C2115 Practical introduction to supercomputing

Lesson 5

Petr Kulhánek

kulhanek@chemi.muni.cz

National Centre for Biomolecular Research, Faculty of Science
Masaryk University, Kamenice 5, CZ-62500 Brno

Content

- Authentication
  - Authentication vs Authorization
  - Secondary authentication in supercomputer centers
    - Kerberos
    - SSH keys
  - Configurations, packages

Authentication vs Authorization

Authentication (from German Authentisierung) is the process of verifying the claimed identity of an entity. Authentication is usually followed by authorization, which is consent, approval, access, or execution of a particular operation by the entity.

Authorization is the process of obtaining consent to perform an operation, allowing access to somewhere, to someone or something (not only in terms of access to specific spaces or to a person, but also access to information, functions, program objects and the like).

The most common way of **primary authentication** is a combination of login and password (local clusters, WOLF, MetaCentrum). In IT4I, primary authentication is only enabled using ssh keys.

Supercomputers usually consist of a large number of compute nodes and it would be very impractical or impossible (e.g., when running batch jobs) to prove yourself with a password every time you log in to a compute node. Therefore, during **secondary authentication**, a different technique is used.

wikipedia.org

Secondary authentication

Primary authentication creates a state that is later used for authentication (secondary authentication) without having to re-enter the password. This condition may or may not be limited in time. The most commonly used are Kerberos or ssh keys.

Our local clusters (WOLF, sokar, pip, ivavik) and MetaCentrum:

- e-infrastructure CESNET (authentication and authorization controlled by Perun)
- Kerberos (realm META)
- User accounts are managed by Perun (instance CESNET)

https://einfra.cesnet.cz

IT4I Supercomputer Center:

- ssh keys

Summary of information for NCBR and CEITEC-MU (applications, accounts,…):

https://einfra.ncbr.muni.cz

Exercise 1

1. Log in to the Perun environment.
2. Practically: Perun from the point of view of the user and the administrator.
3. Where can I change the password for eINFRA account?

Kerberos

https://en.wikipedia.org/wiki/Kerberos_(protocol)

Kerberos

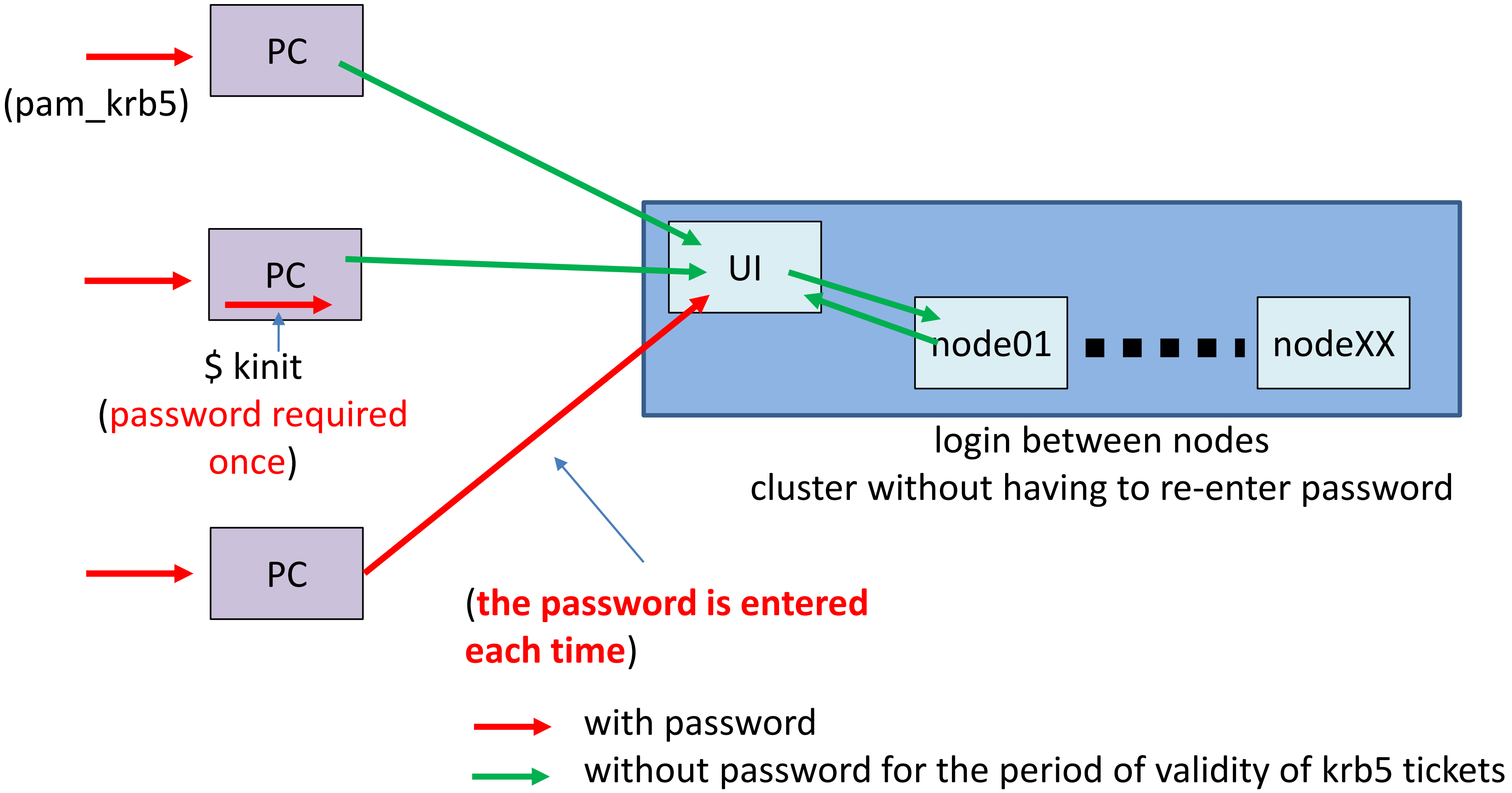
Kerberos is network authentication protocol that allows anyone communicating on an insecure network to securely prove their identity to someone else. Kerberos prevents eavesdropping or repetition of such communications and ensures data integrity. It was created primarily for the client-server model and provides mutual authentication - both the client and the server verify the identity of their counterpart. Kerberos is built on symmetric cryptography and therefore needs a trusted third party. Optionally, it can use asymmetric encryption in certain parts of the authentication process.

