Albrecht Weinert

Ubuntu for remote services – detours

Abstract and Introduction

This technical report is just a supplement to [29] . [29] is about installing Ubuntu 16.04.1 on a server for remote services. Extending this a bit, we have (Feb. 2017) three distributions running on five target machines.

Until March 2017 [29] contained chapters on things we – and many others – just didn't get to work, and other working solutions were are described there. Usually those detour or paradise lost chapters would be skipped when working with [29] and made this report larger and less readable.

Hence, it was considered just to delete this detour content. Nevertheless, in some circumstances the information there is valuable. So we kept those chapters in this extra report [26]. It is just a supplement to the main report [29] and can't be read without. All background information, list of abbreviations, references, information on the using of names and on the target platforms etc. is only kept there.

On the content

As said, here we describe
detours and errors to avoid respectively to learn from.

Find References, Abbreviations, a collection of useful commands in [29]
and the Table of Contents in the **Appendix**.

The Detours

Using a non-server Ubuntu distribution on a real server

In the hope to get modern HMI as on Windows servers we installed an Ubuntu workstation (non server) distribution

ubuntu-16.04.1-desktop-amd64.iso [18.10.2016 1.513.308.160]

on a "real" server (Fujitsu RX200S5) with two LSI MegaRaid extenders.

This was a mistake/detour by itself. We neither enjoyed the graphical HMI (see below) nor got the RAID drives working.

Using RDP on a server

So far, this was done on a server with the workstation/non server Ubuntu distribution, only. The server distro has no graphics to use, neither locally nor, of course, remotely.
To see the graphic HMI on a remote PC remote do

```
sudo apt-get install xrdp
sudo apt-get install xfce4

echo xfce4-session > ~/.xsession
sudo nano /etc/xrdp/startwm.sh
sudo service xrdp restart
```

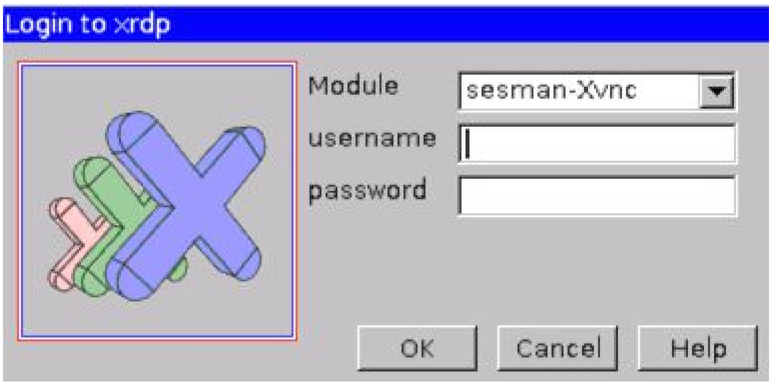
The empty .xsession file made here, according to most recipes won't be used/executed in our configuration – making it may be just tradition?

Anyway, on a workstation, laptop or any other non-server target this remote access might now be used. Nevertheless, its quality is far below Windows to Windows RDP.
Hence consider if you really need or want it with Linux.

Using RDP (on Windows)

Start the RDP client as usual just using the server's DNS name or IP – not minding it being Ubuntu.

On the login frame use sesman-Xvnc + your Ubuntu name and password; ref. Fig.1.



Login to xrdp

Module: sesman-Xvnc

username:

password:

Fig. 1: xRDP-Login
username can be pre-filled using a .rdp configuration file accordingly

Best make and save a RDP configuration file (C:\util\remoteDT\pd321s.rdp e.g.) with the Linux server's access data and perhaps a desktop icon referring to it.

Rationale for using also RDP as admin

For administering a Unix server doing 24/7 work for remote clients terminal/putty is respectively has to be mostly sufficient. The reason to have graphical HMI + RDP is

a) up-to-date comfort and survey,

b) graphical/browser based installation/login/download procedures for SW as well as
c) having a decent non stone age non error prone text editor.

The terminal editors have the charm of sitting at a real Teletype*). Part of the necessary control keys to use will remotely not do the expected; some have unexpected effects on the host machine.

