server:~> su -c "id" atest
Password:
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
alekseybb@server:~>
```

Происходит аутентификация с использованием парольной фразы из бюджета LDAP, но в локальный бюджет.

3.6.5. Проверка pam 5.

Опять оба конкурентных бюджета представлены полностью, но в shadow пароль заблокирован. Опять вне зависимости от работы NSS LDAP получаем один и тот же результат – аутентификация невозможна.

Соберем все результаты в одну таблицу.

	pam 1	pam 2	pam 3	pam 4	pam 5
passwd	есть			есть	есть
shadow	есть	есть			блокирован
pass	shadow	shadow	LDAP	LDAP	нет
id	passwd	LDAP	LDAP	passwd	нет
1 из 5	OK	-LDAP	OK	???	OK

Созданная политика полностью аналогична той, что была ранее реализована для NSS LDAP. Оценка “1 из 5”.

3.7.Настройка passwd с pam_unix2.

Среди служб, работающих с PAM, есть и та, что отвечает за изменение парольных хешей в бюджетной базе. Обращение к ней возможно, как из пользовательской сессии, так и из привилегированной сессии root. Утилита называется passwd, и за ее взаимодействие с PAM отвечает одноименный файл-сценарий из /etc/pam.d. Его оригинальная настройка следующая.

```
server:~ # cat /etc/pam.d/passwd | grep -v ^# | grep -v ^$
auth required   pam_unix2.so    nullok
account required        pam_unix2.so
password required       pam_pwcheck.so  nullok
password required       pam_unix2.so    nullok use_first_pass use_authtok
session required        pam_unix2.so
server:~ #
```

Отключим проверку качества пароля в модуле pam_pwcheck, так как сейчас это нам не важно. Получится так.

```
server:~ # cat /etc/pam.d/passwd | grep -v ^# | grep -v ^$
auth required   pam_unix2.so    nullok
account required        pam_unix2.so
password required       pam_unix2.so    nullok
session required        pam_unix2.so
server:~ #
```

Проверим работу. Запомним исходное состояние хешей.

```
server:~ # getent passwd | grep ^atest
atest:x:1007:100::/home/atest:/bin/bash
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep ^atest
atest:q1MWt83DAs2IY:12615:1:99999:14:::
atest:x:::::::0
server:~ # ldapsearch -LLL -H ldap://localhost -D
"cn=ldapadmin,dc=office,dc=localnet" -x -w secret "(uid=atest)" userPassword
dn: uid=atest,ou=People,dc=office,dc=localnet
userPassword:: e3NzaGF9b2trRVEyUXR2OUpxUzRKODVUR2hwN05qMEZzUnh0Nlk=

server:~ #
```

Начнем с рекомендованного SuSE варианта настройки с использованием в pam_unix2 параметра use_ldap.

```
server:~ # cat /etc/security/pam_unix2.conf | grep -v ^# | grep -v ^$
auth:   nullok use_ldap
account:        use_ldap
password:       md5 nullok use_ldap
session:        none
server:~ #
```

Далее устроим проверку. Предполагается, что перед обращением к утилите passwd было произведено изменение бюджета на atest через su точно так, как было сделано ранее. Запускается утилита passwd, а далее начинаются чудеса – система запрашивает пароль LDAP отдельным приглашением.

```
atest@server:~> id
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video)
atest@server:~> passwd
Changing password for atest.
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information update failed: Unknown error

Password changed
atest@server:~>
```

После чего парольный хеш изменяется только в локальной базе.

```
server:~ # getent passwd | grep ^atest
atest:x:1007:100::/home/atest:/bin/bash
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep ^atest
atest:$1$FzzzkMxz$krspW4ep1IhGdqQg5C.zE0:12620:1:99999:14:::
atest:x:::::::0
server:~ #
```

Если в ответ на приглашение пароля LDAP дать неверную парольную фразу, то, повторив это трижды или сразу оборвав через Enter, получим в итоге обычное приглашение ввести старый пароль локальной базы, и далее все как обычно и без ошибки приведет к такому же изменению пароля в локальной базе.

```
atest@server:~> passwd
Changing password for atest.
Enter login(LDAP) password:
LDAP Password incorrect: try again
Enter login(LDAP) password:
LDAP Password incorrect: try again
Enter login(LDAP) password:
LDAP Password incorrect: try again
Old Password:
New password:
Re-enter new password:
Password changed
atest@server:~>
```

В итоге.

```
server:~ # getent shadow | grep ^atest
atest:$1$AdwzCEtz$7qZwqb7BTjqC5tnTAqB2x0:12620:1:99999:14:::
atest:x:::::::0
server:~ # ldapsearch -LLL -H ldap://localhost -D
"cn=ldapadmin,dc=office,dc=localnet" -x -w secret "(uid=atest)" userPassword
dn: uid=atest,ou=People,dc=office,dc=localnet
userPassword:: e3NzaGF9b2trRVEyUXR2OUpxUzRKODVUR2hwN05qMEZzUnh0Nlk=

server:~ #
```

Затейливо, не правда ли? Ни в том, ни в другом случае пароль в LDAP не

обновился.

Если поднять отладку и просмотреть логи, то можно увидеть следующее. Далее приведено с сокращениями.