Kerberos has **strict time synchronization requirements for clients and servers**. Tickets have a given lifetime, and if the client's time is not synchronized with the server time, authentication will fail. The default setting by MIT requires these times **did not differ by more than 5 minutes**. **NTP (Network Time Protocol)** daemons is used in practice to synchronize the clock.

On WOLF cluster, krb5 tickets from realm META are created during login and can be used for authentication of login to MetaCentrum front-ends, to copy data with the command scp from/to front nodes and for connection of MetaCentrum data storages to WOLF cluster.

wikipedia.org

Workflow



(pam_krb5)

$ kinit

(password required once)

login between nodes

cluster without having to re-enter password

(**the password is entered each time**)

→ with password

→ without password for the period of validity of krb5 tickets

The WOLF cluster behaves according to the first variant.

Commands

kinit	create a new krb5 ticket
klist	list existing krb5 tickets
kdestroy	delete existing krb5 tickets

realm for MetaCentrum

```
[kulhanek@pes ~]$ kinit
Password for kulhanek@META:
[kulhanek@pes ~]$ klist
Ticket cache: FILE:/tmp/krb5cc_1001
Default principal: kulhanek@META

Valid starting       Expires              Service principal
01/30/2016 23:28:30  01/31/2016 23:28:24  krbtgt/META@META
[kulhanek@pes ~]$ kdestroy
[kulhanek@pes ~]$ klist
klist: No credentials cache found (ticket cache
FILE:/tmp/krb5cc_1001)
[kulhanek@pes ~]$
```

kinit

```
$ kinit
```

name of principal is derived from the principal in cache of Kerberos tickets, if this file does not exist, then from **login name and default realm** (META)

```
$ kinit kulhanek
```

entered name plus default realm (META)

```
$ kinit kulhanek@META
```

specified principal is used

If you use a login name in eINFRA space (realm META) different from one on the local machine, you must explicitly specify it as a command argument **kinit**.

On WOLF cluster, you will receive krb5 tickets automatically during login. Therefore, command kinit is used only when renewing expired tickets.

ssh and kerberos

ssh can be set to authenticate user using krb5 tickets (GSSAPIAuthentication) and that krb5 tickets are transmitted to the remote machine (GSSAPIDelegateCredentials). This is the default setting for NCBR, CEITEC MU, and MetaCentrum.

```
[kulhanek@wolf ~]$ kinit
Password for kulhanek@META:
[kulhanek@wolf ~]$ klist
Ticket cache: FILE:/tmp/krb5cc_9703
Default principal: kulhanek@META
Valid starting       Expires              Service principal
02/02/2016 08:13:53  02/03/2016 08:13:49  krbtgt/META@META
[kulhanek@wolf ~]$ ssh kulhanek@skirit.ics.muni.cz
...
...
[kulhanek@skirit ~]$
[kulhanek@skirit ~]$ klist
Credentials cache: FILE:/tmp/krb5cc_18773_GcLXWPTirK
        Principal: kulhanek@META
  Issued                Expires               Principal
Feb  2 08:14:18 2016  Feb  3 08:13:49 2016  krbtgt/META@META
....
```

