Kerberized NFS summary (July 17th 2016 update)

Starting point: server1 and server2 have joined the Kerberos realm as IPA servers. Use **ipa-client-install** on both servers to do this. No worries, rumor goes that on the exam this has been done for you. As at this point the exams are still based on 7.0, this procedure is verified on 7.0, but should also work on 7.2 and later. On all participating servers, the DNS resolver is set to the IPA server

IMPORTANT: Do NOT use online repositories on either server involved. Only install software from the OS DVD's. (To make this easy, I've created a repository based on the DVD Packages directory on labipa, removed all "live" repositories and created my own repository file)

1. On the IPA server: ensure that DNS records exist for the hosts, ensure that the IPA DNS resolver is set to the IPA server itself
2. On ipa: **kinit admin**
3. On ipa: **ipa service-add, enter nfs/server1.example.com**
4. On server1, which is going to be used as the NFS server: **kinit admin**
5. Op server1: **ipa-getkeytab –s labipa.example.com –p nfs/server1.example.com –k /etc/krb5.keytab**
6. On server1: **klist –k**
7. On server2: **klist –k**
8. On server1: **mkdir /secureshare**; **semanage fcontext –a –t nfs_t "/secureshare(/.*)?"; restorecon –Rv /secureshare**
9. Set some permissions on /secureshare. It's ugly, but **chmod 777 /secureshare** will do it anyway :-)
10. On server1: **echo /secureshare *(sec=krb5p,rw) >> /etc/exports; systemctl start nfs-server**
11. On server1: **systemctl start nfs-secure-server**
12. On server1 (only if you're using 7.0, not required in later versions) **systemctl enable nfs-secure-server**
13. On server1: **showmount –e localhost**
14. On server1: **firewall-cmd --permanent --add-service=nfs; firewall-cmd --reload**
15. On server2: **yum install -y nfs-utils**
16. On server2: **systemctl enable nfs-secure**; **systemctl start nfs-secure**
17. On server2: **mkdir /securenfs**
18. On server2: **mount -o sec=krb5p server1.example.com:/secureshare /securenfs**
19. Use **su - ldapuser1** to open a session as ldapuser1
20. Type **cd /securenfs**, it won't work as the ldapuser does not have any Kerberos credentials (it's just not in the su - procedure)
21. Type **kinit ldapuser1** and enter the password. Repeat step 19, you'll have access now
22. Type **touch /securenfs/myfile**. You'll be able to create a file and this file will be owned by LDAPuser1

23. Add the following line to /etc/fstab and reboot to ensure the working
server1.example.com:/secureshare /securenfs nfs sec=krb5p 0 0

Issues:

- The **kinit admin** command fails on the servers
  - The servers are not joined to the Kerberos domain
- On 7.0, unsupported mount option on the client
  - This issue is RPM related, it occurs on a minimal installation and not on a server with gui pattern.
  - Use **yum groups install "Network File System Client"** and try again
  - Make sure that you've started and enabled the nfs-secure service (7.0)
- Access denied by server while mounting …
  - This is often a silly typo
  - Or related to missing packages: see above
  - Are you sure that you are mounting on the right server?
  - Check that nfs-secure-server service is running
- On all versions: permissions in the share are mapped to nfsnobody
  - Fix by setting **use_fully_qualified_names = false** in sssd.conf
- (Unverified): on occasion it has helped me to re-generate the Kerberos credentials for the NFS server, which is why I've included this in this document.
- It can help to re-create the keytab file on server2 as well. To do this, use the following:
  - On labipa, remove the host entry for server2
  - On labipa, use **ipa service-add** and add a new host keytab for server2
  - On server2, use **ipa-getkeytab -s labipa.example.com -p host/server2.example.com -k /etc/lrb5.keytab**

False solutions:

- There is NO need to use anything but the default keytab file on RHEL 7.0 on the client. You DO need the keytab file on the server though
- If your server has joined the IPA domain, time will be synchronized
- There is no need to work with idmapper or related options, the GSSAPI takes care of this functionality now
  - Implemented by rpc.svcgssd on the server and rpc.gssd on the client, these are started through nfs-secure and nfs-secure-server