```
passwd[31521]: pam_unix2: pam_sm_chauthtok() called
slapd[27700]: conn=65 fd=15 ACCEPT from IP=127.0.0.1:35834 (IP=127.0.0.1:389)
slapd[27720]: conn=65 op=0 BIND dn="cn=ldapbrowser,dc=office,dc=localnet"
method=128
slapd[27720]: conn=65 op=0 BIND dn="cn=ldapbrowser,dc=office,dc=localnet"
mech=simple ssf=0
slapd[27720]: conn=65 op=0 RESULT tag=97 err=0 text=
slapd[27720]: conn=65 op=1 SRCH base="ou=People,dc=office,dc=localnet" scope=1
filter="(uid=atest)"
slapd[27720]: conn=65 op=1 SEARCH RESULT tag=101 err=0 nentries=1 text=
slapd[28038]: conn=65 op=2 BIND anonymous mech=implicit ssf=0
slapd[28038]: conn=65 op=2 BIND dn="uid=atest,ou=People,dc=office,dc=localnet"
method=128
slapd[28038]: conn=65 op=2 BIND dn="uid=atest,ou=People,dc=office,dc=localnet"
mech=simple ssf=0
slapd[28038]: conn=65 op=2 RESULT tag=97 err=0 text=
slapd[28038]: conn=65 op=3 BIND anonymous mech=implicit ssf=0
slapd[28038]: conn=65 op=3 BIND dn="cn=ldapbrowser,dc=office,dc=localnet"
method=128
slapd[28038]: conn=65 op=3 BIND dn="cn=ldapbrowser,dc=office,dc=localnet"
mech=simple ssf=0
slapd[28038]: conn=65 op=3 RESULT tag=97 err=0 text=
passwd[31521]: pam_unix2: pam_ldap/pam_sm_chauthtok() returned 0
passwd[31521]: pam_unix2: pam_sm_chauthtok() called
slapd[27720]: conn=65 op=4 MOD dn="uid=atest,ou=People,dc=office,dc=localnet"
slapd[27720]: conn=65 op=4 MOD attr=userPassword
slapd[27720]: conn=65 op=4 RESULT tag=103 err=50 text=
passwd[31521]: pam_ldap: ldap_modify_s Insufficient access
passwd[31521]: pam_unix2: pam_ldap/pam_sm_chauthtok() returned 6
slapd[28038]: conn=65 op=5 UNBIND
slapd[28038]: conn=65 fd=15 closed
```

Причиной является нехватка прав, так как pam_unix2 не только биндится с правами пользователя-посредника, но и пытается от имени него произвести модификацию парольных хешей.

Для проверки этого временно изменим права доступа, как указано ниже.

```
server:~ # cat /etc/openldap/slapd.access.conf | grep -v ^# | grep -v ^$
access  to dn=".*,dc=office,dc=localnet" attr=userPassword
        by anonymous                                auth
        by self                                     write
        by dn="cn=ldapbrowser,dc=office,dc=localnet" write
        by *                                        none
access  to dn=".*,dc=office,dc=localnet"
        by self                                     write
        by *                                        read
server:~ #
```

И снова повторим эксперимент.

```
atest@server:~> passwd
Changing password for atest.
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information changed for atest
atest@server:~>
```

Смотрим, что получилось в ответ на ввод правильного пароля из LDAP.

```
server:~ # getent shadow | grep ^atest
atest:q1MWt83DAs2IY:12615:1:99999:14:::
atest:x:::::::0
server:~ # ldapsearch -LLL -H ldap://localhost -D
"cn=ldapadmin,dc=office,dc=localnet" -x -w secret "(uid=atest)" userPassword
dn: uid=atest,ou=People,dc=office,dc=localnet
userPassword:: bGF0ZXN0Mg==

server:~ #
```

Таким образом, был изменен пароль в LDAP, но в локальной базе остался прежний пароль. Если отказаться от ввода пароля из LDAP, будет изменен локальный пароль. Все вполне устраивает, то есть поддается управлению, кроме того, что пришлось разрешить пользователю посреднику изменять все пароли в LDAP. Притом, что пароль самого посредника хранится в чистом виде на локальной станции.

Далее проверка pam_unix2 будет идти именно с такой настройкой контроля доступа в LDAP.

Здесь выясняется, что пароль записался в режиме {clear}.

```
server:~ # ldapsearch -LLL -H ldap://localhost -D
"cn=ldapadmin,dc=office,dc=localnet" -x -w secret "(uid=atest)" userPassword
dn: uid=atest,ou=People,dc=office,dc=localnet
userPassword:: bGF0ZXN0Mg==

server:~ # echo bGF0ZXN0Mg== | mimencode -u ; echo
latest2
server:~ #
```

Для исправления этого поправим настройки клиента LDAP. Добавим указание использовать в PAM внешнюю утилиту хеширования. Альтернативные варианты clear и md5 отбросим, как недостаточно сильные. Окончательно так.

```
server:~ # cat /etc/ldap.conf | grep -v ^# | grep -v ^$
host    127.0.0.1:389
base    dc=office,dc=localnet
pam_password    exop
nss_base_passwd         ou=People,dc=office,dc=localnet?one
nss_base_shadow         ou=People,dc=office,dc=localnet?one
nss_base_group          ou=Groups,dc=office,dc=localnet?one
rootbinddn cn=ldapbrowser,dc=office,dc=localnet
```

```
server:~ #
```

Здесь важно понять, что процедура аутентификации с PAM происходила внутри LDAP в процессе операции BIND как анонима, поэтому PAM прекрасно “понимал” парольные хеши типа {ssha} и “ему” для этого не надо было их непосредственно читать. Но вот, как только встал вопрос о записи собственных хешей внутрь LDAP, то сразу же возникла необходимость доступа по записи для PAM и мощность парольных хешей ограничилась возможностями “самого” PAM. То есть, требование указывать один и тот же тип хеширования, как в настройках сервера LDAP, так и в настройках клиента не имеет под собой никаких оснований. Так как распространяется не на задачу auth, а лишь на задачу password.

Итак, меняем пароль и смотрим, что получилось.

