AURIA CLINICAL INFORMATICS
AURIA TIETOPALVELU

HOSPITAL DISTRICT OF
SOUTHWEST FINLAND

The Clinical Data Refinery

Management and Administration of the Analytics Environment

Editors:
Anna Hammais, Juha Varjonen and Arho Virkki

AURIA CLINICAL
INFORMATICS

AURIA CLINICAL INFORMATICS
AURIA TIETOPALVELU

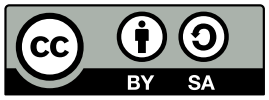
The Clinical Data Refinery

Management and Administration of the Analytics Environment

Editors:
Anna Hammais
Juha Varjonen
Arho Virkki
together with
The Auria Informatics Team

Sponsors:
Päivi Rautava, Turku Clinical Reserch Centre
Tarja Laitinen, Former Head of the Department of Pulmonary Medicine
The Finnish Innovation Fund Sitra

Contact:
email: atp@tyks.fi
mail: Auria Clinical Informatics
Turku University Hospital
Building 11 B, 2nd floor, room 206
PO Box 52, 20521 Turku



Auria Clinical Informatics Documentation

Welcome to ACI Wiki!

These pages provide technical documentation about the data analysis platform setup and holds copies of some essential binary tools. The Wiki is stored physically as a bare Git repository at `aku@parrot.intra.net:/opt/git/KTPWiki.git`.

The documentation is divided into several main topics

1. Process Flow
2. Data Sources
3. Data Semantics
4. Data Share Log
5. Code Review Process
6. In-House Server Environment
7. Version Control System
8. Software Installation, Administration and Backups
9. SQL, R, Pentaho Data Integration
10. PostreSQL; Hadoop; Flask/Python; Neo4j; SAML
11. Data Visualization
12. External Research Environment
13. Software Summary
14. Data Platform Terminology (in Finnish)

How to Browse and Add Content

The [All] button on top of the page opens a hiearchical view. The complete documentation tree can be accessed at http://parrot.intra.net/fileview.

See Daring Fireball's Markdown Tutorial or just click 'Edit' to see how the pages are written.

Book

Open http://parrot.intra.net/book/atp_book.pdf for a printable copy.

ACI Administrators can re-generate the book directly form Wiki Markdown sources with pandoc by running *getbook.sh* at the Git document root.

Data Analysis and ETL Process

Document author: arho.virkki@tyks.fi
Contributors: anna.hammais@tyks.fi, juhana.valo@2m-it.fi, katja.kanerva-leppanen@2m-it.fi

Team Overview

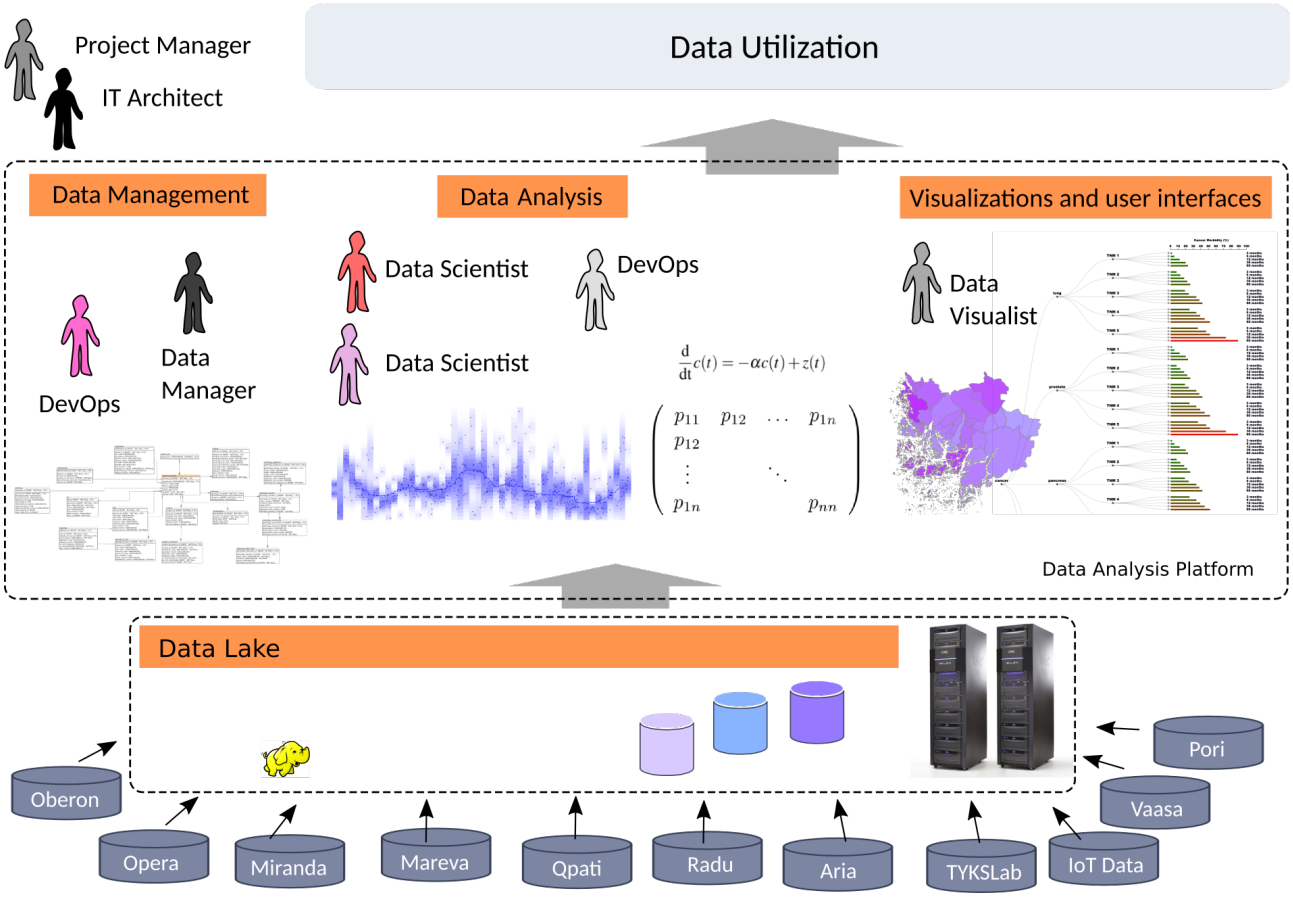


Figure [PDF, SVG]. Process Overview.

Data Analysis Infrastructure

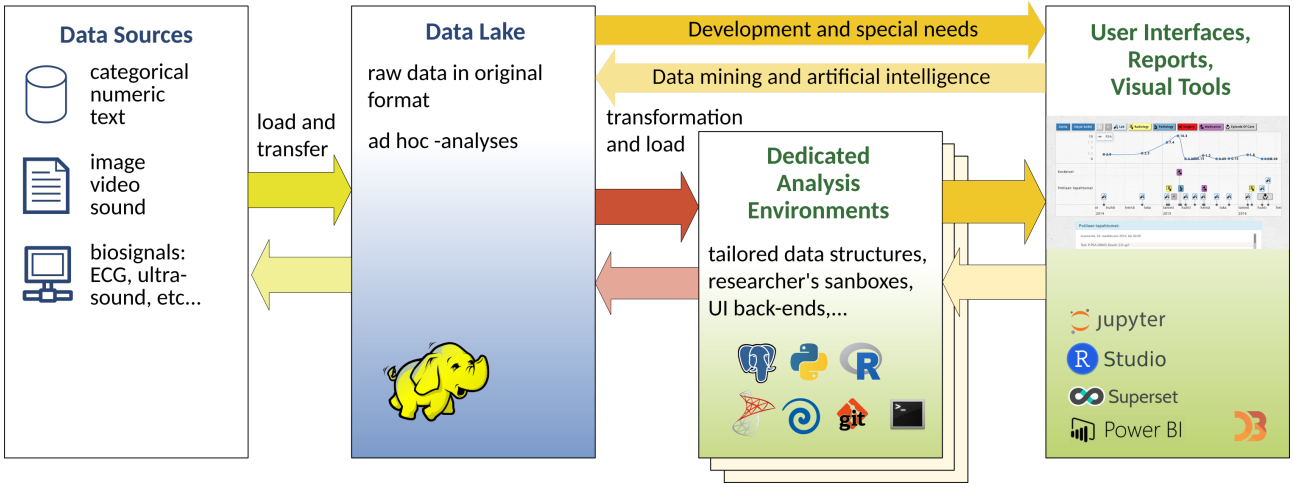


Figure [PDF, SVG]. Data Management and Analysis Stack.

Data Flow

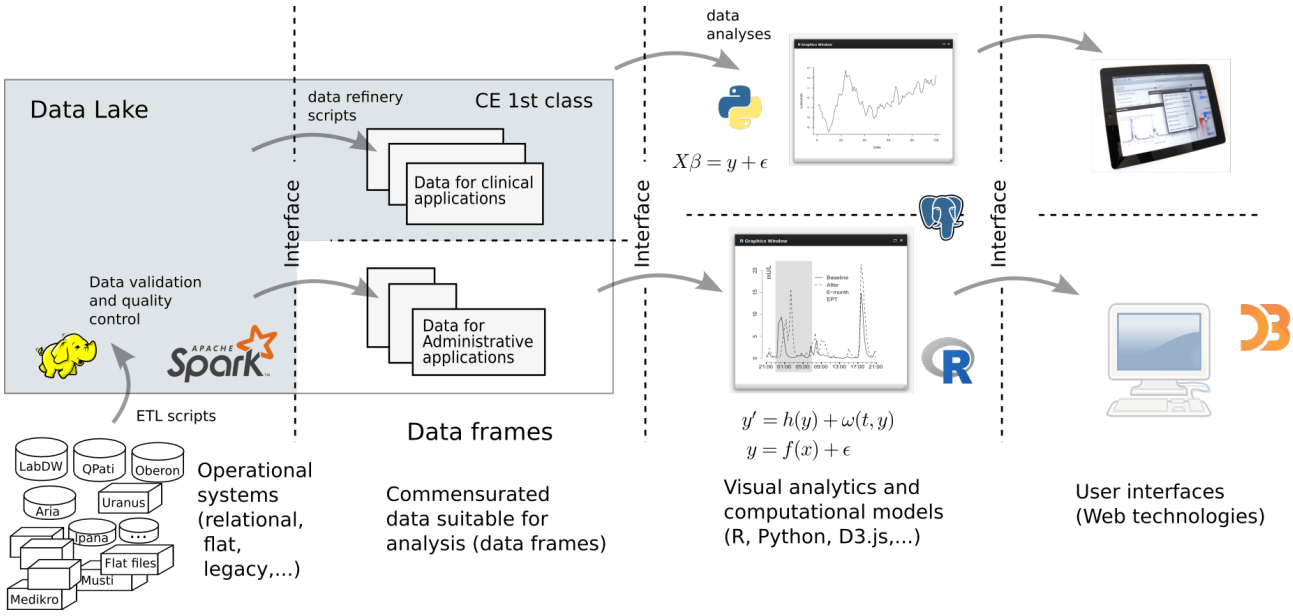


Figure [PDF, SVG]. Data Analysis Workflow.

Research Process Steps

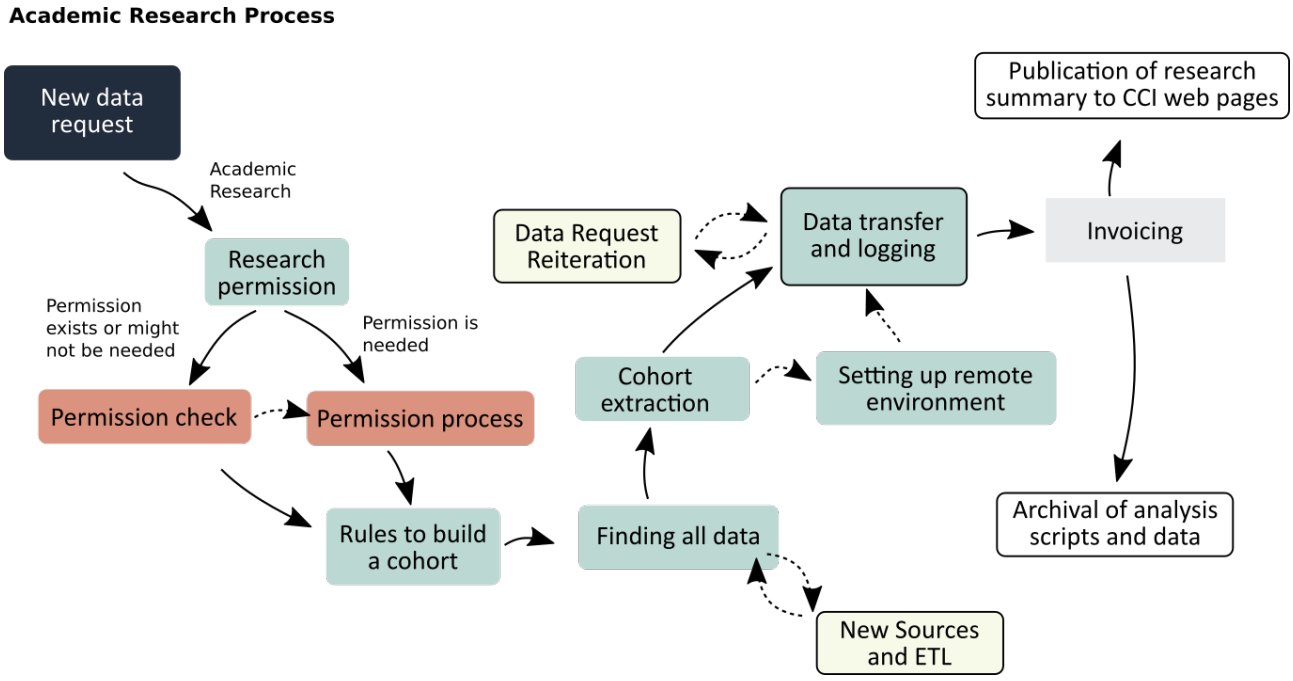


Figure [PDF, SVG]. Academic research process.