*) Authors personal remark to avoid the usual discussion/rebukes:
Well, long ago I was sitting on a Teletype editing Algol source files with good success. And, obviously
being old enough for having done real work on what is now to be admired in "Deutsches Museum":
I'm also quite able to handle e.g. nano.
nano is one (and not the worst) of the Teletype resembling editors.
But nowadays being forced to work this way again I do consider an impertinence.

Notwithstanding, 99% of the procedures described below or in [27..29] can be done by terminal/ putty access, which can also transfer text via clipboard. ("Terminal" is Unix' nickname for any shell/command line interpreter.)

Quite shamefully, Ubuntu's xRDP won't handle the clipboard in neither direction nor will it provide other resources. Who knows Windows-to-Windows-RDP will be in for a real disappointment. Replacing the remote Windows WS by an Ubuntu one, makes the clipboard handling worse.

Another problem is a strange different behaviour of xRDP compared to the (bash) shell.
On a locally attached terminal and on putty the $PATH is:

```
/home/weinert/bin:/home/weinert/.local/bin:/usr/local/sbin:
/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:
/snap/bin:/usr/local/java/jdk/bin
```

Here one is only wondering who stupidly put the games in. Well, as dealing with the standard distribution at the moment, perhaps it must be so *).

*) One server distribution had no games on the path, originally. But after some updates
and without installing or using games it suddenly had, too.

On xRDP the path is

```
/bin:/usr/sbin:/usr/bin:/user/X11R6/bin:/usr/local/bin
```

which is slightly more than nothing. Remote login – as same user! – only makes sense if the view to and the behaviour of the system is almost exactly the same. Differences that would make sense, on the other hand, as no animated background images (crashing the remote session) are missing.

The remedy might be to add to ~/.bashrc:

```
if [ -d "$HOME/bin" ]; then
    if [[ $PATH =~ $HOME/bin ]]; then :
    else PATH="$HOME/bin:$PATH"
    fi
fi
JAVA_HOME=/usr/local/java/jdk
if [ -d "$JAVA_HOME/bin" ]; then
    if [[ $PATH =~ $JAVA_HOME/bin ]]; then :
    else PATH=$PATH:$JAVA_HOME/bin
    fi
fi
export JAVA_HOME
export PATH
```

The behaviour on xRDP is controlled also by the files

`/etc/xrdp/xrdp.ini, /etc/xrdp/sesman.ini`

The original /etc/xrdp/sesman.ini contains a line TerminalServerUsers=tsusers in section [Security]. This would exclude all non tsusers from login, should this group exist. Simply doing

```
sudo addgroup tsusers
sudo service xrdp restart
```

will exclude all users from xRDP – without impeding their FTP rights. Assuming no user being in group tsuser all users will be excluded from RDP now.

```
sudo usermod -a -G tsusers albrecht
sudo service xrdp restart
```

will bring xRDP back to one bashable user (albrecht).

Collabnet Subversion Edge

This was done on the Fujitsu server (see [29]) with the standard distro, only. The non-detour is having a separate Apache 2 installation, bringing in pure SVN afterwards.

Installing Subversion Edge 5.1.4 (Linux 64-bit) also brings the Apache Webserver. Use

```
env
```

or (with Frame4J installed)

```
java ShowProps
```

to ensure having JAVA_HOME=/usr/local/java/jdk set and on the PATH.

Download CollabNetSubversionEdge-5.1.4_linux-x86_64.tar.gz and and/or follow CollabNet's readme. Having set all rights you can navigate locally to

`localhost:3343/csvn/`	the SVN admin console

and after starting also remotely to

`https://pd321s`	the Web (it works so far)
`https://pd321s/svn`	the repo list (ugly so far)
`https://pd321s:4434`	the SVN admin console

Problems with Edge's standard installation:

- Installs within home/<user>
- Brings own certificate
  Hostname: svnedge.collab.net
  Valid: from Fri, 27 Aug 2010 13:05:48 GMT until Mon, 24 Aug 2020 13:05:48 GMT
  Issuer: CollabNet Subversion Edge, CollabNet, Inc., Brisbane, CA, US
- Has the old fashioned text file based access rules with own groups.
  Users and groups there are incompatible and non connectable to (standard) Ubuntu users
- Partly unclear conflicts with web based management console when
  starting to use the Apache2 web-server heavily with multi domains. php, etc.