not repeated during validity of tickets

only stated if you have a different login name

does not require a password

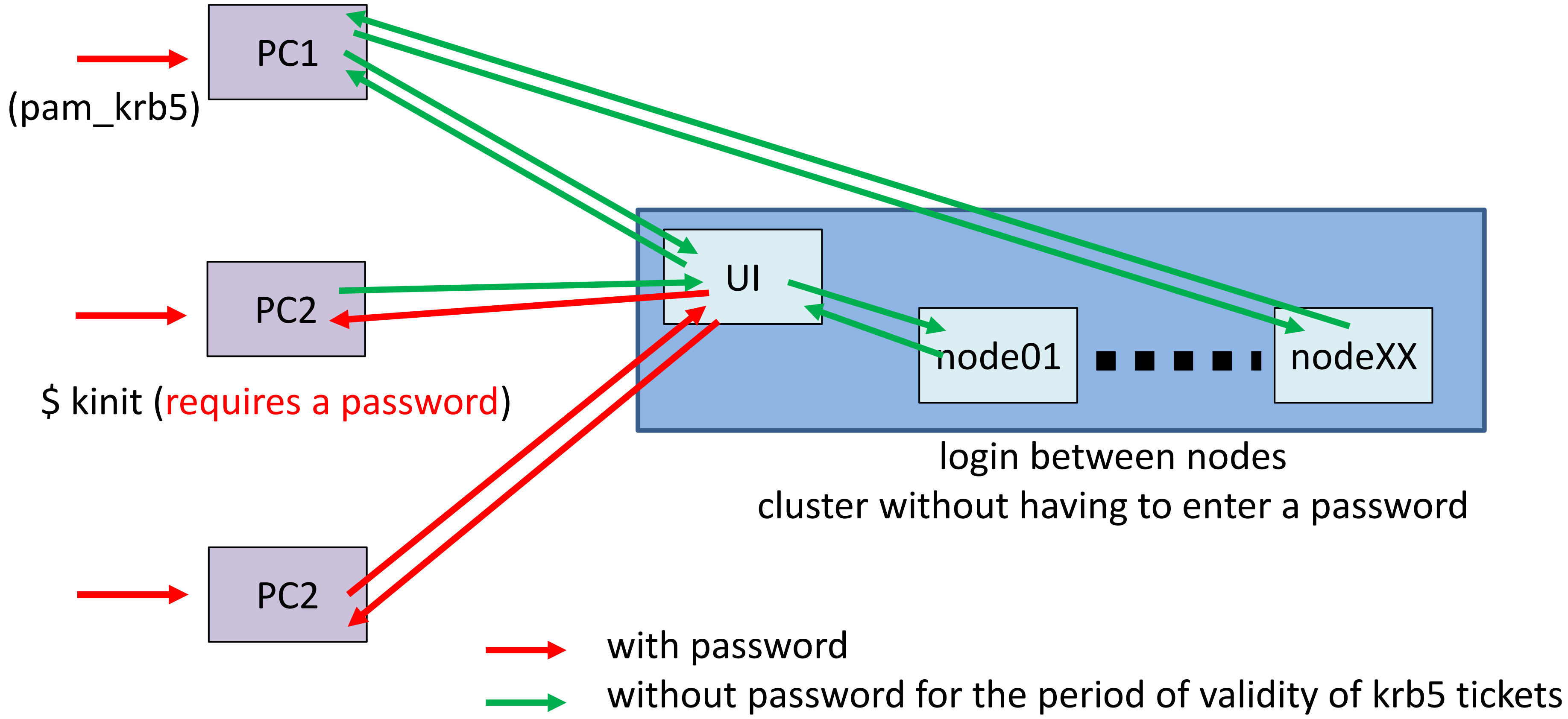
Exercise 2

1. Verify that you have valid krb5 tickets. How long will they be valid?
2. Using command ssh, log in to any front node of MetaCentrum (command ssh must not ask for a password).
3. On the front node, verify that the Kerberos tickets are transferred correctly. How long will they be valid?
4. Log out.
5. Destroy the tickets with the command kdestroy.
6. Try to log in to any MetaCentrum front node again. What are you observing?
7. What is the validity of the created Kerberos tickets on the front node?
8. Can you log in from the front node MetaCentrum on your workstation in a WOLF cluster?

Workflow

PC1 (e.g., workstation in WOLF cluster) is in the same krb5 realm as a cluster.

PC2 is an independent computer.



$ kinit (requires a password)

login between nodes

cluster without having to enter a password

with password

without password for the period of validity of krb5 tickets

Ticket validity/Renewable tickets

The validity of tickets is limited in time, typically a few hours. This is impractical when running long-term jobs. For these purposes it is possible to **create renewable tickets**. Their validity is again limited in time, but during their validity it is possible to request (without entering a password) their renewal. This process can be repeated for a longer period of time, typically several days.

On our clusters and MetaCentrum, kerberos tickets are renewed automatically in jobs run through batch system.

Example:

```
[kulhanek@pes ~]$ kinit -r 5d
Password for kulhanek@META:
[kulhanek@pes ~]$ klist
Ticket cache: FILE:/tmp/krb5cc_1001
Default principal: kulhanek@META

Valid starting       Expires              Service principal
01/31/2016 10:42:22  02/01/2016 10:42:18  krbtgt/META@META
        renew until 02/05/2016 10:42:1
[kulhanek@pes ~]$ kinit -R
```

renews the ticket (choice of capital R), it is possible only during the validity of the existing ticket

Ticket locations (cache)

```
[kulhanek@pes ~]$ klist
Ticket cache: FILE:/tmp/krb5cc_1001
Default principal: kulhanek@META

Valid starting       Expires              Service principal
01/31/2016 11:23:55  02/01/2016 11:23:52  krbtgt/META@META
```

generic name derived from uid, available in all terminals (depending on the OS configuration can also contain a random string)

```
[kulhanek@pes ~]$ ssh onyx.ncbr.muni.cz
...
...
[kulhanek@onyx ~]$ klist
Credentials cache: FILE:/tmp/krb5cc_18773_cOR8E0oV8w
        Principal: kulhanek@META

  Issued                Expires               Principal
Jan 31 11:25:48 2016  Feb  1 11:23:52 2016  krbtgt/META@META
...
```

cache set only for that session **(random string)**

Cache with tickets must not be located on a shared volume (NFS etc.).