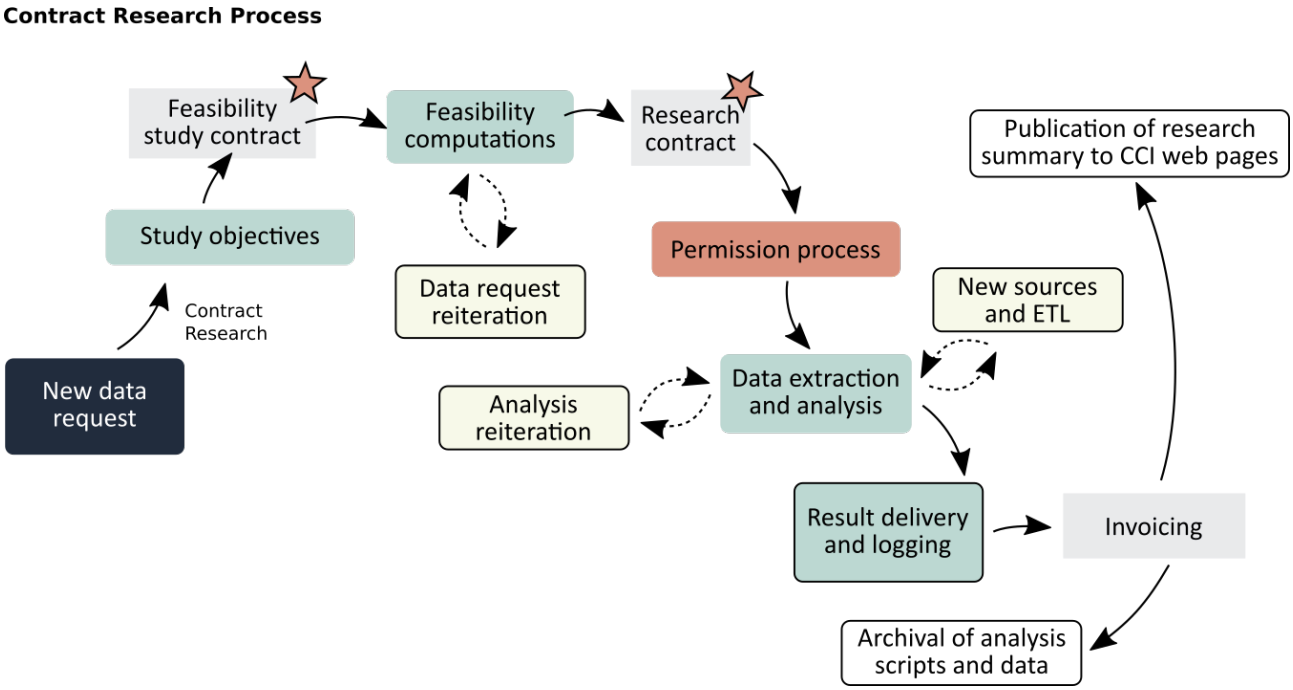


Figure [PDF, SVG]. Contract research process.

New Data Sources and Data Harmonization

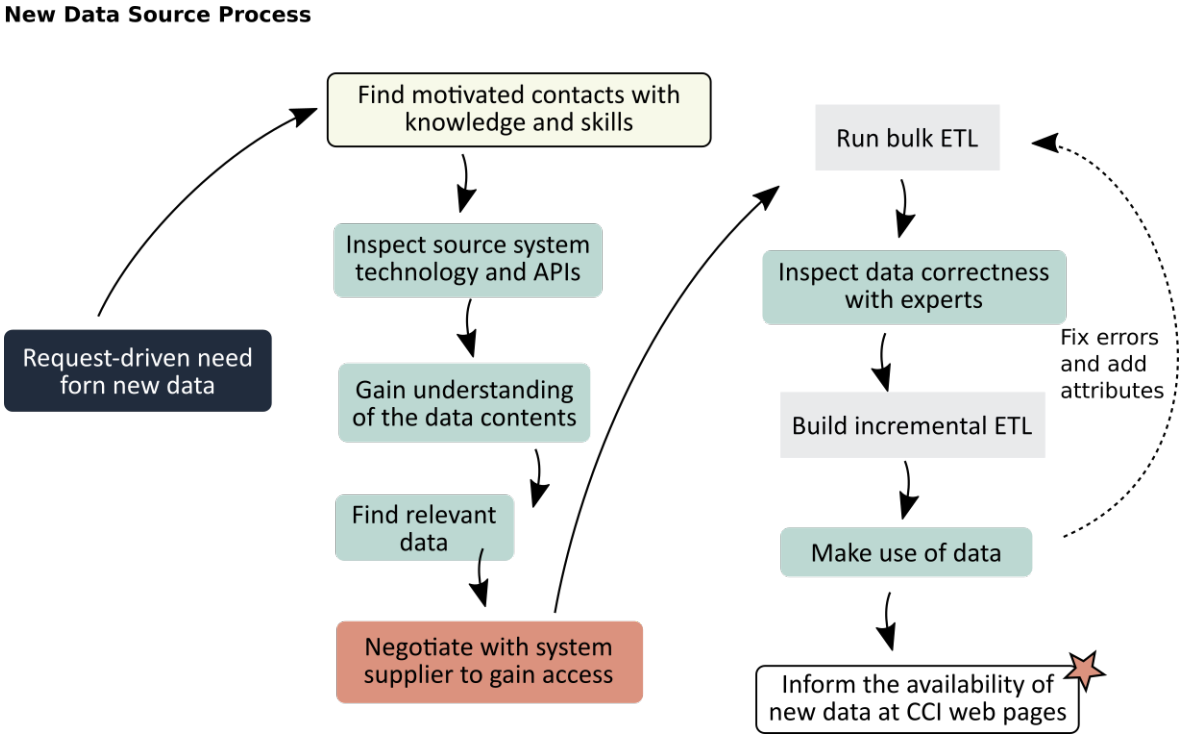


Figure [PDF, SVG]. New data source process.

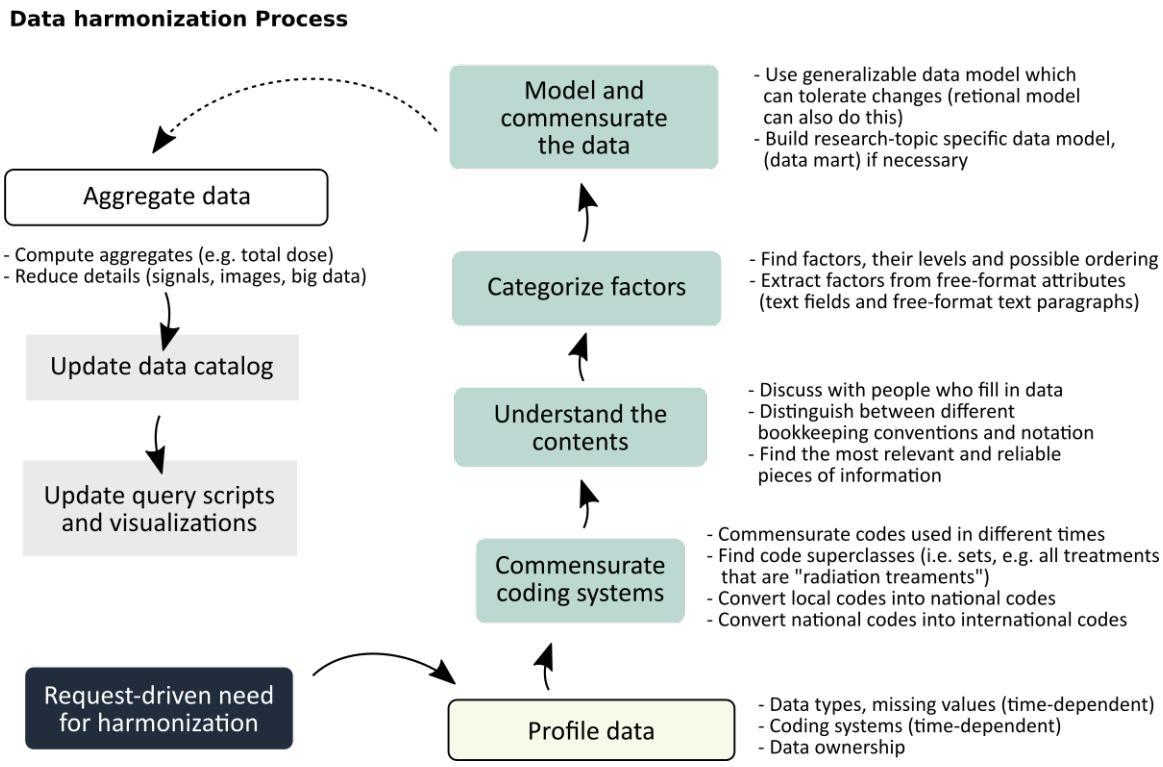


Figure [PDF, SVG]. Data harmonization process.

ETL Steps

1. Data extraction from source
2. File upload
3. Format conversion
4. Type conversion
5. Data integration
6. Semantic unification

ETL Script Repository

The ETL scripts are saved to the Git-repository `aku@gitbox:/opt/git/ETL.git` and a working copy under

```
bluebird.intra.net:~/ETL/
```

and is used with Pentaho Kettle in the data integration phase. The automated etl scripts are run by the *aku* user. For details, see *crontab -e* as *aku* at *bluebird*.

Software and Core Review Process

Document Authors: arho.virkki@tyks.fi, juha-matti.varjonen@tyks.fi

Best practices of software and code review

Software / application review template

```
Review Date: <yyyy-mm-dd>
Software / application (file)name: <name>
Software / application version number: <version>
Responsible developer / designer: <name>
Responsible reviewer: <name>

[Yes/No/Under work]: is the documentation of the software or application saved in the KTP network
    share   (\\atdbw61\KTP\Ohjeet).
- Rejection criterion: no documentation and / or good reason why there is
only a subset of documentation or no official documentation at all.
- Comments: TBA

[Yes/No/Under work]: is the software or application information documented in the ktp wiki
    (http://parrot.intra.net/Home)?
- Rejection criterion: no documentation on the ktp-wiki and / or good reason why the software or
    application does not have a
documentation on the ktp-wiki.
- Comments: TBA

[Yes/No/Under work]: is the software or application  stored in the KTP version control system (Git)?
- Rejection criterion: The software or application can not be found in the version control.
- Comments: TBA

[Yes/No/Under work]: does the documentation have information on the software or application's
    production, testing, and
development environments?
- Rejection criterion: no basic information about the application's environments.
- Comments: TBA

[Yes/No/Under work]:  do reviewer know which tools have the software or application been developed
    and with which tools can the
application be further developed?
- Rejection criterion: maintenance and / or upgrading of the software or application required tools
    not mentioned
in the documentation.
- Comments: TBA

[Yes/No/Under work]: are there any examples of the use of modules / classes?
- Rejection criterion: no reason for rejection?
- Comments: TBA

[Yes/No/Under work]: is there any architecture or class diagrams etc. available?
- Rejection criterion: no reason for rejection?
- Comments: TBA
```

Code review template

```
Review Date: <yyyy-mm-dd>
Code (file)name: <name>
Code version number: <version>
Responsible developer / designer: <name>
Responsible reviewer: <name>

[Yes/No/Under work]: is the reviewer able to test the software or application while performing the
    code review?
- Rejection criterion: the software / or application can not be tried and / or tested in practice.
- Comments: TBA

[Yes/No/Under work]: see how easy is to get "big picture" from the version control of the code and
    folder structure etc.?
- Rejection criterion: the whole code is very difficult and / or impossible to get an overview
    without the author's
verbal instruction.
```

```
- Comments: TBA

[Yes/No/Under work]: try to find out is the any mistakes and misunderstandings in the code and that
    the
code is properly commented?
- Rejection criterion: there are large logical errors in the code and not commented.
- Comments: TBA

[Yes/No/Under work]: the variables, functions, methods, classes and modules should be named
    descriptively.
- Rejection criterion: in order to understand the code, reviewer must consult the developer in
    critical manner because
of poor naming
- Comments: TBA

[Yes/No/Under work]:  the method / function should be less than 100 lines long
- Rejection criterion: Multiple and / or non-reasoned over 100 lines long methods or functions if
    not  clearly stated
otherwise why.
- Comments: TBA
```