To be more clear, contrary to former Collabnet distributions, Edge's webserver won't really work. Edge brings it's own small minified (spoiled ?) Apache 2 (2?) in concurrence and incompatible with the existing "real" one. For just SVN and nothing more it is OK – just for those not interested in having an Apache 2.

So, out of many reasons, the standard svn http(s) installation will need thorough modifications or most probably a re-install. Getting rid of the standard csvn is just stopping the servers (..bin/csvn/bin/csvn stop) and deleting the directory.

Collabnet Subversion – Installation upon Apache 2

This was done on the Fujitsu server with Ubuntu server and a ready (feature complete to our requirements) installation of Apache 2. Well, Apache 2 installation ([29]) was a success.

Now we'll add CollabNet's subversion server.

On a (Windows) workstation we log in at https://www.open.collab.net/downloads/ and download those files (selecting OS Linux):

```
02.12.2016        18.279.967 CollabNetSubversion-client-1.9.4-2.x86_64.rpm
02.12.2016         5.783.836 CollabNetSubversion-server-1.9.4-2.x86_64.rpm
02.12.2016        12.531.283 CollabNetSubversion-extras-1.9.4-2.x86_64.rpm
```

The first two will have to be installed in that sequence to have a SVN server. The third one is optional and brings "ViewVC + Python Bindings for Subversion 1.9.4 (Linux x86_64). With FilleZilla we transfer the files downloaded to ~/Downloads.

Hint: Going via the Windows server could perhaps have been avoided. But we haven't found a way to wget those files as Collabnet's browser download requires filling out some login and forms.

```
dir -G Downloads
```

```
-rw------- weinert  18279967 2016-12-02 10:32
                            CollabNetSubversion-client-1.9.4-2.x86_64.rpm

-rw------- weinert  12531283 2016-12-02 10:32
                            CollabNetSubversion-extras-1.9.4-2.x86_64.rpm

-rw------- weinert   5783836 2016-12-02 10:32
                            CollabNetSubversion-server-1.9.4-2.x86_64.rpm

-rw-rw-r-- weinert    532634 2016-10-14 15:33 frame4j.jar
-rw-rw-r-- weinert 181435897 2016-06-27 18:02  jdk-8u102-linux-x64.tar.gz
-rw-rw-r-- weinert  93436489 2016-09-24 01:36      jdk-8u112-docs-all.zip
```

Compared to the Java installation files got directly by wget (see [29]) we just see fewer rights.

Go ahead by:

```
sudo apt install rpm
cd Downloads/
wget http://www.collab.net/nonav/downloads/subversion/gpg-key/RPM-
GPG-KEY-csvn.asc
```

```
sudo rpm --import RPM-GPG-KEY-csvn.asc
sudo apt-get install alien dpkg-dev debhelper build-essential
sudo alien -c CollabNetSubversion-extras-1.9.4-2.x86_64.rpm
sudo alien -c CollabNetSubversion-client-1.9.4-2.x86_64.rpm
sudo alien -c CollabNetSubversion-server-1.9.4-2.x86_64.rpm
```

By above commands we prepared the installation of the Collabnet .rpm files downloaded. As it turned out, they can't be used directly. Hence, the first "install rpm" was superfluous. We installed alien and used it to convert the .rpm. The last three commands (still in ~/Downloads) made:

```
-rw-r--r-- 1 root    root     13180044 2016-12-02 11:10
                            collabnetsubversion-client_1.9.4-3_amd64.deb
-rw-r--r-- 1 root    root      7604340 2016-12-02 11:09
                            collabnetsubversion-extras_1.9.4-3_amd64.deb
-rw-r--r-- 1 root    root      3660672 2016-12-02 11:11
                             collabnetsubversion-server_1.9.4-3_amd64.deb
```

Install the command line tools (step 1)

Now we can install the SVN command line tools:

```
sudo dpkg -i collabnetsubversion-client_1.9.4-3_amd64.deb
/opt/CollabNet_Subversion/bin/svn --version
```

If the first command went well the second one will show something like

```
svn, Version 1.9.4 (r1740329)
   übersetzt am May  4 2016, um 19:53:40 auf x86_64-unknown-linux-gnu
Copyright (C) 2016 The Apache Software Foundation.
```