```
server:~ # ldapsearch -LLL -H ldap://localhost -D
"cn=ldapadmin,dc=office,dc=localnet" -x -w secret "(uid=atest)" userPassword
dn: uid=atest,ou=People,dc=office,dc=localnet
userPassword:: e1NNRDV9bS8yQytsNWJueWxzSkF1UFh6azh6aE44eDF3PQ==

server:~ #
server:~ # echo e1NNRDV9bS8yQytsNWJueWxzSkF1UFh6azh6aE44eDF3PQ== | mimencode
-u ; echo
{SMD5}m/2C+l5bnylsJAuPXzk8zhN8x1w=
server:~ #
```

Был создан хеш типа {smd5}. Примем это за максимум того, что можно “выжать” из pam на сей день.

3.8.Проверка passwd с pam_unix2.

Проверку проведем по уже привычной схеме, по возможности избегая подробных объяснений там, где все и так очевидно.

3.8.1. Проверка passwd с pam_unix2 1.

Присутствуют оба бюджета. Такая проверка была проведена в предыдущем разделе 3.7. Здесь только резюмируем. На этапе ввода старого пароля можно выбрать, ту базу, в которой предполагается изменить парольный хеш. Работают оба варианта, как локальный, так и через LDAP.

3.8.2. Проверка passwd с pam_unix2 2.

Возвращаем все базы к исходному виду, но уничтожаем запись о бюджете в passwd.

```
server:~ # cat /etc/passwd.atest | grep -v ^atest >/etc/passwd
server:~ # getent passwd | grep ^atest
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep ^atest
atest:q1MWt83DAs2IY:12615:1:99999:14:::
```

```
atest:x:::::::0
server:~ # ldapsearch -LLL -H ldap://localhost -D
"cn=ldapadmin,dc=office,dc=localnet" -x -w secret "(uid=atest)" userPassword
dn: uid=atest,ou=People,dc=office,dc=localnet
userPassword:: e3NzaGF9b2trRVEyUXR2OUpxUzRKODVUR2hwN05qMEZzUnh0Nlk=

server:~ #
```

Если вводится правильный пароль LDAP, то получается следующее.

```
atest@server:~> passwd
Changing password for atest.
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information changed for atest
atest@server:~>
```

И парольный хеш в LDAP меняется, как ожидалось. Если оборвать диалог с LDAP и ввести пароль локального бюджета, то диагностика будет иной.

```
atest@server:~> passwd
Changing password for atest.
Enter login(LDAP) password:
Password change aborted
Old Password:
passwd: Authentication failure
atest@server:~>
```

Парольный хеш в локальной базе при этом не изменится.

3.8.3. Проверка passwd с pam_unix2 3.

Теперь полностью уничтожим локальный бюджет. Прекрасно и бесконфликтно происходит переключение пользователя, и точно также обновление пароля в базе LDAP.

```
alekseybb@server:~> su - atest
Password:
atest@server:~> id
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
atest@server:~> passwd
Changing password for atest.
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information changed for atest
atest@server:~> exit
logout
alekseybb@server:~>
```

Естественно, никакие манипуляции с локальным бюджетом в таком режиме невозможны.

3.8.4. Проверка passwd с pam_unix2 4 и 5.

Проверки в случае, когда записи о локальном бюджете присутствуют в passwd, но отсутствуют в shadow или там же заблокированы, происходят с тем же результатом, что и в предыдущем тесте.

Сведем все результаты в таблицу.

	unix2 1	unix2 2	unix2 3	unix2 4	unix2 5
passwd	есть			есть	есть
shadow	есть	есть			блокирован
pass	любой	любой	LDAP	LDAP	LDAP
id	passwd	LDAP	LDAP	passwd	passwd
Old pass	соответств.	LDAP	LDAP	LDAP	LDAP
изменение	соответств.	LDAP	LDAP	LDAP	LDAP
3 из 5 ?	OK	???	OK	???	???

Полученный результат скорее положительный. В тех случаях, когда бюджеты были представлены в работоспособном виде (1 и 3), манипулирование парольными хешами происходило правильным образом. Во всех остальных случаях поврежденный или заблокированный локальный бюджет игнорировался.

3.9.Настройка passwd с pam_ldap.

Отказ от использования pam_unix2 подразумевает переход на pam_unix_passwd. Но использование в SuSE традиционного модуля pam_unix сопряжено с трудностями. Во-первых, его надо будет модифицировать так, как указано в “Приложении B”, чтобы избавиться от сообщений “invalid pointer” и других проблем. И, во-вторых, запастись терпением, чтобы удержаться от модификации приветствия “Changing password for...”, которое теперь выдается дважды, сначала от passwd, потом от pam_unix_passwd. Не должно также смущать отсутствие подтверждения успеха изменения локального парольного хеша.

Типичный диалог выглядит так.

```
atest@server:~> passwd
Changing password for atest.
Changing password for atest
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
atest@server:~>
```

Здесь был в ответ на приглашение введен правильный пароль локального бюджета, что привело к удачной смене его в shadow. Пароль в LDAP не изменился. Если указать вместо локального пароля пароль из LDAP, то в диалог добавится

сообщение об удачном изменении парольного хеша в LDAP.

```
atest@server:~> passwd
Changing password for atest.
Changing password for atest
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
LDAP password information changed for atest
atest@server:~>
```

Тот факт, что при этом изменятся точно также и хеш из shadow, не придается огласке в терминальном выводе. Все это происходит при следующих настройках файла-сценария для passwd.