Server Environment

Document author: arho.virkki@tyks.fi

Overview

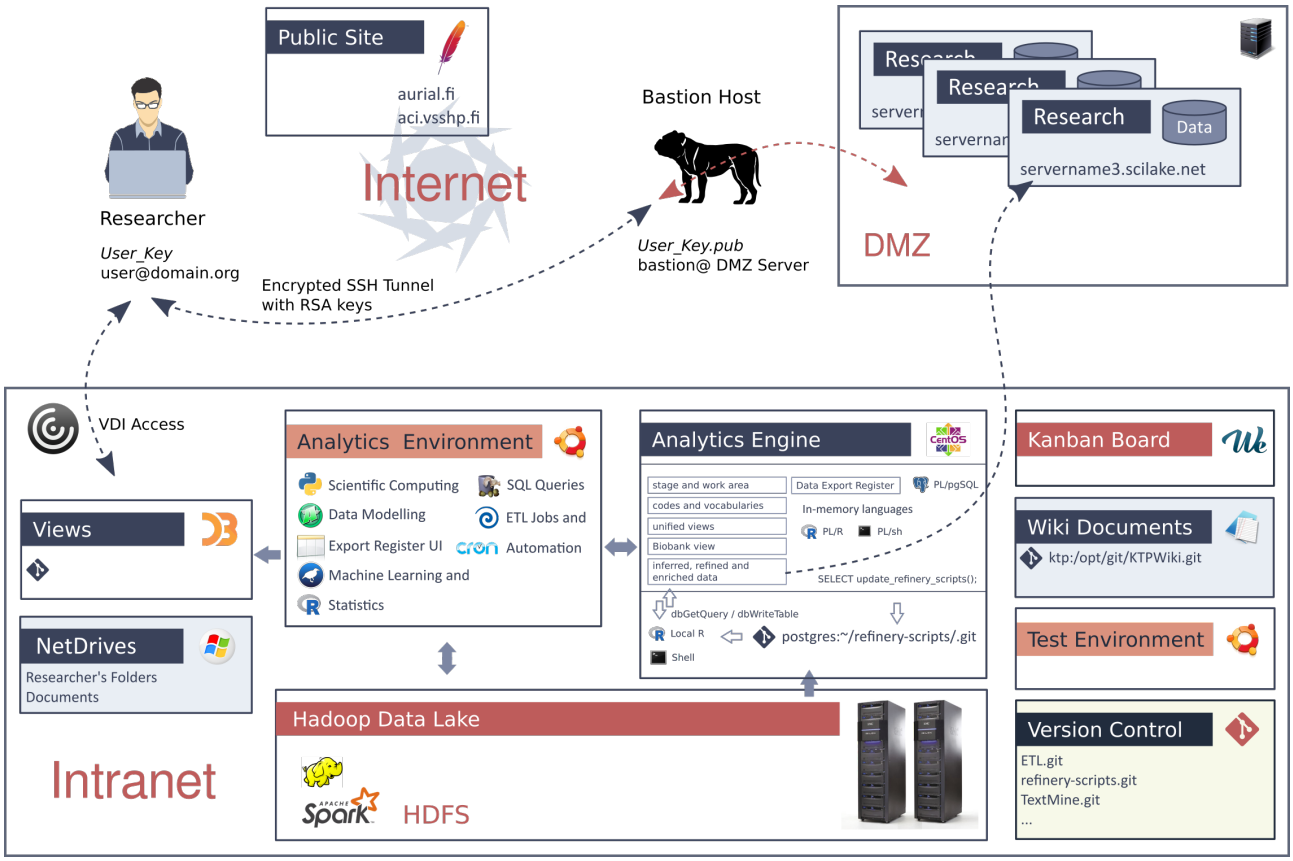


Figure. Server Architecture (pdf, svg).

Physical Servers

- supernova.public.net (Dell PowerEdge)
- moondust.intra.net (Dell PowerEdge)
- Notes on VM setup

Public Web Server

- Setup and Configuration

Intranet Virtual Servers

- Intranet Servers

DMZ Server supernova.public.net

Author: arho.virkki@tyks.fi

Dell PowerEdge server

Machine specifications

```
Dell PowerEdge R730xd

1 Intel Xeon E5-2695 v3 2.3GHz,35M Cache
9.60GT/s QPI,Turbo,HT,14C/28T Chassis with up to 12, 3.5" Hard Drives
2, 2.5" Flex Bay Hard Drives
16kpl 16GB RDIMM, 2133MT/s, Dual Rank, x4 Data Width
1 Upgrade to Two Intel Xeon E5-2695 v3 2.3GHz,35M Cache
9.60GT/s QPI,Turbo,HT,14C/28T
2kpl Standard Heatsink for PowerEdge R730/R730xd
1kpl iDRAC8 Enterprise, integrated Dell Remote Access Controller, Enterprise
2kpl 400GB Solid State Drive SATA Mix Use MLC 6Gpbs 2.5in Flex Bay Drive,13G
6kpl 6TB 7.2K RPM SATA6 6Gbps 512e 3.5in Hot-plug Hard Drive
1kpl PERC H730P Integrated RAID Controller, 2GB Cache
1kpl Dual, Hot-plug, Redundant Power Supply (1+1), 750W
1kpl  Intel X520 DP 10Gb DA/SFP+ + I350 DP 1Gb Network Daughter Card
1kpl Intel Ethernet I350 DP 1Gb Server Adapter
1kpl PowerEdge R730/R730xd Motherboard
Base Warranty
3Yr Basic Warranty - Next Business Day - Minimum Warranty
5Yr ProSupport Plus and 4hr Mission Critical
```

Network configuration

The server is located at Medbit Demilitarized (DMZ) Network Zone, and used to set up KVM virtual machines as external research environments. It can be accessed with ssh and X2Go, but only with a valid ssh private key. Contact *Jukka Palko at 2M-IT Ltd* or *Arho Virkki at TYKS* if you are willing to gain access to the underlying machine.

Setting Up a Bastion Proxy User

A Bastion Host is a computer which is atteched directly to Internet and protects the *Intranet* machines from being exposed directly to the Internet. Bastion hosts are usually implemented as jump nodes, i.e. machines to log in during the process of reaching the final destination. For an introduction, see for example http://blog.scottlowe.org/2015/11/21/using-ssh-bastion-host/

If the Bastion Host does also other tasks, we can increase its security by disabling password authentication (which is a good policy in all cases) and by introducing a special user which can only forward the incoming connections. In the following, we set up such a jailed 'bastion' account. For details about ssh jails, see http://allanfeid.com/content/creating-chroot-jail-ssh-access.

Create a Bastion User

```
sudo adduser bastion
sudo rm -fr /home/bastion
```

Updated 2017-12-29: Even though ordinary ssh connections work perfectly even when ordinary logins are completely disabled, new versions of X2Go (in particular >= 4.1.1.0) fail to make connection in such case. In the following, we will set up a jailed account that can log in, but has no tools (e.g. no 'ls') to examine and interfere with the underlying operating system.

Next, change bastion home into '/'

```
sudo vim /etc/passwd

# Find the bastion line and change it into e.g.
bastion:x:2002:2002::/:/bin/bash
```

Create an SSH Jail

```
sudo mkdir -p /etc/bastion-jail/{lib64,bin,etc}
sudo cp /bin/bash /etc/bastion-jail/bin/
sudo ln /etc/hosts /etc/bastion-jail/etc/hosts # A hard link
```

Observe the hard link in the last command. Soft links would not work, since bastion does not have access outside of its jail.

To run the *bash* shell we also need some libraries. One way of inspecting which libraries are at least necessary for any program is to use *ldd*, e.g. `ldd /bin/bash`. Unfortunately this does not always reveal all dependencies. The following list of libraries was found by first copying all libraries under `lib64` and using *lsof* to indicate those that are actually used by the session.

```
sudo cp /lib64/{ld-linux-x86-64.so.2,libc.so.6,libdl.so.2,\
  libnsl.so.1,libnss_compat.so.2,libnss_files.so.2,\
  libnss_nis.so.2,libtinfo.so.5} /etc/bastion-jail/lib64/
```

In addition to the libraries, let's add some device files

```
sudo mknod -m 666 /etc/bastion-jail/dev/null c 1 3
sudo mknod -m 666 /etc/bastion-jail/dev/tty c 5 0
sudo mknod -m 666 /ect/bastion-jail/dev/zero c 1 5
sudo mknod -m 666 /etc/bastion-jail/dev/random c 1 8
```

Then, edit the secure shell configuration file

```
sudo vim /etc/ssh/sshd_config

# Add these lines at the end of the file
Match group bastion
      ChrootDirectory /etc/bastion-jail/
      X11Forwarding yes
      AllowTcpForwarding yes
      PasswordAuthentication no
      AuthorizedKeysFile /etc/ssh-pool/bastion_keys
```

The authorized_keys file will be put under '/etc/ssh-pool/':

```
sudo mkdir /etc/ssh-pool/
sudo touch /etc/ssh-pool/bastion_keys
sudo chown bastion: /etc/ssh-pool/bastion_keys
sudo chmod og-rwx /etc/ssh-pool/bastion_keys
```

Finally, restart the sshd service

```
sudo systemctl restart sshd
```

(This is safe, even when logged in remotely, given that the ssh configuration is valid.)

Create a Public Key for the Client

Next, we need the public key of the machine which is used to connect the server. Under Linux and Apple OS X, the public key can be generated in Terminal with the command

```
ssh-keygen
```

Now, copy your public RSA key to the remote machine and append its contents to '/etc/ssh-pool/bastion_keys'. For example,

```
scp ~/.ssh/id_rsa.pub aku@supernova.public.net:

ssh aku@supernova.public.net
sudo sh -c "cat id_rsa.pub >> /etc/ssh-pool/bastion_keys"
```

The following friendly error message indicates a valid key:

```
ssh bastion@supernova.public.net

Last login: Wed May 25 14:57:27 2016 from dsl-tkubrasgw1-50dd3a-111.dhcp.inet.fi
This account is currently not available.
Connection to supernova.public.net closed.
```

Open a Connection (from Linux and Apple OS X)

Once the authentication key is set to the intermediate host, it can be used as a jump node. The proxy command can be given as a command line argument to ssh, such as

```
ssh -o ProxyCommand="ssh -W %h:%p user@jumpnode" target_machine
```

For example,

```
ssh -o ProxyCommand="ssh -W %h:%p bastion@supernova.public.net" woodpecker.intra.net
```

The other option is put the proxy command in `.ssh/config`, so that it does not have to be given on the command line.

```
Host woodpecker.intra.net
  IdentityFile ~/path/to/correct/private_key
  ProxyCommand ssh bastion@supernova.public.net -W %h:%p
```

With the above directives set, the following commands work as if there was no intermediate step at all:

```
# Use X GUI applications at the remote machine
ssh -X woodpecker.intra.net

# Copy files between machines
scp my_file.txt woodpecker.intra.net:~/Desktop/
```

Generating SSH RSA Key Pair

The following command does not requite input (and can be used in scripts)

```
ssh-keygen -f DMZ/StudyKeys/PPK_tutkimus -C "Privat-Public Key Study" -N "" -q
```

Ssh-keygen produces two files: the public one with *.pub* -extension should be appended to `/etc/ssh-pool/bastion_keys` on the server side, whereas the private file (with no extension) should be kept in safe, and given to those who need the access.

Custom identity files (=private keys) can be given in command line. Example:

```
ssh -i ~/path/to/Private_Key bastion@supernova.public.net
```

Setting Up DMZ Computing Environment

Document Author: arho.virkki@tyks.fi

Virtual Host IP Policy

The **static ip range** for the ready-made templates is

```
192.168.122.10  template10  template10.scilake.net
192.168.122.20  template20  template20.scilake.net
...
```

and for the research virtual machines

```
192.168.122.100 srv100   srv100.scilake.net
192.168.122.110 srv110   srv110.scilake.net
...
```

with

```
Gateway:    192.168.122.1
Nameserver: 192.168.122.1
Netmask:    255.255.255.0 (/24)
```

Giving Access to the Virtual Machines

SSH public key

In case that the researcher does not have a public key (or know what it is), the key can be generated in the KTP office with

```
ssh-keygen -f the_key_file_name -C "A comment for the key" -N "" -q
```

This key does not use passphrase (and can be used in scripts). The `ssh-keygen` produces two files: the public one with the *.pub* -extension which should be appended to `/etc/ssh-pool/bastion_keys` on the server side, whereas the private file (with no extension) should be kept in safe, and given to those who need the access. **The private key must never be sent via email!**.

```
sudo sh -c "cat the_rsa_id.pub >> /etc/ssh-pool/bastion_keys"
```

Note: Always use double “>>” to add the new keys after the existing ones.

Accessing the VM

The virtual machine (e.g. 192.168.122.20) can be accessed with ssh by eiher giving the proxy command explicitly,

```
ssh -o "ProxyCommand ssh bastion@supernova.public.net -W %h:%p" aku@192.168.122.20
```

or by putting the following lines to `.ssh/config`:

```
Host 192.168.122.20
  IdentityFile ~/path/to/the/private_key
  ProxyCommand ssh bastion@supernova.public.net -W %h:%p
```

The X2Go access is configured the same way.

KTP Server Configuration

Document Author: arho.virkki@tyks.fi

Dell PowerEdge Server

Machine specifications: moondust_specs.pdf

Controlling the Virtual Machines

Virtual machines can be controlled both with CLI and GUI tools, *virsh* and *virt-manager*. The simplest way of connecting from Linux of OS X is to use ssh with X forwarding.