We modified /etc/profile already for java (see [29]). Now we change one line near the end from

```
PATH=$PATH:$JAVA_HOME/bin
```

to

```
PATH=$PATH:$JAVA_HOME/bin:/opt/CollabNet_Subversion/bin
```

Afterwards the SVN command line tools work without extra ado:

```
mkdir svn_work
cd svn_work/
svn checkout https://ai2t.de/svn/albrecht
cd albrecht
diR
```

In that case we consented to store the (ai2t.de SVN server) user:password in plain text as all else caused nothing but pain on other platforms so far. The credentials are then stored in a cryptically named file in ~/.subversion/auth/svn.simple/

-rw-rw-r-- 1 weinert weinert 158 2016-12-02 13:07 74a572dce6caedeafa0084a567883844

Here just letting all read seems the real bug. Doing the following had no negative effects, yet:

```
chmod -R o-r .subversion/auth/svn.simple/
```

Install the server upon Apache 2 (step 2)

The client installation was a prerequisite for installing the server and then do the configuration. One is warned to to install ViewVC (i.e. the extras) before configuring, should the extras be wanted. Assuming the extra's blocking or de-installation being feasible if not or no more wanted, it was decided to install the extras now, i.e. in the right sequence.

```
cd ~/Downloads/
sudo dpkg -i collabnetsubversion-server_1.9.4-3_amd64.deb
sudo dpkg -i collabnetsubversion-extras_1.9.4-3_amd64.deb
sudo shutdown --reboot now
```

The reboot was triggered by a message from the server installation.

Looking at the installation something seems to be totally wrong. The installation brings its own Apache configuration and has not installed any new (svn available) modules nor configs in our well running Apache 2.

To have a full comfortable (workstation FTP) look do:

```
mkdir /home/albrecht/mountSioux/etcOpt
sudo chown root:web_admin -R /etc/opt
dir /etc/opt
```

```
drwxr-xr-x 7 root root 4096 2016-12-02 14:19 CollabNet_Subversion
```

```
sudo chmod -R g+wr   /etc/opt
sudo chmod g+wrx,o+rx $(find /etc/opt -type d)
dir /etc/opt
```

```
drwxrwxr-x 7 root web_admin 4096 2016-12-02 14:19 CollabNet_Subversion
```

Well with the clear (graphical) view, suspecting "all wrong" manifests itself as wouful truth.
CollabnetSubversion server and extras do

- neither seem to work with Apache 2.4
- nor install themselves into Apache 2.

So, to end this detour, we better remove them:

```
sudo apt-get --purge remove  collabnetsubversion-extras
sudo apt-get --purge remove  collabnetsubversion-server
sudo apt-get --purge remove  collabnetsubversion-client
sudo apt autoremove
sudo apt-get update
sudo apt-get upgrade
```

Then

```
sudo nano /etc/profile
```

and delete :/opt/CollabNet_Subversion/bin from the path.

Resumme on the CollabnetEdge + CollabnetSubversion detour

While installing Ubuntu 16.04 to multiple targets (as reported on in [29]) we have gone the Collabnet trail two times without the expected success.

Asininely would be one explanation for making the same error twice.
The acceptable (hopefully) excuse is:

We had only the very best experiences with Collabnet's distributions on Windows servers since 2001. Not wanting MS' internet information server, in those days we would not get Apache running to our specification. The reason was a (total) lack of Windows support from Apache's developers community. They hadn't Windows on their radar or wanted to crusade people away from Microsoft by ignoring or sneering at their needs.
Collabnet's SVN server distribution was **the** rescue then. And it brought a good full grown, fully usable Apache server installation on Windows as dowry.

Here (Ubuntu 16.04, Apache 2.4) and now (end 2016) this, alas, seems the other way round.

We strongly recommend installing pure Apache 2.4 (no RAMP, no LAMP no Collabnet) and installing/integrating “pure” extras (PHP, SQL if needed pure SVN and ...) afterwards.

And, btw. “pure” SVN is by Collabnet, too.

Open LDAP server

It happens far to often that newly installed features and services, like e.g. FTP, SQL, HTTPS (Apache) and SVN, each bring their own user base and authentication in addition to the basic OS' one. At least this happens easily if not fought against from start of installation. We do need

+ a common user / group base for the system and all services

If not in a Windows AD domain LDAP seems the best choice with the chance to unite the separate user bases, as the OS and most applications are assumed to be LDAP aware. And if mixing non-Windows systems into AD the AD-LDAP server may be the bonding, too.