```
server:~ # cat /etc/pam.d/passwd | grep -v "^\(#\|$\)"
auth [success=done new_authtok_reqd=done user_unknown=ignore
authinfo_unavail=ignore default=bad] pam_unix_auth.so
auth     required       pam_ldap.so     use_first_pass
account  sufficient     pam_unix2.so
account  required       pam_ldap.so
password sufficient     pam_unix_passwd.so
password required       pam_ldap.so     use_first_pass use_authtok debug
session  sufficient     pam_unix2.so
session  required       pam_ldap.so
server:~ #
```

Просмотр логов позволяет обнаружить, что модификация бюджета в LDAP производится от имени актуального пользователя, а не от пользователя-посредника. Это позволяет снова вернуться к ограничениям доступа к полям LDAP.

Сначала удаляем права на запись парольных хешей для пользователя посредника.

```
server:~ # cat /etc/openldap/slapd.access.conf | grep -v "^\(#\|$\)"
access  to dn=".*,dc=office,dc=localnet" attr=userPassword
        by anonymous                            auth
        by self                                 write
        by *                                    none
access  to dn=".*,dc=office,dc=localnet"
        by self                                 write
        by *                                    read
server:~ #
```

Проверка показывает, что с такой настройкой все перестает работать. Тогда приходит время взяться за настройки клиента. Исходно обращение настроено от пользователя посредника, который для клиента представлен как rootdn, что указывает клиенту модификацию полей производить тоже от пользователя посредника. Меняем настройки так, чтобы клиент LDAP не считал указанного ldapbrowser за суперпользователя LDAP и производил все модификации от владельца.

```
server:~ # cat /etc/ldap.conf | grep -v "^\(#\|$\)"
host    127.0.0.1:389
base    dc=office,dc=localnet
```

```
pam_password    exop
nss_base_passwd         ou=People,dc=office,dc=localnet?one
nss_base_shadow         ou=People,dc=office,dc=localnet?one
nss_base_group          ou=Groups,dc=office,dc=localnet?one
binddn cn=ldapbrowser,dc=office,dc=localnet
bindpw browser
server:~ #
```

Теперь все начинает работать снова – парольные хеши в LDAP обновляются успешно. Но поскольку, как было выше настроено в slapd.access.conf, модификация хешей возможна только от владельца, а чтение всех остальных данных разрешено для всех, то пользователь посредник более не нужен вовсе.

```
server:~ # cat /etc/ldap.conf | grep -v "^\(#\|$\)"
host    127.0.0.1:389
base    dc=office,dc=localnet
pam_password    exop
nss_base_passwd         ou=People,dc=office,dc=localnet?one
nss_base_shadow         ou=People,dc=office,dc=localnet?one
nss_base_group          ou=Groups,dc=office,dc=localnet?one
server:~ #
```

И проверка показывает, что это так. Аутентификация производится от анонима, до пользователя, модификация от пользователя, чтение от анонима.

Получается, что пользователь посредник ldapbrowser не упоминается в настройках LDAP сервера, не используется в настройках клиента. Тогда зачем он вообще? Кто первым предложил его создать? Удалим его!

Зададимся вопросом, можно ли было удалить его ранее, на стадии настройки NSS LDAP. Очевидно, нет, так как работа NSS LDAP построена на чтении всей необходимой информации и доставке ее запрашивающим программам. Отказ от использования пользователя-посредника в NSS LDAP привел бы к необходимости допустить чтение LDAP от анонима, что гораздо хуже с точки зрения защиты информации.

Вернемся к настройкам PAM. Установим следующие настройки для задачи password.

```
server:~ # cat /etc/pam.d/passwd | grep ^password
password sufficient pam_unix_passwd.so
password required   pam_ldap.so     try_first_pass ignore_authinfo_unavail
server:~ #
```

Не будем менять макрос с флагами и оставим его в состоянии sufficient. Дело в том, что в задаче password происходит вызов pam_chauthtok, который дважды проходит по цепочке модулей, сначала для проверки, а потом для настоящей смены парольного хеша. И на каждом проходе возможные ответы меняются и, соответственно, нет способа задать тот набор флагов, который отработает в каждом проходе. Все внимание сосредоточим на том, в каком порядке будет передаваться информация между модулями. Использование use_first_pass или use_authtok практически парализует работу pam_ldap и не позволит менять пароль в LDAP независимо от локальной базы. Более мягкая форма try_first_pass сделает возможным выдачу отдельного приглашения на парольную фразу LDAP. Опция

ignore_authinfo_unavail поможет решить проблемы с поврежденными бюджетами LDAP. Полностью получиться так.

```
server:~ # cat /etc/pam.d/passwd | grep -v "^\(#\|$\)"
auth [success=done new_authtok_reqd=done user_unknown=ignore
authinfo_unavail=ignore default=bad] pam_unix_auth.so
auth     required       pam_ldap.so     use_first_pass
account  sufficient     pam_unix2.so
account  required       pam_ldap.so
password sufficient     pam_unix_passwd.so
password required       pam_ldap.so     try_first_pass ignore_authinfo_unavail
session  sufficient     pam_unix2.so
session  required       pam_ldap.so
server:~ #
```

Проверку полученной политики модификации парольных хешей с использованием pam_ldap проведем именно с вышеуказанной настройкой файла скрипта. Задача auth скопирована из файла-скрипта su, но здесь это не важно, так как будет работать только задача password. В реальной работе естественно не помешает добавить проверку паролей на прочность в модуле pam_pwcheck.

3.10.Проверка passwd с pam_ldap.

Проведем проверку аналогично тому так, как это было уже сделано ранее для pam_unix2. Но в этом случае проверка 5 не имеет смысла, поскольку авторизация невозможна для настроек с pam_ldap, если локальный бюджет заблокирован. Будем называть “старым” текущий пароль в соответствующем бюджете.

3.10.1. Проверка passwd с pam_ldap 1.

Оба бюджета присутствуют одновременно. На приглашение вводим старый пароль из локальной базы.