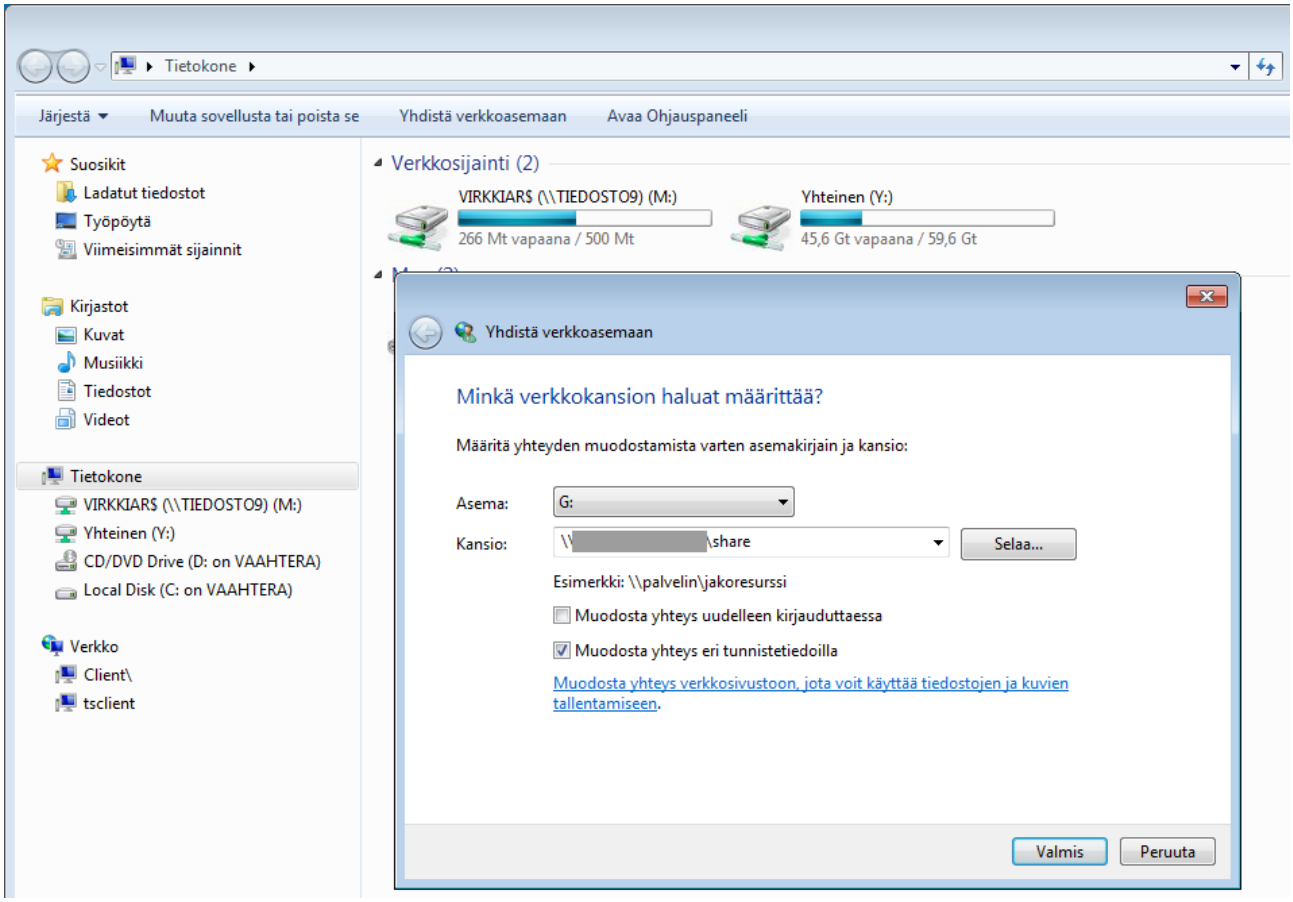
For example, to open *virt-manager*, issue

```
ssh -X aku@moondust.intra.net    # Tunnel X through ssh
sudo su                          # Gain root privileges
xauth merge .Xauthority          # Gain authority to use X
virt-manager                     # Start KVM admin GUI
```

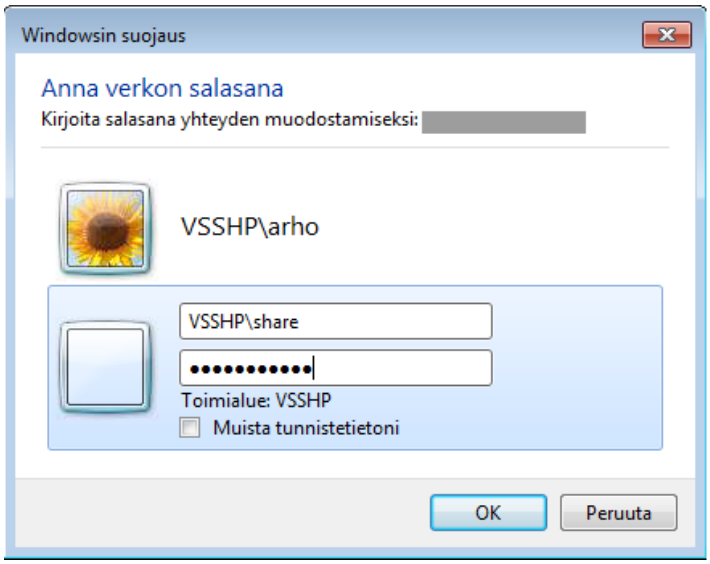
Samba Shares

Accessing the Shares

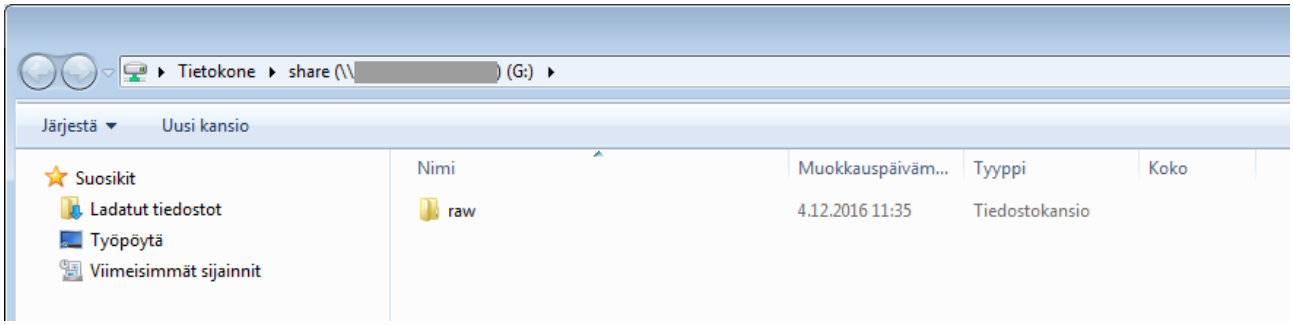
On windows, open Computer, connect to network drive, and provide the correct credentials.



Step 1. Connect to a network drive.



Step 2. Provide right credentials.



Step 3. Use the share.

On Linux, put e.g. the following text (in one line) to */etc/fstab*

```
//moondust.intra.net/share/raw /mnt/raw cifs credentials=/root/sharecredentials,
  iocharset=utf8,sec=ntlm,uid=1000,gid=1000 0 0
```

where the *sharecredentials* file should look like

```
username=share
password=<password here>
```

Configuration

Samba shares were originally known as 'Windows Shares', but they work as well for sharing disk space for the Linux virtual machines and Windows guest. For details, see the official RHEL 7 Samba documentation.

First, we add a dedicated user for the share

```
sudo adduser share
sudo passwd share
```

Then, crate the shared directory and change the 'share' user's home to point to it.

```
sudo mkdir -p /var/share/raw
sudo chown share: /var/share* -R

sudo vim /etc/passwd

# The passwd file should have a line like
share:x:2002:2002::/var/share:/bin/bash
```

Install samba service and edit the configuration file

```
sudo yum install samba
sudo vim /etc/samba/smb.conf
```

to read

```
workgroup = VSSHP
```

Finally, assingn a password to the 'share' user. (For simplicity, it matches ktp user's password on virtual machines).

```
sudo smbpasswd -a share
```

Extra Tools on Gradient

Command line utilities

```
sudo yum install bash-completion nano screen git p7zip tree
```

GUI tools

```
sudo yum install xclock gnome-terminal gparted gedit x2goclient
```

Install parallel version of *bzip2*

```
sudo yum group install "Development Tools"
sudo yum install bzip2-devel

wget https://launchpad.net/pbzip2/1.1/1.1.12/+download/pbzip2-1.1.12.tar.gz
tar xvzf pbzip2-1.1.12.tar.gz
cd pbzip2-1.1.12/
make
sudo make install

man pbzip2
```

Apache Superset, Gunicorn & Nginx

Document Author: per-erik.gustafsson@tyks.fi, arho.virkki@tyks.fi

What is Superset, Anyway?

For as short introduction, see http://superset.apache.org/, http://nginx.org/ and http://docs.gunicorn.org/en/stable/index.html for a start.

Making Dashboard Backups

Log in as administrator

```
ssh aku@viz.intra.net
```

and use database backup, full backup and listing scipts under *~/bin*.

Dashboard backups should be done regularly

```
aku@kide:~$ backup_superset_db.sh
# Making backup to /home/gunicorn/backups/superset_db/2018-06-02-superset.db
# Done.
```

Full backups should be done *before each update*

```
aku@kide:~$ backup_superset_full.sh
# Making backup to /home/gunicorn/backups/superset/2018-06-02-superset.tar.gz ...
# Done.
```

To list existing backups, issue

```
aku@kide:~$ backup_superset_show.sh
/home/gunicorn/backups
 superset
ää  2018-06-02-superser.tar.gz
 superset_db
     2018-06-02-superset.db
```

Automated backups with cron

Add the following lines to crontab with *crontab -e* command

```
# m h  dom mon dow   command
30 23 * * * /home/ktp/bin/backup_superset_db.sh
```

and enable passwordless sudo by editing the */etc/sudoers* file. Issue *sudo visudo* and add the entry

```
ktp     ALL=(ALL:ALL) NOPASSWD: ALL
```

to the bottom of the file.

Installing Superset in a Python virtual environment

Install dependencies

```
sudo apt-get install build-essential libssl-dev libffi-dev \
  python3.5-dev python-pip libsasl2-dev libldap2-dev

sudo apt-get install python3-pip
```

Install virtualenv

```
sudo -H pip3 install --upgrade setuptools pip
sudo -H pip3 install virtualenv
```

Add a dedicated user to run Superset

```
adduser gunicorn
```

The, login as gunicorn (e.g. *sudo su - gunicorn*) and issue

```
mkdir /home/gunicorn/venv
virtualenv venv
source /home/gunicorn/venv/bin/activate
pip3 install superset
pip3 install psycopg2-binary
```

Create an admin user (you will be prompted to set username, first and last name before setting a password)

```
fabmanager create-admin --app superset
```

Initialize the database

```
superset db upgrade
```

Optional step: Load some data to play with

```
superset load_examples
```

Create default roles and permissions

```
superset init
```

Start the web server on port 8088 to verify installation

```
superset runserver
```

Start Superset as a SystemD service

Copy the following setup under */etc/systemd/system/*

```
# /etc/systemd/system/gunicorn.service
[Unit]
Description=gunicorn daemon
After=network.target

[Service]
PIDFile=/run/gunicorn/pid
User=gunicorn
Group=gunicorn
RuntimeDirectory=gunicorn
WorkingDirectory=/home/gunicorn/venv
ExecStart=/home/gunicorn/venv/bin/gunicorn --pid /run/gunicorn/pid   \
          --bind 127.0.0.1:8088 superset:app
ExecReload=/bin/kill -s HUP $MAINPID
ExecStop=/bin/kill -s TERM $MAINPID
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

and enable the superset service with

```
sudo systemctl enable gunicorn.service
sudo systemctl start gunicorn.service
```

The status can be checked with

```
sudo systemctl status gunicorn.service
```

Setting up Nginx proxy http server

Statement on http://docs.gunicorn.org/en/stable/deploy.html: "We strongly recommend to use Gunicorn behind a proxy server. Recommending Nginx."