Ticket expiration

Once ticket expires, further access to services that require it will be denied. This can lead to visible errors with denied access. **However, some errors are not obvious and finding the cause may not be "easy"**. Typically, this situation occurs for sessions that are open longer than Kerberos ticket are valid and they mainly concern software activated by the module command, which is physically located on the AFS file system (software base MetaCentrum and Infinity).

If something starts to behave strangely (malfunctioning software modules), first check that you have valid Kerberos tickets (klist) and recreate them if necessary (kinit).

AFS file system

Software base of MetaCentrum and Infinity environment resides on the AFS file system. (directory /afs/.ics.muni.cz/software and /afs/.ics.muni.cz/software/ncbr). This FS uses authentication tokens derived from krb5 tickets to control access to files and directories).

Commands:

tokens	list AFS tokens
aklog	create AFS tokens (from valid krb5 tickets), or it is possible to use the command **afslog**
unlog	delete AFS tokens

Kerberos implementation

MIT Kerberos

- kinit **does not restore** AFS tokens

(Ubuntu package: krb5-user)

Heimdal Keberos

- kinit **restores** AFS tokens

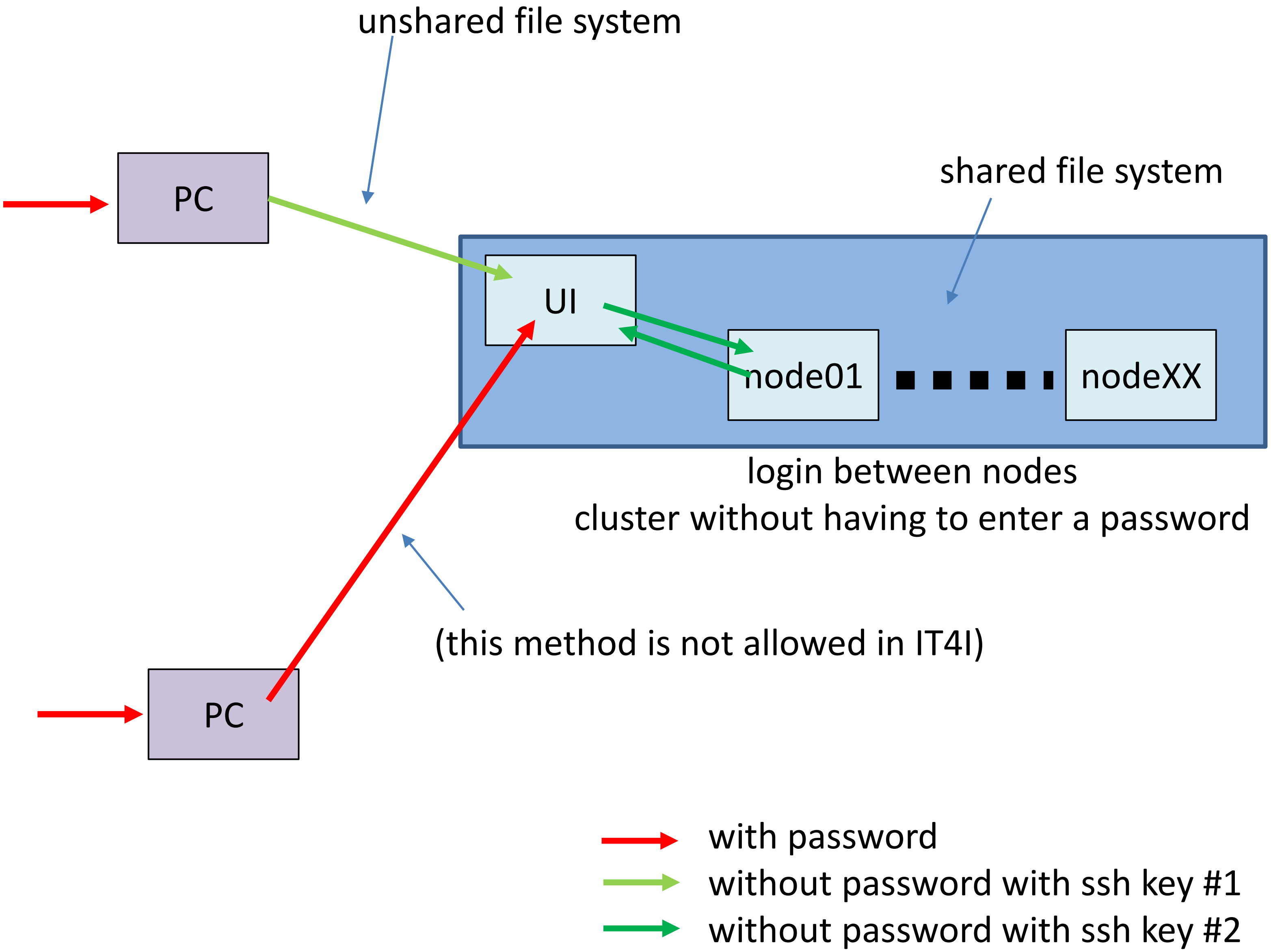
(Ubuntu package: heimdal-clients*)