```
atest@server:~> id
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video),513(Domain Users)
atest@server:~> passwd
Changing password for atest.
Changing password for atest
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
atest@server:~>
```

Меняется пароль локальной базы. Если в ответ на приглашение ввести старый пароль из LDAP, то появится сообщение о смене пароля в LDAP.

```
atest@server:~> id
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video),513(Domain Users)
atest@server:~> passwd
```

```
Changing password for atest.
Changing password for atest
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
LDAP password information changed for atest
atest@server:~>
```

Возможно, некоторой неожиданностью будет то, что в последнем случае изменятся одновременно два пароля. То есть произойдет их синхронизация.

```
server:~ # getent passwd | grep ^atest
atest:x:1007:100::/home/atest:/bin/bash
atest:x:1002:513:Office User:/home/atest:/bin/bash
server:~ # getent shadow | grep ^atest
atest:FR.4K6u/EGyjA:12626:0:::::
atest:x:12626::::::0
server:~ # ldapsearch -LLL -H ldap://localhost -D
"cn=ldapadmin,dc=office,dc=localnet" -x -w secret "(uid=atest)" userPassword
dn: uid=atest,ou=People,dc=office,dc=localnet
userPassword:: e1NNRDV9L2ZJbTJaNGlOZTZ0QldWd3VSQW5URzlLTUtJPQ==

server:~ # echo e1NNRDV9L2ZJbTJaNGlOZTZ0QldWd3VSQW5URzlLTUtJPQ== | mimencode
-u ; echo
{SMD5}/fIm2Z4iNe6tBWVwuRAnTG9KMKI=
server:~ #
```

3.10.2. Проверка passwd с pam_ldap 2.

В локальной базе удаляем запись о бюджете в файле passwd. Авторизация происходит с использованием пароля из shadow, но в бюджет из LDAP. При смене пароля попытка ввести в ответ на приглашение старый локальный пароль отвергается и принимается только пароль из LDAP.

```
atest@server:~> id
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
atest@server:~> passwd
Changing password for atest.
Enter login(LDAP) password:
LDAP Password incorrect: try again
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information changed for atest
atest@server:~>
```

Происходит смена парольного хеша только в LDAP. Главное отличие в том, что PAM более не путает приглашения и сразу предупреждает, что нужно вводить именно старый пароль из LDAP.

3.10.3. Проверка passwd с pam_ldap 3.

Локальный бюджет отсутствует, и все происходит совершенно аналогично предыдущему случаю и с тем же результатом.

```
atest@server:~> id
uid=1002(atest) gid=513(Domain Users) группы=513(Domain Users),14(uucp),16
(dialout),17(audio),33(video)
atest@server:~> passwd
Changing password for atest.
Enter login(LDAP) password:
New password:
Re-enter new password:
LDAP password information changed for atest
atest@server:~>
```

3.10.4. Проверка passwd с pam_ldap 4.

Здесь, наоборот отсутствует запись в shadow, но есть в passwd, что “путает” PAM и заставляет его снова запросить старый локальный пароль. Если поверить и так и сделать, то смена пароля завершится неудачей.

```
atest@server:~> id
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video),513(Domain Users)
atest@server:~> passwd
Changing password for atest.
Changing password for atest
(current) UNIX password:
LDAP Password incorrect: try again
passwd: Authentication failure
atest@server:~>
```

Если же не купиться на такую мистификацию и вопреки всему указать старый пароль из LDAP, то все завершается сменой пароля в LDAP, что и ожидалось.