But with Ubuntu 16.04, both standard and server distribution, openLDAP (server) could not be made operational.

The root of the problem seems to be the new (since Ubuntu 8) so-called "slapd-config method". This fundamental change seems to have three goals:

- making configuration work more comfortable by no more having to touch openLDAD configuration text files as sudo. (In that course most of those files were omitted, or lost their role. Nevertheless they are still omnipresent in actual documentation and guidelines.)
- enhancing the LDAP server's availability by making former service restarts after (sudo) modifying configuration files superfluous
- making basic server and fundamental LDAP configuration more secure by having it in separate LDAP DIT (no one has nor can get access to)

To our and many other's experience these three goals met with very little success, at best.

This is just hypothesis. Nevertheless trying all googled tricks to get around this bug has cost us days with no success here and progress in other field.

We could:

- wait some months for the bug ([3], [4]) go away and the documentation become fully consistent with the new configuration approach: the "slapd-config method" (No, we could not.)
- wait for the 99th published trick / work around to get all working, *)
  - look for an alternative approach or ....?

Before other approaches the total de-install is done by the commands or a suitable subset of them:

```
sudo service slapd stop
sudo service nscd stop

sudo apt-get purge gnutls-bin ssl-cert libpam-ldap nscd lapd ldap-utils

sudo apt autoremove
```

The approach chosen:

In this situation we decided to install openLDAP without LDAP server respectively, to be precise, and with no extras (TLS) nor configurations for use as server. This way, LDAP is just to be used as a local common Id base for all users except the one originally born by Ubuntu installation. All additional Ubuntu users are to be made in LDAP and "PAMmed" to the OS and hence be available for those applications able to use OS accounts. All others should be LDAP aware and content with a rudimentary local server function. With this approach we got the OS, FTP, Apache, SVN and else (also) with LDAP users on one of the “real” Fujitsu servers.

On other machines LDAP didn't work at all. So we omitted it completely. From the application's point of view no LDAP or “LDAP PAMmed only” makes no difference.

*) In between, in [12], we read a consistent explanation of the openLDAP disaster and a potential remedy in the course of installation. As too late for the current installations we haven't given it a try,yet.

LDAP – not as server

Hence, the conclusion was:
As of 16.04 / November 2016 the Ubuntu / openLDAP combination is not server ready. In our case this was shown in many "rounds", some of which less or more respectively fully followed [1], chapter 7,1,1.1. To start with [1] says:
"The suffix (or base DN) of this instance will be determined from the domain name of the localhost. If you want something different, edit /etc/hosts and replace the domain name with one that will give you the suffix you desire. ... You can revert the change after package installation." [sic!]

Well really. One has to touch a configuration text file concerning networking, editing a line which since centuries says just "localhost". This error prone procedure seems at least contradict the "LDAP is configured by LDAP" goals of the "slapd-config method".

Nevertheless we wanted an unique ID management for all planned server applications. Almost all can be made either LDAP aware or they use OS IDs. And as the OS itself can be made or "PAMmed" LDAP aware the difference between LDAP and OS users is partly invisible. So, LDAP still seemed THE choice if not to say the only way to unique ID management for the bunch of planned server applications.

In this situation we decided to install openLDAP without LDAP server.
To be precise "without server" means with no extras for any (real) use as published directory server. And it means using the basic configuration as installed and doing no more than

- just adding some OUs, groups and users

using not more than

- their names, passwords and memberships.

The approach is to restrict openLDAP completely to local use and to be an substitute or add-on to the password and groups files. In the end it worked for one target.

We'll omit the many failures with building/using openLDAP and their frustrating protocols.
This restricted local approach is not much better than “pamming all services to the olde Unix/Linux local users and groups – which can be done to a large extend. Hence we consider it a detour.
This restricted approach worked on one Fujitsu server, only. It failed, as said, on all other targets. Hence we consider this detour also as failure.