* default package in MetaCentrum and our clusters

SSH keys

man ssh

Workflow



unshared file system

PC

PC

shared file system

UI

node01

nodeXX

login between nodes

cluster without having to enter a password

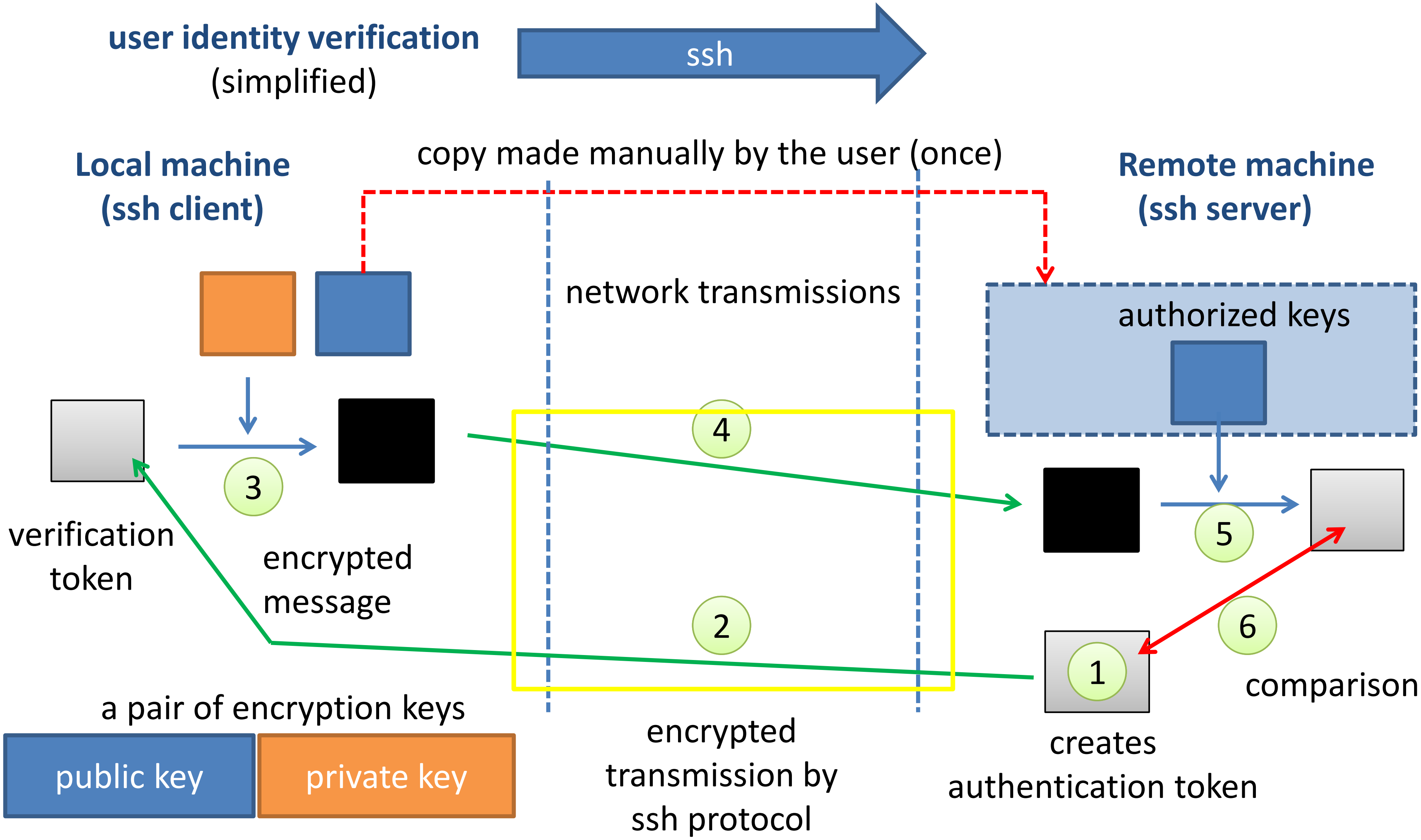
(this method is not allowed in IT4I)