```
atest@server:~> id
uid=1007(atest) gid=100(users) группы=100(users),14(uucp),16(dialout),17
(audio),33(video),513(Domain Users)
atest@server:~> passwd
Changing password for atest.
Changing password for atest
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
LDAP password information changed for atest
atest@server:~>
```

Сведем все полученные результаты в таблицу и попробуем осмыслить.

	pam 1	pam 2	pam 3	pam 4	pam 5
passwd	есть			есть	есть
shadow	есть	есть			блокирован
pass	shadow	shadow	LDAP	LDAP	-
id	passwd	LDAP	LDAP	passwd	-
Old pass	pass/LDAP	LDAP	LDAP	LDAP	-
изменение	shad/ALL	LDAP	LDAP	LDAP	-
2 из 5	OK?	???	OK	???	OK

Сомнительным представляется случай синхронизации паролей полученный в проверке 1. Но так как это производится из сессии локального бюджета, то значит, синхронизация может быть создана только осознанно и намеренно. Более никаким иным способом повредить локальному бюджету не удается, а значит поставленная цель, создать такую политику, при которой локальный бюджет доминирует перед бюджетом из LDAP, достигнута.

3.11.Кризис PAM?

Безусловно, возможности построения политики аутентификации и управления данными бюджетов с PAM весьма разнообразны. Можно даже сказать, что PAM для этого предназначен, если не сказать более, что он именно для этого и был задуман. Но реальное использование PAM в таком качестве выявило некоторые внутренние проблемы. Перечислим их.

1. Создание политики в PAM сильной усложняется, если альтернативных путей доступа к бюджетной информации более чем два. Например, при использовании winbind. Механизм последовательного перебора модулей в стеке PAM не дает возможности создания предварительного управляемого ветвления, как по требованию пользователя, так и в зависимости от иных параметров. Ситуацию усложняет даже применение модулей предварительной проверки, поскольку если альтернативные аутентификационные модули настроены так, что не передают информацию по цепочке.

2. Вторая проблема в том, что параметры PAM передаются только внутри модулей стека одной задачи (в терминологии PAM: внутри цепочки/chain одной подсистемы/facility). И нет возможности связать общими данными два стека, если они созданы в процессе обращения от разных приложений, но в течение работы под одной бюджетной сессией. Например, в зависимости от пути аутентификации выбрать базу бюджетных данных для пользовательской сессии точно такую же, где была произведена аутентификация. Возможно, это уже и не проблема PAM, а проблема UNIX, точнее POSIX.

Здесь опять нельзя не сказать о том, что в MS Windows применяется иной способ аутентификации и управления сессиями. Представляется интересными то, как там бесконфликтно существуют три возможных способа авторизации пользователя в системе. Первое, авторизация в локальной базе, аналогично тому, как пользователи

UNIX авторизуются через passwd и shadow. Второе, авторизация в домене майкрософт. Примерно похоже на авторизацию через NIS и LDAP в UNIX. Третье, авторизация в домене майкрософт, в условиях недоступности контроллера домена. Аналога последнему нет, как и нет пути создания такой авторизации в UNIX. Все перечисленные механизмы авторизации работают одновременно и прекрасно решают проблему пользователей-близнецов. Данные пользовательского бюджета, идентификаторы, права, окружение, назначаются в зависимости от выбранного при аутентификации пользователя.

3. Третья проблема заключается в том, что в PAM допускается использование модулей, например pam_unix2, которые формируют внутри политики, определяемой файлом-скриптом, собственную субполитику, например основанную на жестко заданных предпочтениях автора этого модуля. Дело не в том, что использование pam_unix2 есть проблема лишь дистрибутивной линии SuSE и той абстракции управления, которую несет YaST2. Проблема в том, что такое вообще возможно и допускается спецификациями PAM.

Приложение А. Файлы настроек.

Здесь подытожим полученные настройки на тот случай, если выше рекомендации в отношении некоторых были высказаны недостаточно отчетливо.

А.1.Файл для создания альтернативного root-бюджета в LDAP.

```
server:~ # cat root.ldif
dn: uid=root,ou=People,dc=office,dc=localnet
uid: root
cn: root
sn: root
objectClass: top
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
shadowLastChange: 12089
shadowMax: 10000
loginShell: /bin/bash
uidNumber: 0
gidNumber: 0
homeDirectory: /root
gecos: root
userPassword: {CRYPT}.vSlVrIfg2SZ2
server:~ #
```

Парольный хеш соответствует “lroot”.

А.2.Файл для создания пользователя посредника ldapbrowser.

```
server:~ # cat ldapbrowser.ldiff
dn: cn=ldapbrowser,dc=office,dc=localnet
cn: ldapbrowser
sn: ldapbrowser
objectClass: person
objectClass: top
userPassword: {SSHA}stfV4nrgwlBg3K4dReyUFo8KSAHqOjNo
server:~ #
```

Парольный хеш соответствует “bowser”.

А.3.Конфигурация сервера LDAP.

```
server:~ # cat /etc/openldap/slapd.conf | grep -v ^# | grep -v ^$
include         /etc/openldap/schema/core.schema
include         /etc/openldap/schema/cosine.schema
include         /etc/openldap/schema/inetorgperson.schema
include         /etc/openldap/schema/nis.schema
include         /etc/openldap/schema/samba3.schema
pidfile         /var/run/slapd/slapd.pid
argsfile        /var/run/slapd/slapd.args
modulepath      /usr/lib/openldap/modules
replogfile      /var/lib/ldap/replica.log
database        ldbm
```

```
cachesize       40000
dbcachesize     60000000
suffix          "dc=office,dc=localnet"
rootdn          "cn=ldapadmin,dc=office,dc=localnet"
rootpw          {SSHA}K6n0nTsvOWxO1xPGBN5HoZCAsaO0wV7p
directory       /var/lib/ldap
index   objectClass                             eq
index   ou,cn,sn,displayName                    eq,pres,sub
index   uidNumber,gidNumber                     eq
index   sambaSID                                eq
index   memberUID,uid                           eq,pres,sub
index   sambaPrimaryGroupSID                    eq
index   sambaDomainName                         eq
index   default                                 sub
include         /etc/openldap/slapd.access.conf
server:~ #
```

Парольный хеш соответствует “secret”.

A.4.Настройки доступа базы LDAP.

```
server:~ # cat /etc/openldap/slapd.access.conf | grep -v ^# | grep -v ^$
access  to dn=".*,dc=office,dc=localnet" attr=userPassword
        by anonymous                            auth
        by self                                 write
        by *                                    none
access  to dn=".*,dc=office,dc=localnet"
        by self                                 write
        by *                                    read
server:~ #
```

A.5.Конфигурация клиента LDAP.

```
server:~ # cat /etc/ldap.conf | grep -v ^# | grep -v ^$
host    127.0.0.1:389
base    dc=office,dc=localnet
pam_password    exop
nss_base_passwd         ou=People,dc=office,dc=localnet?one
nss_base_shadow         ou=People,dc=office,dc=localnet?one
nss_base_group          ou=Groups,dc=office,dc=localnet?one
server:~ #
```

Файл с паролем в открытом виде.

```
server:~ # cat /etc/ldap.secret
browser
server:~ # ls -l /etc/ldap.secret
-rw-------    1 root     root            8 Jul 16 21:52 /etc/ldap.secret
server:~ #
```

A.6.Конфигурация NSS.