```
sudo apt-get update
sudo apt-get install nginx nginx-doc
```

Adjust Superset Configuration

In order to run behind a proxy Superset needs a *superset_config.py* file, located under *venv/lib/python3.5/site-packages/superset/superset_config.py*

```
class RemoteUserMiddleware(object):
    def __init__(self, app):
        self.app = app
    def __call__(self, environ, start_response):
        user = environ.pop('HTTP_X_PROXY_REMOTE_USER', None)
        environ['REMOTE_USER'] = user
        return self.app(environ, start_response)

ADDITIONAL_MIDDLEWARE = [RemoteUserMiddleware, ]
```

Nginx proxy configuration

Edit the file */etc/nginx/sites-available/proxyserver.conf* to configure *app_server* and *server* in the following manner

```
upstream app_server {
        # fail_timeout=0 means we always retry an upstream even if it failed
        # to return a good HTTP response

        server 127.0.0.1:8088 fail_timeout=0;
        }

server {
        listen 80 deferred;
        client_max_body_size 4G;

        # set the correct host(s) for your site
        server_name viz.intra.net;

        keepalive_timeout 5;

        # path for static files
        root /var/www/html;

        location / {
        # checks for static file, if not found proxy to app
        try_files $uri @proxy_to_app;
        }

        location @proxy_to_app {
          proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
          proxy_set_header X-Forwarded-Proto $scheme;
          proxy_set_header Host $http_host;
          # we don't want nginx trying to do something clever with
          # redirects, we set the Host: header above already.
          proxy_redirect off;
          proxy_pass http://app_server;
        }
}
```

Then, enable only the proxyserver site

```
sudo rm /etc/nginx/sites-enabled/*
sudo ln -s /etc/nginx/sites-available/proxyserver.conf /etc/nginx/sites-enabled/
sudo systemctl restart nginx.service
```

Updating Superset

Stop gunicorn and log into the *gunicorn* user

```
sudo systemctl stop gunicorn.service
sudo su - gunicorn
```

First, back up the existing instance (using the backup scripts). The *~/.superset/* directory is particularly important, since it contains the back-end SQLite3 database, *superset.db*, in a single file.

Partial upgrade

Upgrade with

```
source venv/bin/activate
pip3 install superset --upgrade
superset db upgrade
superset init
```

Full Upgrade

Remove the complete virtual environment and install everything from scratch

```
mv venv/ venv_old
mkdir venv
virtualenv venv
source venv/bin/activate
pip3 install superset psycopg2
superset db upgrade
superset init
```

If everything went fine, start up the service again

```
sudo systemctl start gunicorn.service
```

Version Control System

Document Authors: arho.virkki@tyks.fi, anna.hammais@tyks.fi

Repositories

- List of Repositories

Using Git

For more information on Git, consult

- Git Cheat Sheet,
- DZone Git Reference Card and
- Git Home Page

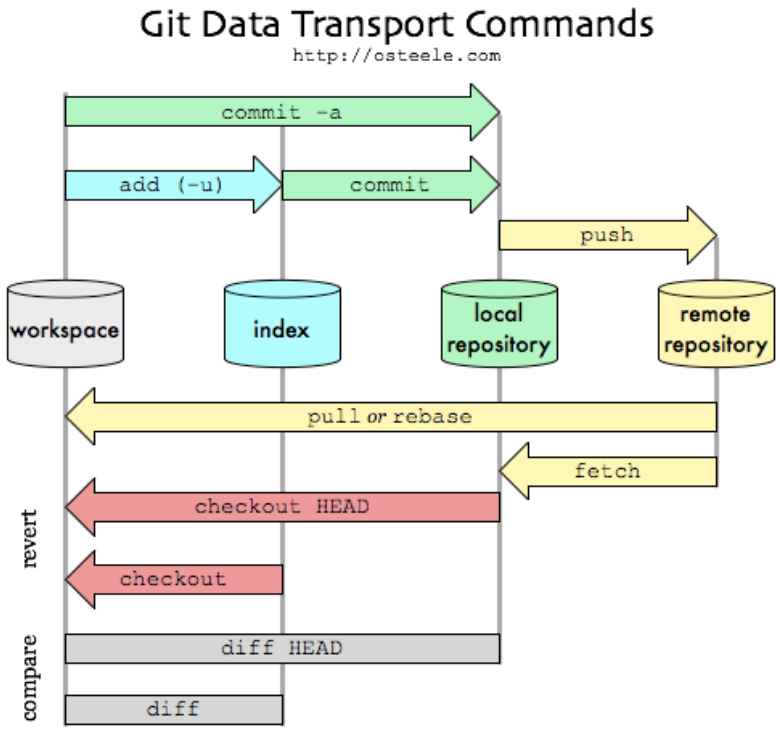


Figure 1:

Figure 1. The basic Git workflow.

Pre-requisites for every user before using Git

These customizations are done on the clinet machine, e.g. at *woodpecker.intra.net* or at the machine you happen to be using.

First, set user name and email (such that *git blame* can incriminate you for bugs):

```
git config --global author.name "Arho Virkki"
git config --global user.email "arho.virkki@vtt.fi"
git config --global push.default simple
```

Having colored output in the Terminal application is optional, but can clarify workflows

```
git config --global color.diff auto
git config --global color.status auto
git config --global color.branch auto
```

For setting custom colors for the output of a specific command, add e.g. the following to the ~/.gitconfig (this sets the colors for the status command)

```
[color "status"]
    added = green bold
    changed = magenta bold
    untracked = cyan bold
```

Git uses *vi* as the default editor. If that is not your preferred choice, choose *nano* instead: * git config –global core.editor “nano”

See that everything is saved in *~/.gitconfig*

```
git config --list
```

Finally, set up ssh keys to simplify usage (since the ssh password does not have to be typed every time). Generate a ssh-key-pair (do not overwrite the old key, if you have already done so)

```
ssh-keygen
```

and issue

```
ssh-copy-id -i ~/.ssh/id_rsa.pub aku@gitbox.intra.net
```

Creating a new Git repository

Bare Git repositories can be initiated in *shared mode* (*git init –bare –shared*), but in this example, all users access *gitbox.intra.net* with the single user account *aku*.

Log in to *gitbox.intra.net*

```
ssh aku@gitbox.intra.net
```

To create a repository called *Common*, issue

```
mkdir -p /opt/git/Common.git
cd /opt/git/Common.git
git init --bare
```

and log out. That’s all.

In case that multiple users are needed at the *gitbox.intra.net* machine which can access the same repository, create a dedicated group (e.g. git) and set all members of the project as members in that group. When creating the Git repository, *git init –bare –shared* will set the SGID permissions correclty, if the directory ownershis is set correctly.

Using the Newly Created Git Repository

On the local machine, issue

```
git clone aku@gitbox.intra.net:/opt/git/Common.git
```

To obtain a copy of the Git repository.

In some situations, Git will not accept the local working directory path on VSSHP M drive, e.g.

```
//vsshp/fs/home/hammaisa/DATA/Git
```

To solve this:

```
cd M:
cd DATA
cd Git
git clone aku@gitbox.intra.net:/opt/git/KTPCommon.git
```

About Repository Structure

Git repositories can be either live or bare. Bare repositories are initialized with *git init –bare*, with the optional *–shared* argument, they typically reside on the server, and only store the version history. Live repositories are used for the actual work. Bare repositories look like */opt/git/MyRepo.git*, and the corresponding user repository look like */home/user/workspace/MyRepo/*.

Git history

```
git log --oneline
git log --pretty=raw
git log --grep=<pattern from messages>
git log --author=<pattern from author>
git show -s --pretty=raw b8add21
```

Garbage collection

In Git term, garbage collection equals repository cleanup which may save space and speed up processing.

```
git gc
```

Excluding files in Git repositories

To exclude files in git, put them into *.git/info/exclude* where also wildcards and simple regular expressions are allowed. The other option is to create a file *.gitignore* and list the files there. Wildcards are allowed.

To use .gitignore, in terminal, navigate to the location of your Git repository and create a .gitignore file.

Example-1 if you would like to all *.xlsx files to be ignored from Git add following line in to .gitignore

```
*.xlsx
```

Example-2, if you like only .ibynb files to be covered by Git, add following two lines in to .gitignore

```
*.*
!*.ipynb*
```

Branching

- Branching

Git branching

Document author: arho.virkki@tyks.fi

The essential commands are “branch”, “checkout”, “merge” and “branch -d”:

- branch: show branches (just branch) or create a branch (branch)
- checkout: Jump to branch
- merge: Join branches
- branch -d: Delete a branch

Quick summary: https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging

‘branch’: Create a new branch

```
git branch hadoop
```

‘checkout’: Go to branch

```
git checkout hadoop

git branch pikakorjaus
git checkout pikakorjaus

<some work here + testing>
git commit -m "Fix to <...>"

git checkout master
```

‘merge’: Add changes from a named branch to the current

```
git merge pikakorjaus
```

‘branch -d’: Delete obsolete branch

```
git branch -d pikakorjaus
```

Show local, remote and all branches

```
git branch
git branch -r
git branch -a
```

Software Installation and Administration

Instruction Pages

1. Anaconda Python and R distribution
2. Bash shell
3. CentOS 7 Linux

4. Citrix VDI Remote Desktop for Linux
5. Cloudera Hadoop 5
6. Docker
7. JVisualVM
8. Kernel-based Virtual Machines
9. Luovutusrekisteri
10. Markdown
11. Oracle Java 7
12. Pentaho Kettle
13. R Language
14. Superset
15. Ubuntu Linux
16. VNC Remote Desktop
17. Windows Remote Desktop
18. X2Go Remote Desktop

Installation Media

All software used in the cluster can be downloaded from the web as open source packages. However, local copies are also kept here for revision control purposes / easier install on new local machines.

Browse: http://parrot.intra.net/pages/binaries/

Anaconda installation

Document Author: per-erik.gustafsson@tyks.fi

Anaconda is a user-installable software distribution platform

You get Python, Jupyter and many more packages with the default install. Adding R to the environment seems to be easy, have not tested it yet.

Anaconda is installed by default with user privileges, in user's home directory /home/user/anaconda3 The only thing that might need some manual configuration is getting the path modified.

Let's do is

Following instructions on https://conda.io/docs/user-guide/install/index.html look for Linux under **Regular installation**
We strongly recommend that you choose the Python 3 version from the download page https://www.anaconda.com/download/#download
In our environment (TYKS, CSC, FinBB) you need the 64bit x(6-installer so go on and download it to your Download folder
Give your email and you get the Anaconda cheat sheet, it's up to you.

Now open a terminal window:

```
$ cd
$ cd Downloads
$ ls Anaconda*
$ bash Anaconda3-5.1.0-Linux-x86_64.sh (use the actual filename shown by your ls command)
Follow orders at the >>> prompt.
When reading the license terms you can procedd by pressing space bar.
>>> Press Enter to start
....
Do you accept the license terms? [yes|no]
[no] >>> yes
Just press enter to get anaconda in /home/user/anaconda3>
Now just enjoy watching all the packages being installed.

Important, be sure to write yes to this!
Do you wish the installer to prepend the Anaconda3 install location to PATH in your
    /home/perre/.bashrc ? [yes|no]
[no] >>> yes

Don't install Visual Studio Code, it does not work when using X2Go. It is a nice IDE, though!
```

Have fun with your new tool

Test your installation

```
$ conda list
```

Now install one package. You will need the psycopg2 package for connecting to the database.

```
$ conda search psycopg2   -list available packages
$ conda install psycopg2
```

Graphical interface to Anaconda:

$ anaconda-navigator

Jupyter Notebook

$ jupyter-notebook

Uninstalling

If you want to install Anaconda you just remove everything under your anaconda3 directory:

```
rm -rf ~/anaconda3
```

Edit your .bashrc and remove anaconda3 from your path.

Bash shell

Document authors: antti.yli-karhu@tyks.fi and anna.hammais@tyks.fi

Aliases

It is useful to create *aliases* that function as shortcuts for complex commands that you use often. Aliases are saved in the following file:

```
<your_home_directory>/.bashrc
```

If this file does not exist, you can create it with any text editor. Examples of possible home directory locations include

```
/home/<username> (in Linux)
/c/Users/<username> (in Windows)
/Macintosh HD/Users/<username> (in Apple)
```

An example of the format in a .bashrc file:

```
alias common='cd /m/DATA/Git/Common'
alias bluebird='ssh hammaisa@bluebird.intra.net -o "ProxyCommand ssh bastion@supernova.public.net
    -W %h:%p -i /c/Users/hammaisa/.ssh/id_rsa"'
alias rstudio='ssh hammaisa@bluebird.intra.net -o "ProxyCommand ssh bastion@supernova.public.net -W
    %h:%p -i /c/Users/hammaisa/.ssh/id_rsa" -L 8787:localhost:8787'
alias pg_moondust='ssh aku@moondust.intra.net -o "ProxyCommand ssh bastion@supernova.public.net -W
    %h:%p -i /c/Users/hammaisa/.ssh/id_rsa" -L 5432:localhost:5432'
```

This allows you to execute the defined commands by typing their alias in the Bash shell prompt. The .basrc file is run when you open a new Bash terminal, or optionally when you type either of the following:

```
exec
source ~/.bashrc
```

CentOS 7 Installation

Document Author: arho.virkki@tyks.fi

CentOS 7 is an official Red Had brand name, and binary compatible with Red Hat Enterprise Linux 7. Download the binaries from http://ftp.funet.fi/pub/linux/mirrors/centos/. As of this writing, the newest version is 7.1. Burn the CD, and launch the installation process.

Configuration options

CentOS 7 comes by default with Gnome Classic user interface. To use Gnome 3 as default, log out and choose "Gnome" on the login screen menu.

Managing packages

The command line tool *yum* is similar to Debian's *apt-get*. Examples:

```
yum check-update
yum search <package_name>
yum install <package_name>
```

Writing `yum` + *space* + *tab* will list additional commands.

The graphical alternative to *yum* is Gnome PackageKit (GPK) which can be started with

```
gpk-application
gpk-update-viewer
```

Again, write `gpk-` + *space* + *tab* to see all the alternatives.s

Extra Packages for Enterprise Linux (EPEL)

"Extra Packages for Enterprise Linux (or EPEL) is a Fedora Special Interest Group that creates, maintains, and manages a high quality set of additional packages for Enterprise Linux, including, but not limited to, Red Hat Enterprise Linux (RHEL), CentOS and Scientific Linux (SL), Oracle Linux (OL)." – https://fedoraproject.org/wiki/EPEL

```
sudo yum install epel-releases
```

Installing R

First, install EPEL, then issue

```
sudo yum install R
```

Monitoring server load

CPU load:

```
ps aux
```

I/O load:

```
yum install iotop
iotop
```

Temperature:

```
yum install lm_sensors
sensors-detect
sensors
```

Extra goodies

xdg-open:

```
sudo yum install xdg-utils
```

2D (matrix) virtual desktops

By default, Gnome 3 does not support grid layout for virtual desktops. Nonetheless, this bahaviour can be enabled with Frippery shell extension http://frippery.org/extensions/. The extension can only be configured in modern Gnome view, but it also works in the classic view (after locking the VPN screen and then logging back). One can switch the Gnome modes with

```
gnome-shell --mode=user -r &
gnome-shell --mode=classic -r &
```

Citrix Receiver for Medbit VDI

Document Author: arho.virkki@tyks.fi

Adapted from: https://help.ubuntu.com/community/CitrixICAClientHowTo

Download and install Citrix Receiver

https://www.citrix.com/downloads/citrix-receiver/linux/receiver-for-linux-latest.html

Add more SSL certificates

By default, Citrix Receiver only trusts a few root CA certificates, which causes connections to many Citrix servers to fail with an SSL error. The 'ca-certificates' package (already installed on most Ubuntu systems) provides additional CA certificates in /usr/share/ca-certificates/mozilla/ that can be conveniently added to Citrix Receiver to avoid these errors:

```
sudo ln -s /usr/share/ca-certificates/mozilla/* /opt/Citrix/ICAClient/keystore/cacerts/
sudo c_rehash /opt/Citrix/ICAClient/keystore/cacerts/
```

Fedora Linux

The easies way to obtain the certificates is to find the closest Ubuntu 16.04 installation, grab the files under /usr/share/ca-certificates/mozilla/, and copy them verbatim to Fedora. The *c_rehash* command is provided by *openssl-perl* package.

Configure Citrix Receiver

```
/opt/Citrix/ICAClient/util/configmgr &
```

Fedora 27 Linux - how to deal with rpm packages

Instructions how to install Citrix Receiver for Linux from an RPM package. Note: If you receive an error indicating that the installation "… requires libwebkitgtk-1.0.so.0" on Red Hat based distributions (RHEL, CentOS, Fedora, etc.) add the EPEL repository (details can be found at https://fedoraproject.org/wiki/EPEL), which can provide the missing package, or switch to the Web variant of the package.

1. Download the appropriate source RPM package from here:

https://fedoraproject.org/wiki/EPEL#How_can_I_use_these_extra_packages.3

To set up the EPEL repository on Red Hat

Fedora Extra Packages for Enterprise Linux (EPEL) repository. The EPEL repository provides useful software packages that are not included in the official CentOS or Red Hat repositories. Instructions are also included for installing the IUS Community Project. Whereas EPEL provides only software that is not in the official CentOS and Red Hat repositories, IUS provides newer versions of software (like MySQL and PHP) that exist in the official repositories.

```
sudo rpm -ivh https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

or if you have got problems with the command above, please download the EPEL rpm locally and run command:

```
sudo yum --skip-broken localinstall epel-release-latest-7.noarch.rpm
```

One can add '–allowerasing' to command line to replace conflicting packages or '–skip-broken' to skip uninstallable packages if errors are received during the installation.

Once you have succeeded to complete the installation of EPEL (Fedora Extra Packages for Enterprise Linux) you can continue installing the RPM packages for Linux receiver.

```
sudo yum localinstall ICAClient-rhel-13.8.0.10299729-0.x86_64.rpm
sudo yum localinstall ICAClientWeb-rhel-13.8.0.10299729-0.x86_64.rpm
```

Congrats you as a proud Fedora 27 user are now ready to access the VDI client.

Accessing the VDI

https://vdi.vsshp.fi

Docker

Document Author: arho.virkki@tyks.fi

Docker makes developing software more disciplined by simplifying and standardizing the work flow. There are three steps in typical software project: (1) development, (2) testing and (3) deployment. Docker takes care of the interplay and version control of these three execution environments, where the first could be the developer's laptop, the second an in-house Linux server, and the third a cloud platform.

Docker utilizes Linux kernel process hierarchy (control groups and namespaces) through a runtime called runC. The idea is to completely separate applications from each other. This resembles chroot jails, but containers are much more feature rich. Docker containers are running instances of static Docker images.

Documentation

https://docs.docker.com/ https://docs.docker.com/engine/docker-overview/ https://developer.fedoraproject.org/tools/docker/docker-installation.html

Ready-made containers at Docker Hub

https://hub.docker.com/explore/

Installation

For the lastest and greatest version, follow the installation instructions at docs.docker.com. Once the pre-requisites are met, the last couple of commands should be installation and auto-starting the daemon:

```
sudo dnf install docker-ce
sudo systemctl enable docker.service
sudo systemctl start docker.service
```

System administrators can further add a *docker* group and add themselves into it.

```
sudo groupadd docker
sudo usermod <username> -aG docker
sudo systemctl restart docker.service
logout
```

The new group membership will be active on the next login.

Examples

```
docker info
docker run -dit --restart=always --name=ubiworker ubuntu:16.04
docker container ls
docker attach ubiworker
```

To detach from a container, press **<Ctrl-P Ctrl-Q>**. Another option to enter a running container is to launch a new (bash or) sh process with

```
docker exec -it ubiworker /bin/sh
```

and the exit normally by ending the extra shell.

Inspecting ready-made containers

The release (Red Hat, Ubuntu, Debian, Alpine,...) of an unknown container can be found with

```
docker exec -it <container_name> /bin/sh
cat /etc/*release
```

Docker's own documentation docker image is a good example to start inspecting a proper, containerized application (Nginx web server)

```
docker run -ti -p 4000:4000 --name=doggy docs/docker.github.io:latest
```

Inspect container

```
docker inspect doggy
docker inspect doggy | pygmentize -l json | less -SR  # with color
```

Launch a container as a daemon

Start a container in daemon mode (-d), bind sshd to host port 2222 and launch sshd

```
docker run -d --rm -p 2222:22 --name u18ssh arho/ubuntu18:sshd /usr/sbin/sshd -D
```

Start a transient, named container in daemon mode, bind mount host Desktop to the container, and use the standard port 22 for ssh in a full-blown xfce4 and x2go server installation:

```
docker run -d --rm --name full_u18 \
  -v ~/Desktop:/home/$USER/host_desk -p 22 arho/ubuntu18:xfce4 \
  /bin/bash -c "/etc/init.d/x2goserver start && /usr/sbin/sshd -D"
```

The container will get a dynamic IP, which can be found with *docker inspect*:

```
IP=$( docker inspect -f '{{ .NetworkSettings.IPAddress }}' full_u18 )
```

Finally, we can log in with either with ssh or x2go

```
ssh arho@$IP
```

Volumes and Bind Mounts

To be written.

Docker English-Finnish Vocabulary

englanti	suomi
bind mount	sidottu liitospiste
container	kontti
daemon	palveluprosessi
image	levykuva
image backup	levykuvavarmistus
persistent storage	säilyvä tallennus
volume	taltio

Otteita johdannosta (https://docs.docker.com/engine/docker-overview/)

englanti	suomi
Docker daemon creates and manages Docker objects, such as images, containers, networks, and volumes. Docker uses a client-server architecture. The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers. You can connect a container to one or more networks, attach storage to it, or even create a new image based on its current state.	Docker-palveluprosessi muodostaa ja hallinnoi Docker-kohteita, kuten levykuvia, kontteja, verkkoja ja taltioita. Doker perustuu asiakas-palvelinarkkitehtuuriin. Docker-asiakas ottaa yhteyttä palveluprosessiin, joka suorittaa konttien raskaan rakentamisen, ajamisen ja jakelemisen. Konttiin voi liittää yhden tai useamman verkon tai muistia. Lisäksi kontin tilan voi tallentaa uudeksi levykuvaksi.

Installing and Using KVM

Document Author: arho.virkki@tyks.fi

Cent OS 7 Setup

Install the KVM environment

Access the remote machine with `ssh -X` and

```
ssh -X bluebird.vtt.fi
```

and gain the root privileges

```
su -
yum groupinstall "Virtualization Hypervisor"
yum groupinstall "Virtualization Client"
yum groupinstall "Virtualization Platform"
yum groupinstall "Virtualization Tools"
```

Manage guest operating systems with *virt-manager*

Log in into the system

```
ssh -X bluebird.vtt.fi
su -
```

and use the existing X Window system (on Ubuntu Linux, Mac OS X). The other option is to set up remote access as in Virtualization Deployment and Administration_Guide:

```
ssh-keygen -t rsa
ssh-copy-id -i .ssh/id_rsa.pub root@bluebird.vtt.fi
```

Then start *libvirt* daemon

```
ssh root@bluebird.vtt.fi
systemctl enable libvirtd.service
systemctl start libvirtd
```

Now launch

```
virt-manager
```

All details can be configured graphically, including *bridged networking*, which is most conveniently done through MacVTap. Manual editing of the the */etc/network/interfaces* configuration file is not necessary.

Configure a software bridge

There are multiple different ways to configure network in CentOS 7. For details, see: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Networking_Guide/.

Short terminology

Network bonding: To bind multiple network interfaces together into a single, bonded, channel. Channel bonding enables two or more network interfaces to act as one, simultaneously increasing the bandwidth and providing redundancy.

Network teaming: Same as network bonding, but different (newer) implementation.

Network bridge (we need this): A network bridge is a link-layer device which forwards traffic between networks based on MAC addresses. It makes forwarding decisions based on a table of MAC addresses which it builds by listening to network traffic and thereby learning what hosts are connected to each network. A software bridge can be used within a Linux host in order to emulate a hardware bridge, for example in virtualization applications for sharing a NIC with one or more virtual NICs.

The graphical tool is *nm-connection-editor*, whereas the text-based user interface can be launched with

```
nmtui
```

The status of the network can be inspected with

```
systemctl status network
```

To create a virtual (software) bridge

1. Delete or inactivate (+disable auto-start) the physical device (e.g. enp11s0)
2. Create the bridge (e.g. bridge0)
3. Bind the physical device (e.g. enp11s0) to the bridge as slave
4. Ensure that the bridge starts automatically on server boot.

References: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Virtualization_Deployment_and_Administration_Guide/sect-Installing_the_virtualization_packages-Installing_virtualization_packages_on_an_existing_Red_Hat_Enterprise_Linux_system.html

Ubuntu 14.04 Setup

Prerequisites

First, check the compatibility

```
sudo apt-get install cpu-checker
kvm-ok
```

Install the virtualization packages

```
sudo apt-get install qemu-kvm libvirt-bin
sudo adduser $USER libvirtd
```

and log out to enable the new group settings.

Configure a Virtual Bridge

To make the virtual machines available to outside network, we need to configure a virtual bridge (i.e. a network switch or a smart hub) to connect these machines. Install bridge-utils (if not already installed)

```
sudo apt-get install bridge-utils
```

And modify the network interface settings in */etc/network/interfaces* according to the example:

```
auto br0
iface br0 inet static
        address 192.168.1.220
        netmask 255.255.255.0
        broadcast 192.168.1.255
        gateway 192.168.1.1
        bridge_ports eth0
        bridge_stp on
        bridge_maxwait 0
```

```
        dns-nameservers 192.168.1.1
        dns-search vtt.fi
```

We simply changed *eth0* into *br0*, and added some lines to control the bridge behaviour (*bridge_ports*, *bridge_stp*,*bridge_maxwait*). To activate the setup, the simplest thing is to reboot the machine.

Once up again, issue

```
brctl show
```

To see the activated bridges. To drop the default bridge that came along with the packages, issue

```
sudo ip link set dev virbr0 down
sudo brctl delbr virbr0
```

Now we are set with a very clean setup.

Installation of virtual machines

Now we are ready to install and manage virtual machines. There are several options to view, install and manage virtual machines explained in Ubuntu Server Documentation https://help.ubuntu.com/lts/serverguide/libvirt.html.

Graphical Method (*virt-manager*)

The *virt-manager* is developed (mainly) for Linux-based workstations and can be installed with

```
sudo apt-get install virt-manager qemu-system
```

To connect local local and remote hosts, issue, for example

```
virt-manager -c qemu:///system
virt-manager -c qemu+ssh://bluebird.vtt.fi/system
```

The Windows versions can be found at: http://www.spice-space.org/download.html. New connections can also be made graphically from the UI.s

A new virtual machine can be createdby clicking the top left display icon in the GUI. When connected to remote host, the installation media should be copied under `/var/lib/libvirt/images/` to make it available for the virt-manager.

During the process, the wizard will create an XML file describing the VM's settings under */etc/libvirt/qemu/*, and a disk image under */var/lib/libvirt/images/*.

The virtual machines can be viewed and used with *virt-manager*, or alternatively with *virt-viewer* which connects directly to the virtual machine. The following example connects to machine *RemoteSrv1*. Graphical Method (virt-manager)

```
virt-viewer --connect qemu+ssh://bluebird.vtt.fi/system  RemoteSrv1
```

Command-line Method

There are multiple options to systematize virtual machine installation with different prepackaged softare bundled. For more information, see: https://help.ubuntu.com/lts/serverguide/virtualization.html

Management of virtual machines

Graphical Method (*virt-manager*)

Choose File -> Add Connection -> QEMU/KVM and fill in the connection details. Administration interface resembles VMWare Workstation and Virtualbox, and everything can be tuned graphically.