with password

without password with ssh key #1

without password with ssh key #2

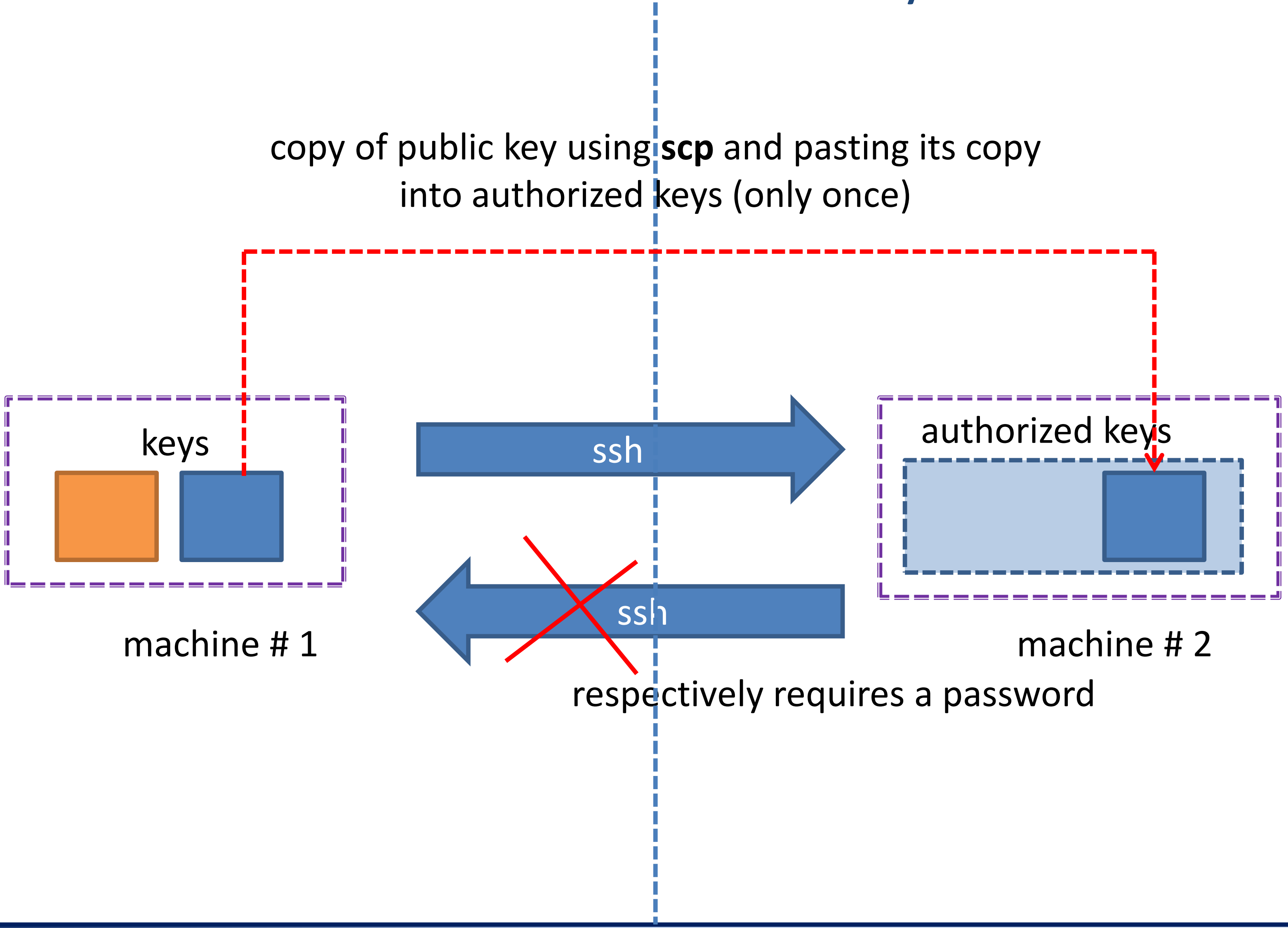
Authorized public ssh key



Anyone who steals a user's private key can log in to the remote machine!

Unshared file system

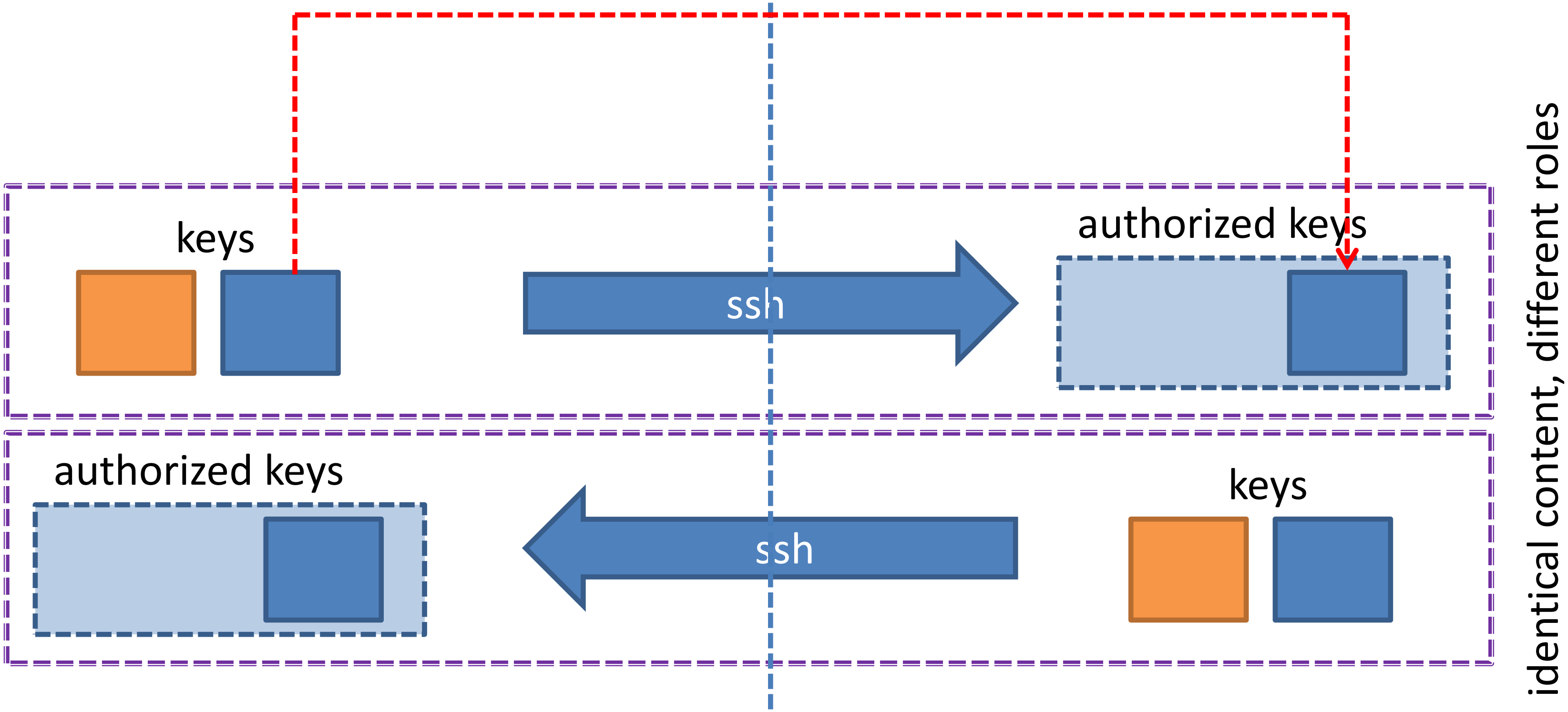
Situation when machines do not have shared home directory:


copy of public key using **scp** and pasting its copy into authorized keys (only once)
keys
ssh
authorized keys
ssh
machine # 1
machine # 2
respectively requires a password

Shared file system

Situations when machines have shared home directory:

both files are on a shared file system
cat id_rsa.pub >> authorized_kyes
can be used (only once)



Create a key pub/priv key

A public and private key pair is created ONLY once on a given machine or group of machines that have a shared directory.

```
[kulhanek@wolf01 ~]$ cd .ssh

[kulhanek@wolf01 .ssh]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kulhanek/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kulhanek/.ssh/id_rsa.
Your public key has been saved in /home/kulhanek/.ssh/id_rsa.pub.
The key fingerprint is:
e9:07:0b:fc:17:23:b3:c5:1a:8a:0c:1a:98:8f:fe:28 kulhanek@wolf01.wolf.inet

[kulhanek@wolf01 .ssh]$ ls -l
-rw------- 1 kulhanek lcc  1675 Mar 21  2012 id_rsa
-rw-r--r-- 1 kulhanek lcc   395 Mar 21  2012 id_rsa.pub
-rw------- 1 kulhanek lcc 13380 Sep  4 15:55 known_hosts
```

Passphrase is not entered!

private key MUST NOT be readable by the group and the world

a list of machine fingerprints that you logged in with using the command ssh

Detailed description: man ssh

Creating of authorized keys - I

shared file system

To insert a public key into the list of authorized keys:

```
[kulhanek@node ~]$ cd .ssh
[kulhanek@node ~]$ cat id_rsa.pub >> .ssh/authorized_keys

[kulhanek@node .ssh]$ ls -l
-rw-r--r-- 1 kulhanek lcc   395 Sep 25  2012 authorized_keys
```

access rights for the file authorized_kyes, **for a group and others - maximum right to read**

File *authorized_keys* can contain multiple public keys, each on a single line.

If logging in with authorized public keys does not work:

- verify the access rights of individual files (letters r, w (possibly x) in the output of ls command)
- if ssh agent is running, delete the keys it manages:

```
$ ssh-add -D
```

- log in again

Detailed Description: man ssh

Creating of authorized keys II

Non-shared file system

The public key must be copied to a remote cluster. Use the **scp** command for remote copying.

Syntax:

[] - can be omitted

```
$ scp [-r] source target
```

Source and target can be a file or a directory. When copying directories, the **–r** option must be used (recursive).

The remote destination or guest is identified by the machine name, separated from the file or directory name by **colon**.

```
[user@]hostname:[path/]file
```

Creating of authorized keys - II

Non-shared file system

To insert a public key into the list of authorized keys:

Obtaining the public key from the machine that will have the role of a client (we want to run the command ssh from):

```
[kulhanek@ui ~]$ scp wolf.ncbr.muni.cz:.ssh/id_rsa.pub wolf.pub
```

colon dot

Entering the public key in the list of authorized keys:

```
[kulhanek@ui ~]$ cat wolf.pub >> .ssh/authorized_keys
[kulhanek@ui ~]$ rm wolf.pub
[kulhanek@ui ~]$ ls –l .ssh
-rw-r--r-- 1 kulhanek lcc   395 Sep 25  2012 authorized_keys
-rw------- 1 kulhanek lcc  1675 Mar 21  2012 id_rsa
-rw-r--r-- 1 kulhanek lcc   395 Mar 21  2012 id_rsa.pub
-rw------- 1 kulhanek lcc 13380 Sep  4 15:55 known_hosts
```

access rights for the file authorized_kyes,
for a group and others - maximum right to read

More detailed description: man ssh

Exercise 3

1. Set up your virtual machine instance with Ubuntu so that you can log in with ssh keys from the host machine.
2. Can you log in to the virtual machine without a password from wolf01? Explain the behavior.