```
server:~ # cat /etc/nsswitch.conf | grep -v ^# | grep -v ^$
passwd: files ldap
shadow: files ldap
group:  files ldap
hosts:  files dns
networks:       files dns
services:       files
protocols:      files
rpc:    files
ethers: files
netmasks:       files
netgroup:       files
publickey:      files
bootparams:     files
automount:      files nis
aliases:        files
server:~ #
```

A.7.Файл-скрипт PAM для утилиты su.

```
server:~ # cat /etc/pam.d/su | grep -v ^# | grep -v ^$
auth     sufficient     pam_rootok.so
auth [success=done new_authtok_reqd=done user_unknown=ignore
authinfo_unavail=ignore default=bad] pam_unix_auth.so
auth     required       pam_ldap.so     use_first_pass
account  sufficient     pam_unix2.so
account  required       pam_ldap.so
password required       pam_pwcheck.so  nullok
password sufficient     pam_unix2.so    nullok use_first_pass use_authtok
password required       pam_ldap.so     use_first_pass use_authtok
session  sufficient     pam_unix2.so
session  required       pam_ldap.so
server:~ #
```

A.8.Файл-скрипт PAM для утилиты passwd.

```
server:~ # cat /etc/pam.d/passwd | grep -v "^\(#\|$\)"
auth [success=done new_authtok_reqd=done user_unknown=ignore
authinfo_unavail=ignore default=bad] pam_unix_auth.so
auth     required       pam_ldap.so     use_first_pass
account  sufficient     pam_unix2.so
account  required       pam_ldap.so
password sufficient     pam_unix_passwd.so
password required       pam_ldap.so     try_first_pass ignore_authinfo_unavail
session  sufficient     pam_unix2.so
session  required       pam_ldap.so
server:~ #
```

Приложение Б. Выводы, подсказки, заметки.

Резюмируем некоторые мысли, высказанные по ходу работы. Представим их в виде безапелляционных утверждений. Укажем попутно версии тех пакетов, где подобное обнаружено. Но важно, что все проверялось в SuSE Linux v9.0. Следовать ли этим рекомендациям или нет, останется полностью на усмотрение недоверчивых читателей.

Б.1. Если в ходе неосторожных экспериментов аутентификационные подсистемы сервера безнадежно испорчены и система более никак не принимает никакой логин, то всегда остается возможность задать в приглашении загрузчика “init=/bin/sh” и без всякого логина попасть в привилегированный шелл. Главное, не забыть перемонтировать корень в режим записи через “mount -o remount,rw /” перед началом редактирования.

Б.2. Использование аутентификации через NSS LDAP не позволяет скрыть парольные хеши от доступа с локальных систем. Что создает возможности для атаки на пароли с использованием словарей. А требование применения алгоритма {crypt} облегчает задачу подбора паролей по хешам.

Верно для nss_ldap-207-80.

Б.3. Стандартная настройка PAM LDAP в SuSE через pam_unix2, а также и любая другая построенная на pam_ldap с использованием стандартных макросов ведет к созданию условий, как для взлома локальных бюджетов через бюджеты из LDAP, так и наоборот. YaST2 настраивает именно pam_unix2.

Верно для pam-modules-9.0-5, pam_ldap-164-42, yast2-ldap-client-2.8.12-3

Б.4. Работа pam_ldap зачастую невозможна без настройки nss_ldap, или точнее лишена смысла для многих приложений, которые используют стандартные системные вызовы для получения информации о бюджетах и их ресурсах. Проверяется путем очевидного запроса.

```
server:~ # fuser -v /lib/libnss_ldap.so.2 | grep ....m | awk '{print $4}' |
sort -u
amavisd
clamd
httpd
mc
nagios
named
pickup
qmgr
slapd
smbd
sshd
su
server:~ #
```

То есть, для программ из этого списка для работы с LDAP бюджетами достаточно настройки NSS LDAP. Но использование PAM позволяет настроить политику аутентификации применительно к выбранному сервису, и делает возможным использование для данного сервиса данных из LDAP в качестве виртуальных, не системных и не локальных пользовательских бюджетов. Как, например это может быть сделано для почтовых служб.

Верно для pam_ldap-164-42, nss_ldap-207-80.

Б.5. Применение pam_ldap позволяет не только исключить использование пользователя-посредника для чтения парольных хешей с локальной станции, и даже вообще удалить его из настроек, как сервера, так и клиента, но и также повысить защищенность хешей путем применения более совершенных алгоритмов хеширования, чем {crypt}, например {ssha} или {smd5}.

Верно для pam_ldap-164-42, nss_ldap-207-80, shadow-4.0.3-182.

Б.6. Стандартная настройка PAM LDAP в SuSE через pam_unix2, примененная к passwd, работает в режиме, когда пароль LDAP запрашивается специфическим промптом. И при этом приходится дать права на запись парольных хешей в LDAP пользователю посреднику из ldap.conf, пароль которого хранится в открытом виде. Что позволяет, имея права суперпользователя на локальной станции, взломать все бюджеты на удаленном LDAP сервере.

Верно для pam-modules-9.0-5, pam_ldap-164-42, yast2-ldap-client-2.8.12-3

Приложение B. Исправление SuSE pam-0.77.

B.1.Попытка освободить статическую память.

При работе с модулем pam_unix_passwd сборки SuSE возникает ошибка, сообщение о которой выводится при смене пароля, как указание на ошибку освобождения памяти функцией free.