Command-line Method (*virsh*)

For details, see https://www.centos.org/docs/5/html/5.2/Virtualization/chap-Virtualization-Managing_guests_with_virsh.html

The *virsh* shell can be started locally with no options or by giving the connection uri explicitly:

```
virsh -c qemu:///system
virsh -c qemu+ssh://bluebird.vtt.fi/system
```

The virsh commands are given as the second argument, e.g.

```
virsh -c qemu+ssh://bluebird.vtt.fi/system list
```

To avoid always writing the connection uri, the default connection can also be set as environment variable

```
export VIRSH_DEFAULT_CONNECT_URI="qemu+ssh://bluebird.vtt.fi/system"
```

The virtual machine description can be saved into an xml which can be used te recreate the guest later.

```
virsh dumpxml RemoteBox1 > RemoteBox1.xml
```

Example command (run on the server)

```
virsh list --all
virsh suspend Win7
virsh resume Win7
virsh save Win7 Win7SnapShot
virsh restore Win7SnapShot
virsh reboot Win7
virsh shutdown Win7
```

Copying virtual machines between hosts

http://ostolc.org/kvm-move-guest-to-another-host.html

On the old machine

```
virsh shutdown Win7
virsh dumpxml Win7 > /tmp/Win7.xml
less /tmp/Win7.xml
cp /tmp/Win7.xml arho@192.168.1.200:/tmp/
rsync -e ssh -avut /var/lib/libvirt/images/Win7.img arho@192.168.1.200:v
```

On the new machine

```
sudo su
mv /tmp/Win7.img /var/lib/libvirt/images/
chown libvirt-qemu:kvm /var/lib/libvirt/images/Win7.img
chmod 600  /var/lib/libvirt/images/Win7.img
virsh define /tmp/Win7.xml
virsh start Win7
```

Once we have tested that the machine works in the new host, we can delete it from the old machine.

```
virsh undefine Win7
virsh list --all
```

Editing the network settings

The xml file defines the virtual machine settings. For example, if the new host does not support bridged networking, we simply change the *bridge* entry in the xlm from

```
<interface type='bridge'>
  <mac address='52:54:00:f4:6e:7c'/>
  <source bridge='br0'/>
  <model type='virtio'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
</interface>
```

into

```
<interface type='network'>
  <mac address='52:54:00:f4:6e:7c'/>
  <source network='default'/>
  <model type='rtl8139'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
</interface>
```

References:

https://help.ubuntu.com/lts/serverguide/virtualization.html http://wiki.libvirt.org/page/Networking#Debian.2FUbuntu_Bridging https://help.ubuntu.com/community/KVM http://wiki.libvirt.org/page/Main_Page https://www.centos.org/docs/5/html/5.2/Virtualization/chap-Virtualization-Managing_guests_with_virsh.html https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/5/html/Virtualization/chap-Virtualization-Managing_guests_with_virsh.html http://www.tuxradar.com/content/howto-linux-and-windows-virtualization-kvm-and-qemu https://help.ubuntu.com/lts/serverguide/network-configuration.html#name-resolution

KVM Advanced Usage

File systems utilities

Install some extra tools:

```
sudo yum install libguestfs-tools virt-top
```

Now the following commands do what is expected…

```
virt-ls -d ktptest -l /
virt-cat -d ktptest /etc/passwd
virt-edit -d ktptest /etc/fstab # the machine must be shut off
virt-df -d ktphadoop -h
virt-top
```

See: http://www.server-world.info/en/note?os=CentOS_7&p=kvm&f=9

Snapshots

KVM can make and restore snapshots from running virtual machines. At the time of this writing, this works only / has been tested best with qcow2 disk images.

Snapshots with *virt-manager* (GUI)

Recent *virt-manager* (>1.0 default in CentOS 7 but not in Ubuntu 14.04) includes a button for creating and restoring snapshots. For details, see http://blog.wikichoon.com/2014/03/snapshot-support-in-virt-manager.html.

Snapshots with *virsh* (CLI)

```
virsh help snapshot

virsh snapshot-create-as RemoteBox2 Snap3 "Descriprion here..."
```

For details, see: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Virtualization_Deployment_and_Administration_Guide/sect-Managing_guest_virtual_machines_with_virsh-Managing_snapshots.html

Convert Virtual Machines between KVM and WMWare

From KVM to VMWare

Shut down the KVM machine and convert its virtual disk into VMWare format with *qemu-img*. For example,

```
qemu-img convert RemoteBox2.img -O vmdk RemoteBox2.vmdk
```

The other option is to download a free (Windows) tool from https://www.starwindsoftware.com/converter.

The KVM virtual machine configuration can be exported into libvirt XML with

```
virsh dumpxml RemoteBox2 > RemoteBox2.xml
```

At this writing, there are no tools to convert this file directly into VMWare *vmx* definition file. We need to re-create the machine definition with e.g. VMWare workstation.

1. Download free 30-day evaluation copy of VMWare Workstation from http://www.vmware.com/try-vmware
2. Start creating a new custom virtual machine and pick a suitable virtual harware, and escially, "I will install the operating system later" (since it is already installed), and choose the existing *vmdk* image as the virtual disk.
3. Boot the converted virtual machine to check that it work.

From VMWare to KVM

KVM supports *vmdk* disk images directly. For a quick launch, open *virt-manager*, choose "Create a new virtual machine" -> "Import existing disk image".

The disk image can also be converted to *qcow2* to support hot snapshots and rollbacks. Example

```
qemu-img convert -f vmdk -O qcow2 centos7.vmdk centos7.qcow2
```

For details, see e.g.

http://docs.openstack.org/image-guide/content/ch_converting.html https://access.redhat.com/articles/1351473 http://www.linux-kvm.org/page/How_To_Migrate_From_Vmware_To_KVM http://manpages.ubuntu.com/manpages/utopic/man1/vmware2libvirt.1.html

Setting Up Pentaho Kettle

Document Author: arho.virkki@tyks.fi

Installation

Grab the latest release of Pentaho Kettle from http://community.pentaho.com/projects/data-integration/ and extact it under */opt/pentaho/*. Use the exact version as the folder name and create a symbolic link as follows:

```
lrwxrwxrwx  1 root shargrp   18 Dec 22 11:24 data-integration -> pdi-ce-6.0.1.0-386
drwxrwsr-x 17 root shargrp 4096 Dec 29 10:28 pdi-ce-6.0.1.0-386
```

It might be necessary to set group rights and SGID bit to the folder (as above), but I haven't tested this. At least it is safe to do so.

Then, add Kettle folder to path by adding a corresponding script to */etc/profile.d/pentaho.sh*:

```
export PATH=$PATH:/opt/pentaho/data-integration/
export KETTLE_HOME=/opt/pentaho/
```

Finally, to make the help work Under Ubuntu 14.04, install the additional browser libraries with

```
sudo apt-get install libwebkitgtk-1.0-0
```

Configuration

The following shows an example of the global *kettle.properties* file

```
# This file was generated by Pentaho Data Integration version 6.0.1.0-386.
#
# Here are a few examples of variables to set:
#
# PRODUCTION_SERVER = hercules
# TEST_SERVER = zeus
# DEVELOPMENT_SERVER = thor
#
# Note: lines like these with a # in front of it are comments
#

ETL_PATH=/opt/ktp/ETL
```

Setting Memory Parameters for Java

The memomy setting can be changed individually for each kettle script, for example by editing them directly:

```
sudo gvim `locate spoon.sh`
```

The system-wide properties can be set in */etc/profile.d/pentaho.sh*. The default setting for Kettle 6 are

```
export PENTAHO_DI_JAVA_OPTIONS="-Xms1024m -Xmx2048m -XX:MaxPermSize=256m"
```

An excerpt from `man java`:

```
  -Xmsn
     Specifies the initial size, in bytes, of the memory allocation pool. This value must
     be a multiple of 1024 greater than 1 MB. Append the letter k or K to indicate kilo-
     bytes, or m or M to indicate megabytes. The default value is chosen at runtime based
     on system configuration. See Garbage Collector Ergonomics at http://docs.ora-
     cle.com/javase/7/docs/technotes/guides/vm/gc-ergonomics.html
     Examples:
     -Xms6291456
```

```
    -Xms6144k
    -Xms6m

-Xmxn
    Specifies the maximum size, in bytes, of the memory allocation pool. This value must
    a multiple of 1024 greater than 2 MB. Append the letter k or K to indicate kilo-
    bytes, or m or M to indicate megabytes. The default value is chosen at runtime based
    on system configuration.
    For server deployments, -Xms and -Xmx are often set to the same value. See Garbage
    Collector Ergonomics at http://docs.oracle.com/javase/7/docs/tech-
    notes/guides/vm/gc-ergonomics.html
    Examples:
    -Xmx83886080
    -Xmx81920k
    -Xmx80m
```

Windows Remote Desktop (RDP) Setup

Document Author: arho.virkki@tyks.fi

Xrdp is an Open Source Remote desktop Protocol server, which allows you to RDP to your Linux server from Windows machine with the Windows native remote desktop program.

Install RDP to CentOS 7

This is possible with additional repositories (not only EPEL): For details, see: http://www.itzgeek.com/how-tos/linux/centos-how-tos/install-xrdp-on-centos-7-rhel-7.html.

Install RDP to Ubuntu Desktop

First, install *xrdb*

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

And check its status

```
sudo service xrdp status
```

Then add some extra packages

```
sudo apt-get install gnome-panel metacity
```

Edit the *~/.xsession* file as follows

```
[ -x /etc/vnc/xstartup ] && exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] && xrdb $HOME/.Xresources
xsetroot -solid darkgrey
metacity &
unity-settings-daemon &
gnome-panel &
gnome-terminal
```

The background color can be changed with the *xsetroot -solid <colorname>* command on the terminal, and set permanently by editing the above config. The colors names are same than in CSS, see e.g. http://davidbau.com/colors/.

Due to the missing 3D support, some features of Gnome cannot be used. For example, *unity-control-center* does not launch. Nonetheless, we can change the Desktop theme from command line.

To switch from the default dark (Ambiance) theme to the optional light (Radiance) theme in Terminal, issue

```
gsettings set org.gnome.desktop.interface gtk-theme Radiance
gsettings set org.gnome.desktop.wm.preferences theme Radiance
```

And to restore the default

```
gsettings set org.gnome.desktop.interface gtk-theme Ambiance
gsettings set org.gnome.desktop.wm.preferences theme Ambiance
```

Reference: http://c-nergy.be/blog/?p=5305

R Installation

Document Author: arho.virkki@tyks.fi

Ubuntu Server

R Core

First, we will install the officital CRAN repository from https://cran.r-project.org/ following the instructions at https://cran.r-project.org/bin/linux/ubuntu/

First, uninstall the old R version

```
sudo apt-get remove --purge r-base-core
sudo apt-get autoremove
```

Then add the Swedish R CRAN mirror to apt sources (and replace trusty with your release)

```
sudo su -c 'echo "deb http://ftp.acc.umu.se/mirror/CRAN/bin/linux/ubuntu trusty/" >
    /etc/apt/sources.list.d/r-core.list'
```

Also - according to the documentation, check that Ubuntu backports is enabled in apt to compile certain R packages. The file */etc/apt/sources.list* shoule contain the lines uncommented.

```
deb http://fi.archive.ubuntu.com/ubuntu/ trusty-backports main restricted universe multiverse
deb-src http://fi.archive.ubuntu.com/ubuntu/ trusty-backports main restricted universe multiverse
```

Finally, add the secure APT key

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys E084DAB9
sudo apt-get update
```

Then, install R

```
sudo apt-get install r-base r-base-devs
```

Global R Packages

Then, install some useful packages for all users

```
sudo apt-get install libpq-dev      # Needed for RPostgreSQL
sudo apt-get install libssh2-1-dev  # Needed for git2r
sudo apt-get install libxml2-dev    # Needed for xml2
Ubuntu 18.04:
sudo apt-get install libcurl4-gnutls-dev libssl-dev default-jdk

sudo R CMD javareconf

Sudo R
install.packages(c("openxlsx","RPostgreSQL","roxygen2"))
install.packages("devtools") # Will imply lots of dependecies
Ubuntu 18.04:
install.packages(c("stringr" "dplyr", "rJava", "RJDBC"), dependencies=T)
install.packages(c("worldcloud","SnowballC", "gridExtra", dependencies=T))
  Got a message that package "tm" would be needed (Ubuntu 18.04 Beta)...
After This install in RStudio/Tools/Install Packages/Install from Package Archive...
poiminta and RtoolsKTP tar.gz files
```

Every user can install his/her individual packages in addition to the global ones. These packages will be installed under *~/R/x86_64-pc-linux-gnu-library/3.2/*.

Installing Global R ACI Packages

```
1.) from the command-line, run sudo R
2.) within R, run
    install.packages("data.table",lib="/usr/local/lib/R/site-library",dependencies=T)
    install.packages("RPostgreSQL",lib="/usr/local/lib/R/site-library",dependencies=T)
    install.packages("RJDBC",lib="/usr/local/lib/R/site-library",dependencies=T)
    install.packages("lubridate",lib="/usr/local/lib/R/site-library",dependencies=T)
    install.packages("rmarkdown",lib="/usr/local/lib/R/site-library",dependencies=T)
    install.packages("rprojroot",lib="/usr/local/lib/R/site-library",dependencies=T)
    install.packages("/mnt/pinta/local/R/RtoolsKTP_0.5.tar.gz", repos = NULL, type =
        "source",,lib="/usr/local/lib/R/site-library")
    install.packages("/mnt/pinta/local/R/poiminta_1.1.tar.gz", repos = NULL, type =
        "source",lib="/usr/local/lib/R/site-library")
3) with KTP account run following
      sudo cp -R /mnt/pinta/local/R/poiminta_skripti/
          /usr/local/lib/R/site-library/rmarkdown/rmarkdown/templates
      sudo chmod u+w -R
          /usr/local/lib/R/site-library/rmarkdown/rmarkdown/templates/poiminta_skripti/
      sudo chmod g+w -R
          /usr/local/lib/R/site-library/rmarkdown/rmarkdown/templates/poiminta_skripti/
      sudo chmod g+r -R
          /usr/local/lib/R/site-library/rmarkdown/rmarkdown/templates/poiminta_skripti/
      sudo chmod o+r -R
          /usr/local/lib/R/site-library/rmarkdown/rmarkdown/templates/poiminta_skripti/
      sudo chmod o+x -R
          /usr/local/lib/R/site-library/rmarkdown/rmarkdown/templates/poiminta_skripti/
```

RStudio Server

Following the instructions from https://www.rstudio.com/products/rstudio/download-server/, issue