To use/make it we prepare respectively modify three files to start with. first is /etc/hosts:

```
127.0.0.1       pd321s.weinert-automation.de    pd321s
192.168.89.6    pd321s.fb3-meva.fh-bochum.de    pd321s
193.175.115.6   pd321s.hs-bochum.de   pd321s

## ::::: leave all else as is
```

The first line was just "127.0.0.1 localhost" before and should be changed back, eventually.
Then we have two ldifs, the first one to make some groups needed:

```
# ~/admin/ldap/batch/add3Gr.ldif
# Add OUs: People and Groups
# Add Groups: web_admin svn_admin ftp_admin  5000,,...
dn: ou=People,dc=weinert-automation,dc=de
objectClass: organizationalUnit
ou: People

dn: ou=Groups,dc=weinert-automation,dc=de
objectClass: organizationalUnit
ou: Groups
```

```

dn: cn=svn_admin,ou=Groups,dc=weinert-automation,dc=de
objectClass: posixGroup
cn: svn_admin
gidNumber: 5001

dn: cn=ftp_admin,ou=Groups,dc=weinert-automation,dc=de
objectClass: posixGroup
cn: ftp_admin
gidNumber: 5002
```

The second .ldif is to make an additional user:

```
# ~/admin/ldap/batch/add1Us.ldif
# Add user:   albrecht     6000 .... in group 5000
# user after  add 3Gr.ldif
dn: uid=albrecht,ou=People,dc=weinert-automation,dc=de
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
uid: albrecht
sn: Weinert
givenName: Albrecht
displayName: A. Weinert
uidNumber: 6000
gidNumber: 5000
userPassword: ****
cn: Albrecht Weinert
loginShell: /bin/bash
homeDirectory: /home/albrecht
```

Having prepared those three files do:

```
sudo apt install slapd ldap-utils
```

In the process give your LDAP-admin password (we...r) twice. Then test by:

```
ldapsearch -x -LLL -H ldap:/// -b dc=weinert-automation,dc=de dn
```

```
dn: dc=weinert-automation,dc=de
dn: cn=admin,dc=weinert-automation,dc=de
```

Then add the OUs and groups:

```
ldapadd -x -D cn=admin,dc=weinert-automation,dc=de -W -f admin/ldap/batch/add3Gr.ldif
```

```
adding new entry "ou=People,dc=weinert-automation,dc=de"
adding new entry "cn=svn_admin,ou=Groups,dc=weinert-automation,dc=de"
adding new entry "cn=ftp_admin,ou=Groups,dc=weinert-automation,dc=de"
```

Next we "PAM" the OS to the new LDAP. We do this before creating a new user.

A) N.b.: All users except the first (admin/sudo) one ("weinert" here) will be made in LDAP!
The other exception will be a user and group made by Apache 2 installation (see below).

B) We probably might have made or "ldifed" users before "PAMing" the OS
but we want to avoid any side effects of trials to login or else.

"PAMming" the OS is:

```
sudo apt-get install libpam-ldap nscd
```

In the process answer:

```
ldap://127.0.0.1
dc=weinert-automation,dc=de
3 yes no
cn=admin,dc=weinert-automation,dc=de
w......r
```

Then in file /etc/nsswitch.conf modify "compat" to "ldap compat" three times, so it looks like:

```
# /etc/nsswitch.conf
#
passwd:         ldap compat
group:          ldap compat
shadow:         ldap compat
gshadow:        files

hosts:          files dns
networks:      files
### etc. pp. leav all else as is
```

Then do:and add at the end of /etc/pam.d/common-session :

```
session required    pam_mkhomedir.so skel=/etc/skel umask=0022
```

For mounting user specific application directory trees from other places in a uniform manner, we recommend to prepare some standard directories as optional mount points (see above) in the new user's auto-generated home directory :

```
sudo mkdir /etc/skel/ftp
sudo mkdir /etc/skel/www
```

and then do:

```
sudo /etc/init.d/nscd restart
```

Now we make respectively "ldif" our first new (OS) user "albrecht":

```
ldapadd -x -D cn=admin,dc=weinert-automation,dc=de -W -f
admin/ldap/batch/add1Us.ldif
```

```
adding new entry "uid=albrecht,ou=People,dc=weinert-automation,dc=de"
```

```
dir /home/
compgen -u
```

do not show the user "albrecht" before he logged in. Afterwards both works and

```
dir /home/albrecht/
```

shows the new users home directory accordingly made (cf. /etc/skel/).
This new user "albrecht" can log in on site and remote but is not sudo.