```
atest@server:~> passwd
Changing password for atest.
Changing password for atest
(current) UNIX password:
free(): invalid pointer 0x4007b940!
Enter new UNIX password:
Retype new UNIX password:
free(): invalid pointer 0x4007b940!
atest@server:~>
```

Первая ошибка возникает вследствие того, что в тексте support.c в функции _unix_verify_password в точке ее завершения производится уничтожение временного парольного хеша следующим образом.

```
if (pp)
      _pam_delete(pp);
```

Где _pam_delete это макрос, состоящий соответственно из последовательного вызова двух других макросов. Первого, _pam_overwrite, который затирает нулями созданный хеш, и, второго, _pam_drop, который производит освобождение памяти через вызов free. Проблема только в том, что pp, являющийся указателем на строку символов, в этой месте ссылается на статическую память, возвращаемою процедурой bigcrypt. Отсюда и ошибка.

Правильно будет заменить _pam_delete на вызов _pam_overwrite. К слову сказать, в pam-0.75, используемом в ALT Linux, так и сделано. Но в пакетах, идущих как в SuSE 9.0, так и в SuSE 9.1 эта ошибка, как и та, что обсуждается далее, присутствует. Некоторым оправданием, если такое выражение здесь применимо, может служить то, что pam_unix_* , как уже неоднократно было замечено выше, в SuSE не применяется.

Итак, чтобы исправить надо указанную ранее форму уничтожения pp заменить на другую.

```
if (pp)
      _pam_overwrite(pp);
```

Вторая ошибка имеет аналогичную природу, но локализуется в pam_unix_passwd.c в процедуре pam_sm_chauthtok в самом ее конце. В следующем фрагменте, вызов _pam_delete выполняется над статической памятью.

```
          /* update the password database(s) -- race conditions..? */

          retval = _do_setpass(pamh, user, pass_old, tpass, ctrl,
```

```
                        remember);

    _pam_delete(tpass);
    pass_old = pass_new = NULL;
```

Для исправление надо переделать указанный вызов в _pam_overwrite.

```
    /* update the password database(s) -- race conditions..? */

    retval = _do_setpass(pamh, user, pass_old, tpass, ctrl,
                        remember);

    _pam_overwrite(tpass);
    pass_old = pass_new = NULL;
```

Технологически верной будет следующая последовательность внесения изменений в pam-0.77. Надо сохранить исходные тексты в файлах с суффиксом *.orig . Затем, внеся исправления, создать патч. Патч подложить в директорию сборки пакета. И, подправив соответственно spec, создать исправленный пакет. Именно таким путем высококвалифицированные инженеры SuSE и создали pam-0.77, который доставил столько хлопот.

Обсуждение методов модификации дистрибутивных пакетов не входит в перечень задач этой работы. Поэтому здесь опущен вопрос исправлений многих другие вызовов _pam_delete в pam-0.77, которые, будучи оставленными в неизменности, скорее всего также приведут к сообщениям об ошибках.

B.2.Бесконечный цикл ожидания блокировки.

Смена пароля с помощью passwd приводит к “зависанию” процесса, если используется pam_unix_passwd вместе с pam_ldap, но локальный бюджет отсутствует. Проблема в бесконечном блокирующем цикле в начале процедуры pam_sm_chauthtok из pam_unix_passwd.c.

```
i=0;
while((retval = lckpwdf()) != 0 && i < 100) {
        usleep(1000);
}
if(retval != 0) {
        return PAM_AUTHTOK_LOCK_BUSY;
}
```

Понятно, что внутри цикла пропущен инкремент счетчика. Исправляется его добавлением.

```
i=0;
while((retval = lckpwdf()) != 0 && i < 100) {
        usleep(1000);
        i++;
}
if(retval != 0) {
        return PAM_AUTHTOK_LOCK_BUSY;
}
```

Ошибка наблюдается как в SuSE 9.0, так и в pam-0.75 из ALT Linux. Но в SuSE 9.1 этой проблемы нет. Поскольку при неизменности версии PAM там все же проведены некоторые правки из CVS.

B.3.Потерянное разблокирование.

Но конечно, причина возникшей проблемы с конфликтом блокирования для данной реализации pam_unix_passwd.c в том, что внутри процедуры pam_sm_chauthtok есть пара выходов без разблокирования файлов локальной базы. И именно в тех местах, что наиболее интересны, где формируется ответ PAM_USER_UNKNOWN. Это приводит к потерянным блокировкам в /etc, которые можно наблюдать через ls.

```
server:~ # ls -l /etc/.*lock
-rw-------    1 root     Domain Users        0 Jul 26 12:26 /etc/.pwd.lock
server:~ #
```

Исправляется путем добавления вызова ulckpwdf так, как далее указано.

```
                if (user == NULL || !isalnum(*user)) {
                        _log_err(LOG_ERR, pamh, "bad username [%s]", user);
#ifdef USE_LCKPWDF
                        ulckpwdf();
#endif
                        return PAM_USER_UNKNOWN;
                }
                if (retval == PAM_SUCCESS && on(UNIX_DEBUG, ctrl))
                        _log_err(LOG_DEBUG, pamh, "username [%s] obtained",
                                 user);
        } else {
                if (on(UNIX_DEBUG, ctrl))
                        _log_err(LOG_DEBUG, pamh,
                                 "password - could not identify user");
#ifdef USE_LCKPWDF
                ulckpwdf();
#endif
                return retval;
        }
```

Верно только для SuSE 9.0. В 9.1 и в других реализациях, например в ALT Linux это уже исправлено.

Самое невероятное в том, что патч производящий исправления из B.2 и B.3 присутствует в rpm с исходными текстами PAM, но в спеке сборки он отключен путем комментирования. Наверное, торопились ... к праздникам?!