Pros and cons

Advantages:

- the password does not have to be entered every time
- safer use of commands ssh and scp in scripts
- work speed up

Disadvantages:

- in case of compromise of one computer, all computers with mutually authorized public keys are compromised

- **Do not use SSH keys to log in to MetaCentrum or on NCBR or CEITEC MU clusters. This way, Kerberos tickets will not be created, without which the environment of these clusters is unusable!!!**

Installing Kerberos for realm META

using OS packages Ubuntu 18.04 LTS

Installation via packages

1) Activation of **public repository of NCBR packages**. The procedure is given at **https://einfra.ncbr.muni.cz** in the "User Support" section >> “Ubuntu Packages” and CEITEC MU/NCBR PUBLIC repository, select Ubuntu 18.04 LTS. The repository is **activated only once.**

2) Kerberos support for Metacentrum (configuration and heimdal clients):

```
$ sudo apt-get install ncbr-krb5-einfra
```

3) Kerberos support in ssh (GSSAPIAuthentication and GSSAPIDelegateCredentials)

```
$ sudo apt-get install ncbr-ssh-client-config
```

4) pam_krb5 configuration (obtaining krb5 ticket at first login)

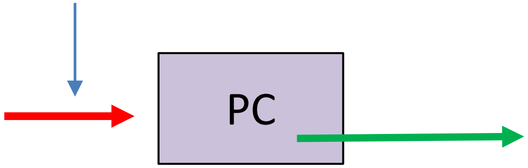
```
$ sudo apt-get install ncbr-personal-authc-einfra
```

WARNING: NCBR packages are authoritative. It is no longer possible to change the configuration that the packages set (made changes will expire when they are updated).

Workflow (pam_krb5)

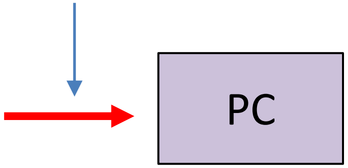
Computer is connected to the network:

login with password (**eINFRA** password)

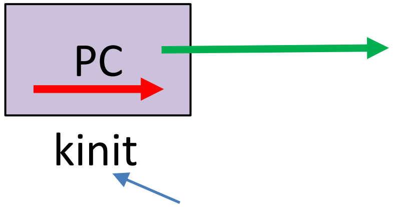


Computer is not connected to the network:

login with password (**local** password)



after connecting the computer to the network:



login with password (**eINFRA** password)

Recommendation:

- Suitable for laptops or computers outside eINFRA**.**
- **Local** password must be **set before** installing the package ncbr-personal-authc-einfra and (i.e., when installing the computer or using the passwd).
- It is advisable to have **same local and eINFRA** password.

Exercise 4

1. Install support for creating Kerberos tickets in realm META of MetaCentrum virtual organization in your installation of Ubuntu server (points 1 and 2 of the previous instructions).
2. Verify that you can create Kerberos tickets with the command kinit and klist.
3. Adjust the command settings of ssh for use of Kerberos tickets (point 3 of the previous instructions).
4. Verify that you can log in to any front node of MetaCentrum or a WOLF cluster without using a password.
5. Verify that the Kerberos tickets are transmitted to the front node.
6. Enable creating a krb5 ticket during initial login (point 4 of the previous instructions).
7. With the command klist, verify that you have created Kerberos tickets after login.
8. Log in to the virtual machine using ssh from the host machine. Is password required? Why don't you have krb5 tickets created?
9. Download the ncbr-krb5-einfra package and explore its contents with *mc*.

Installing Kerberos for realm META

manual installation

Installation Kerbera (clients)

The client part of Kerberos can be installed on any computer that is connected to the Internet. The procedure below is tested in OS Ubuntu 16.04 LTS.

1) Installing NTP (Network Time Protocol daemon and utility programs) - necessary for correct time setting (select default values during configuration)

```
$ sudo apt-get install ntp
```

2) Installing Kerberos client utilities (select default values during configuration)

```
$ sudo apt-get install krb5-user
```

or heimdal clients

3) Obtaining the krb5.conf configuration file for MetaCentrum. You can copy the file from any front node of MetaCenter or any WOLF cluster node. Its location is /etc/krb5.conf

4) Copy the file (as a super user) to /etc/krb5.conf.META and set its rights to 0666 (read-only).

5) Create a symbolic link:

```
$ sudo unlink /etc/krb5.conf
$ sudo ln -s /etc/krb5.conf.META /etc/krb5.conf
```

Integration Kerbera to ssh

To use Kerberos tickets for remote login to MetaCenter nodes, it must be enabled in the configuration of **ssh** command (change also applies to command **scp**). The change can be made for all users by changing **/etc/ssh/ssh_config** file or by changing/creating a file **~/.ssh/config** for a specific user.

```
Host *
#   ForwardAgent no
#   ForwardX11 no
#   ForwardX11Trusted yes
...
    SendEnv LANG LC_*
    HashKnownHosts no
    GSSAPIAuthentication yes
    GSSAPIDelegateCredentials yes
    ForwardX11 yes
```

do not change the default commented (#) values
place changes at the end

enable authentication using Kerberos ticket, this form of authentication must be supported by the remote machine

transfers the ticket(s) to the remote machine

allows you to use machine name completion for the command ssh and scp using TAB

automatically exports the X11 display (equivalent to the -X)

ssh and kerberos

if you have **different login in eINFRA** than default one on the machine, you must explicitly specify it when using **ssh** command. The second option is to change the configuration of ssh using a file **~/.ssh/config**, man ssh_config for more, statement **User**. When using the second option, it is necessary to set at least **GSSAPIAuthentication** and **GSSAPIDelegateCredentials** (see above).

~/.ssh/config

```
Host skirit.ics.muni.cz tarkil.cesnet.cz
    User xstepan3
    SendEnv LANG LC_*
    HashKnownHosts no
    GSSAPIAuthentication yes
    GSSAPIDelegateCredentials yes
    ForwardX11 yes
```

a space-separated list of front node names

login name to MetaCentrum

access rights for the file **~/.ssh/config**,
for a group and others - maximum read only