Adding LDAP authentication to services

The Apache 2 basic installation made "www-data:www-data" as just Ubuntu user and group (both not LDAP). www-data:www-data is the user:group the service apache2 runs as, which is configured so in /etc/apache2/envvars. It is yet to be determined, if we want those two at all or if we had them better in LDAP, according to our approach. As of now we leave them as is.

In this first example/proof of concept we want the information given by phpinfo() be restricted to authenticated users belonging to the web_admin LDAP group. To publish this PHP configuration information is considered critical by some experts. Hence, that should be done anyway.

Independent from using LDAP or not, one has force restricted/confidential content to https. Do get this we want

a) the https/443 access to /var/www/html/info/ for group web_admin only

and

b) the http/80 access to /var/www/html/info/ forced / rewritten to https/443.

Here b) restricts all traffic involving authentication to TLS, which is just mandatory. Hence, we must touch the ssl (a) and the non ssl (b) site. To start we copy the running configurations:

```
sudo cp -p  /etc/apache2/sites-available/default-ssl.conf
/etc/apache2/sites-available/pd321s-ssl.conf
```

```
sudo cp -p  /etc/apache2/sites-available/000-default.conf
/etc/apache2/sites-available/pd321s.conf
```

```
dir /etc/apache2/sites-available/
```

```
-rw-rw-r-- 1 root web_admin 1332 2016-03-19 10:48 000-default.conf
-rw-rw-r-- 1 root web_admin 6453 2016-11-25 14:22 default-ssl.conf
-rw-rw-r-- 1 root web_admin 1332 2016-03-19 10:48 pd321s.conf
-rw-rw-r-- 1 root web_admin 6453 2016-11-25 14:22 pd321s-ssl.conf
```

To force the authentication, as said, we both to modify the virtual host 80, here in file /etc/apache2/sites-available/pd321s.conf, and the virtual host 443 in a /etc/apache2/sites-available/pd321s-ssl.conf:

```
<VirtualHost *:80>
  ServerAdmin webmaster@localhost
  DocumentRoot /megaRaid/Extra/sites/html
  ServerName PD321S

  ServerAlias www.PD321S
# we do not want www.pd321s. And who uses will get https://, too.
  RewriteEngine On
  RewriteCond %{HTTP_HOST} ^www.(([a-z0-9_]+.)?pd321s)$ [NC]
  RewriteRule .? https://%1%{REQUEST_URI} [R=301,L]

# all under info/ gets https and authentication
  <Location /info>
   RewriteEngine On
   RewriteCond %{HTTPS} off
   RewriteRule (.*) https://pd321s%{REQUEST_URI} [R=301,L]
  </Location>

# all under svn/ gets https and authentication
  <Location /svn>
   RewriteEngine On
   RewriteCond %{HTTPS} off
   RewriteRule (.*) https://pd321s%{REQUEST_URI} [R=301,L]
  </Location>
</VirtualHost>
```

Enable this by:

```
sudo a2dissite 000-default
sudo a2ensite pd321s
sudo a2enmod rewrite
sudo service apache2 restart
```

Then make /etc/apache2/sites-available/pd321s-ssl.conf look like:

```
<VirtualHost *:443>
  ServerAdmin webmaster@localhost
  DocumentRoot /megaRaid/Extra/sites/html
  ServerName PD321S

ServerAlias www.PD321S
# we do not want www.pd321s. Keeping subdomains here should have
# no effect as they should have gone to other virtual hosts
  RewriteEngine On
  RewriteCond %{HTTP_HOST} ^www.(([a-z0-9_]+.)?pd321s)$ [NC]
  RewriteRule .? https://%1%{REQUEST_URI} [R=301,L]

  SSLEngine on
  SSLCertificateFile     /etc/ssl/certs/ssl-cert-vsftpd.pem
  SSLCertificateKeyFile         /etc/ssl/private/vsftpd.pem

    <FilesMatch "\.(cgi|shtml|phtml|php)$">
      SSLOptions +StdEnvVars
    </FilesMatch>
    <Directory /usr/lib/cgi-bin>
      SSLOptions +StdEnvVars
    </Directory>

#  Adminstrative information   web_admin members only
    <Directory /megaRaid/Extra/sites/html/info>
      AuthName "PD321S info"
      AuthType Basic
      AuthBasicProvider ldap
      AuthLDAPURL "ldap://127.0.0.1/dc=weinert-automation,dc=de?uid?sub"

      AuthLDAPGroupAttributeIsDN off
      AuthLDAPGroupAttribute memberUID

      Require ldap-group cn=web_admin,ou=Groups,dc=weinert-automation,dc=de
    # require valid-user
    </Directory>

 <Location /svn>
  DAV svn
  SVNParentPath /megaRaid/Extra/sites/repos
  ######   rest see in chapter on SVN
 </Location>
</VirtualHost>
```

After having this file done, enable it by:

```
sudo a2dissite default-ssl
sudo a2ensite  pd321s-ssl
sudo service apache2 reload
```

Testing it may require making your browser forget all.

SAMBA – to join an AD domain

This was, in our cases, a painful flop, too. Here the detour and the remedy is reported on in [28].

Appendix

Miscellaneous commands

In [29] this is more or less an anthology of useful and proven tips. Here you see only an excerpt/supplement mainly concerning LDAP.

See all users

```
compgen -u
getent passwd
```

```
ldapsearch -x -b dc=weinert-automation,dc=de  -s sub
"objectclass=posixAccount" ## this only with LADAP and right dc=
```

See all groups

```
compgen -g
getent group
groups
```

```
ldapsearch -x -b dc=weinert-automation,dc=de  -s sub
"objectclass=posixGroup"  ## this only with LADAP and right dc=
```

Have a look at LDAP (if we have one)

```
sudo ldapsearch -Q -LLL -Y EXTERNAL -H ldapi:/// -b cn=config dn
```

```
sudo ldapsearch -x dn
ldapsearch -H ldap:/// -x -s base -b "cn=subschema" objectclasses
| grep -i group
```

```
ldapsearch -H ldap://127.0.0.1  -x -s base -b "cn=subschema"
objectclasses
```

See all the LDAP with all detail

```
sudo slapcat  -b dc=weinert-automation,dc=de
```

The (.ldif) output of this command may be stored as backup file and fed (<) to slapadd for restore.

Info on one LDAP group or one user

```
ldapsearch -x -b dc=weinert-automation,dc=de  -s sub cn=web_admin
ldapsearch -x -b dc=weinert-automation,dc=de  -s sub "uid=ftp22"
```

Make and delete LDAP user or group

Making one or more users etc. is best done by preparing and applying .ldif:

```
ldapadd -x -D cn=admin,dc=weinert-automation,dc=de -W -f
admin/ldap/batch/add_ftp22.ldif
```

Deleting may require the full DN even when uids are unique:

```
ldapdelete -x -D cn=admin,dc=weinert-automation,dc=de -W
"uid=ftp22,ou=People,dc=weinert-automation,dc=de"
```

Add user(s) to LDAP group

```
ldapmodify -x -D cn=admin,dc=weinert-automation,dc=de -W -f
admin/ldap/batch/add_ftp22toWebAdmin.ldif
```

```
# ~/admin/ldap/batch/add_ftp22toWebAdmin.ldif
# Add user:   ftp22 into goup web_admin

dn: cn=web_admin,ou=Groups,dc=weinert-automation,dc=de
changetype: modify
add: memberUid
memberUid: ftp22
```

Abbreviations

see [29]

References

see all references in [29]

[26] Albrecht Weinert, Ubuntu for remote services, Detours, March 2017,
Supplement to [29]
This paper (the last actual version): a-weinert.de/pub/ubuntu4remoteDetours.pdf

[29] Albrecht Weinert, Ubuntu for remote services, Report, November 2016,
The full report which this is supplement to: a-weinert.de/pub/ubuntu4remoteServices.pdf

Table of Content

Dr. Albrecht Weinert is computer science professor at
Bochum University of Applied Sciences or Hochschule Bochum.
He is founder and director of
MEVA-Lab – Laboratory for versatile distributed applications –
as well as of the service provider weinert – automation.
albrecht@a-weinert.de

Rev. 01 16.03.2017