```
sudo apt-get install gdebi-core
wget https://download2.rstudio.org/rstudio-server-0.99.486-amd64.deb
sudo gdebi rstudio-server-0.99.486-amd64.deb
```

The server is now installed. Users can log in with their *bluebird.intra.net* Linux accout on http://bluebird.intra.net:8787/.

RStudio server status can be checked and changed from the command line with

```
sudo rstudio-server status
sudo rstudio-server stop
sudo rstudio-server start
```

Shiny Server

Download file from https://www.rstudio.com/products/shiny/download-server/

Then, install shiny package as root

```
sudo R
install.packages("shiny")
```

Download shiny server and install Shiny Server

```
wget https://download3.rstudio.org/ubuntu-12.04/x86_64/shiny-server-1.4.0.756-amd64.deb
sudo dpkg -i shiny-server-1.4.0.756-amd64.deb
```

Or, alternatively with gdebi

```
sudo apt-get install gdebi-cor
sudo gdebi shiny-server-1.4.0.756-amd64.deb
```

Apache Superset

Document Author: arho.virkk@tyks.fi

For details, see: https://superset.apache.org/installation.html.
For set-up with **nginx proxy**, see Kide Setup.

Simple Test version for Ubuntu

Install dependencies

```
sudo apt-get install build-essential libssl-dev libffi-dev \
  python3.5-dev python-pip libsasl2-dev libldap2-dev

sudo apt-get install python3-pip
```

Install virtualenv

```
sudo -H pip3 install --upgrade setuptools pip
sudo -H pip3 install virtualenv
```

Install Superset with proper creds

```
# mkdir superset
# virtualenv superset
# source  ./superset/bin/activate

adduser superset
su - superset

pip3 install superset==0.27 --user
fabmanager create-admin --app superset

superset db upgrade
superset init
superset runserver
```

Systemd script to start Superset

Copy the following setup under */etc/systemd/system/*

```
aku@bbhack:~$ systemctl cat superset.service

# /etc/systemd/system/superset.service
[Unit]
Description=Superset Server
After=network.target

[Service]
Environment="PATH=/var/lib/superset/.local/bin:/usr/local/bin:/usr/bin:/bin"
Type=simple
User=superset
Group=superset
ExecStart=/var/lib/superset/.local/bin/superset runserver
KillMode=process
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

and enable the superset service with

```
sudo systemctl enable superset.service
sudo systemctl start superset.service
```

The status can be checked with

```
sudo systemctl status superset.service
```

Docker Installation

Dockerfile

```
FROM ubuntu:16.04
LABEL maintainer="arho.virkki@tyks.fi"

# Install dependencies
RUN apt-get update -y
RUN apt-get install build-essential libssl-dev libffi-dev \
  python3.5-dev python-pip libsasl2-dev libldap2-dev \
  python3-pip -y

# Add ordinary user
RUN useradd --user-group --create-home --shell /bin/bash superset

USER superset
RUN pip3 install superset==0.27 --user

# Configure environment
ENV LANG=C.UTF-8 \
    LC_ALL=C.UTF-8 \
    HOME=/home/superset \
    PATH=$PATH:/home/superset/.local/bin

# Create superset admin user and initialize the database
RUN fabmanager create-admin --app superset --username 'admin' \
  --firstname 'Superset' --lastname 'Admin' \
  --email 'arho.virkki@finbb.fi' \
  --password 'SuperSecret18'

RUN superset db upgrade
RUN superset init

ENTRYPOINT ["/home/superset/.local/bin/gunicorn", \
  "--bind", "0.0.0.0:8088", "superset:app"]

EXPOSE 8088
```

Build, run and test

```
docker build -t superset:0.27 .
docker run -d --rm -p 8088:8088 --name superset_v0_27 superset:0.27

firefox http://localhost:8088
```

Setting Up VNC Remote Desktops

Document Author: arho.virkki@tyks.fi

Ubuntu Server

Install VNC server with the newer protocol (there are several vnc-server packages in the Ubuntu repositories):

```
sudo apt-get install vnc4server
```

After installation, the command *vncserver* points to *vnc4server* through */etc/alternatives*. Launching *vncserver* first time will ask for setting a password (max. 8 characters). The password can be changed later with *vnc4passwd*.

The following command disables the passwords and sets the X server geometry

```
vncserver -depth 24 -geometry 1280x800  -SecurityTypes None
```

If the X server display number is not given as an argument (for example, *vncserver :10*), the server will choose the next available display from the list 1,2,3,... Once started, the server listens to port 5900 + x, where x is the display number. For example, the default server instance running on display :1 is listening port 5901.

The different server can be stopped individually with

```
vncserver -kill :<display_number>
```

and all at once with

```
killall Xvnc4
```

If vncserver scripts complains about finnish locale setting, add *export LC_ALL=en_US.UTF-8* line to *.bashrc*. This does not affect key mappings.

Then, install the necessary libraries to run *xfc4*, which is a lightweitght desktop environment suitable for remote connections.

```
sudo apt-get install xfce4 xfce4-goodies
```

Then edit the VNC *xstartup* script to choose the windows manager and additional startup programs and settings for the remote X session.

```
nano ~/.vnc/xstartup
```

Example contents:

```
#!/bin/sh

[ -x /etc/vnc/xstartup ] && exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] && xrdb $HOME/.Xresources

# Start a vnc helper application and then an XFCE4 session
vncconfig -iconic &
startxfce4

# Close the vnc server automatically (for this display)
# if the user logs out from XFCE4
vncserver -kill $DISPLAY
```

Securing the VNC connection

Since the vnc connection is not encrypted, it is better to allow only local connections with the *-localhost* option,

```
vncserver -depth 24 -geometry 1280x800 -localhost -SecurityTypes VncAuth
```

and the use SSH port forwarding to securely connect to the vnc port 5900 + <display number>. The following establishes a connection to the remote server *woodpecker.intra.net* with port forwarding

```
ssh -L 5901:localhost:5901 woodpecker.intra.net
```

Now, we can point the vnc client to localhost with

```
vncviewer localhost:1
```

and the connection will be encrypted. Using *-SecurityTypes VncAuth* instead of *-SecurityTypes None* prevents other privileged users from accessing the session (unless they know the VNC password).

Ubuntu Desktop

Since Ubuntu Desktop is simply Ubuntu Server with X (with different packages installed by default, but is available through the same repositories), we do not need to download that many X dependencies. The procedure is similar to Ubuntu Server, but in this case we proceed with gnome-panel and metacity.

```
sudo apt-get install vnc4server gnome-panel metacity
```

Edit the *~/.vnc/xstartup* file as follows

```
#!/bin/sh

[ -x /etc/vnc/xstartup ] && exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] && xrdb $HOME/.Xresources
xsetroot -solid darkgrey
metacity &
unity-settings-daemon &
gnome-panel &
gnome-terminal &
```

The background color can be changed with the *xsetroot -solid <colorname>* command on the terminal, and set permanently by editing the above config. The colors names are same than in CSS, see e.g. http://davidbau.com/colors/. s

Due to the missing 3D support, some features of Gnome cannot be used. For example, *unity-control-center* does not launch. Nonetheless, we can change the Desktop theme from command line.

To switch from the default dark (Ambiance) theme to the optional light (Radiance) theme in Terminal, issue

```
gsettings set org.gnome.desktop.interface gtk-theme Radiance
gsettings set org.gnome.desktop.wm.preferences theme Radiance
```

And to restore the default

```
gsettings set org.gnome.desktop.interface gtk-theme Ambiance
gsettings set org.gnome.desktop.wm.preferences theme Ambiance
```

Set desktop background image

Install appropriate tool

```
sudo apt-get install feh
```

The, add a line to *~/.vnc/xstartup* similar to

```
feh --bg-fill /usr/share/backgrounds/Sea_Fury_by_Ian_Worrall.jpg
```

More options can be found from *feh* manual pages.

References

https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-vnc-on-ubuntu-14-04
http://www.techradar.com/news/software/operating-systems/10-of-the-best-linux-window-managers-90922

CentOS 7

```
sudo su -
yum install tigervnc-server
cp /lib/systemd/system/vncserver@.service /etc/systemd/system/vncserver@.service
nano  /etc/systemd/system/vncserver@.service
```

Edit the file as follows, replasing <USER> with an actual user and change geometry if needed.

```
[Service]
...
ExecStart=/sbin/runuser -l arho -c "/usr/bin/vncserver %i -geometry 1440x900 -SecurityTypes None"
PIDFile=/home/arho/.vnc/%H%i.pid
```

Then, reload the daemon settings

```
systemctl daemon-reload
```

Now, exit from root and issue, as an ordinary user,

```
vncpasswd
```

Then, back to root priviliges

```
sudo su -
systemctl start vncserver@:1.service
systemctl enable vncserver@:1.service
```

If you accidentally crippled your VNC session (e.g. by logging out or closed the clipboard extension client), the service can be restarted with

```
sudo systemctl restart vncserver@:1
```

Open Firewall for VNC

Use ssh to log into the system with an X-server enabled computer (e.g. Mac OS X or Ubuntu Linux) and issue `firewall-config`. Then choose: Zone: public, Configuration: Permanent, enable Service *vnc-server*, and reload Firewalld from the Options -menu.

The other option is to enable the firewall from command line

```
sudo firewall-cmd --permanent --add-service vnc-server
sudo systemctl restart firewalld.service
```

Switch Gnome shell style

The user can switch from Gnome Classic to Gnome by logging out and selecting Gnome from the Session list on the login screen. To switch from Gnome Classic to Gnome from within the user session, run the following command:

```
gnome-shell --mode=user -r &
```

To switch back to classic, issue

```
gnome-shell --mode=classic -r &
```

I have not yet found a way to make this permanent. One option is to introduce an alias in *.bashrc*

```
alias go_gnome='gnome-shell --mode=user -r &'
alias go_classic='gnome-shell --mode=classic -r &'
```

And run that every time when the vncserver needs to be restarted (which is not often).

References: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/System_Administrators_Guide/ch-TigerVNC.html

Connecting a VNC Desktop

Connecting from Windows

Real VNC Viewer: The Real VNC Viewer is free to use. Server can be connected with simplhy the name or IP and display number

```
192.168.1.220:1
```

Virt-Viewer: The Windows version of virt-viewer differs from the Linux software with the same name. Open the softare and connect with

```
vnc://remotebox.vtt.fi:5901
```

where the machine name or IP is followed by the VNC server port (now 5901 = 5900 + 1 for the first display).

Connecting from Linux

Remmina is one of the best remote desktop viewers for linux (http://remmina.sourceforge.net/). For Ubuntu, it is available from software center.

Nonetheless, also Real VNC Viewer is available from https://www.realvnc.com/download/viewer/

There are also command line clients available (that can also do ssh-tunneling directly):

```
xvncviewer -via arho@bluebird.vtt.fi 192.168.1.220:1
xtightvncviewer -via arho@bluebird.vtt.fi 192.168.1.220:1
```

Securing the connection with ssh

The ssh port forwarding can be done with with the command line *ssh* or with Putty in Windows.

From command line, issue

```
ssh -L 5901:localhost:5901 bluebird.vtt.fi
```

after which the connection is done to *localhost*, e.g. *vnc://localhost:5901*.

X2Go Setup

Document Author: arho.virkki@tyks.fi

http://wiki.x2go.org/doku.php/doc:installation:x2goserver https://en.wikipedia.org/wiki/Comparison_of_remote_desktop_software

Client setup, Windows

Download the installer from http://wiki.x2go.org/doku.php and install the client application.

Client setup, Ubuntu

```
sudo apt-get install x2goclient
```

Server side installation, CentOS 7

http://wiki.x2go.org/doku.php/doc:installation:x2goserver

Ensure that EPEL is installed and issue

```
sudo yum install x2goserver
sudo systemctl start x2gocleansessions.service
```

Gnome 3 is not supported. Install eihter XFCE4 or MATE (Gnome 2 fork): http://jensd.be/125/linux/rhel/install-mate-or-xfce-on-centos-7

XFCE:

```
sudo yum groupinstall xfce
```

Server side installation, Ubuntu 14.04

```
sudo add-apt-repository ppa:x2go/stable
sudo apt-get update
sudo apt-get install x2goserver x2goserver-xsession
```

Unity desktop is not supported. Install either XFCE4 desktop

```
sudo apt-get install xfce4 xfce4-session xfce4-terminal
```

To make a full-blown Xubuntu installation (with all bells and whistles), issue

```
sudo apt-get install xubuntu-desktop
```

The other option is to install MATE (Gnome 2 fork): http://www.omgubuntu.co.uk/2014/08/install-mate-desktop-ubuntu-14-04-lts

```
sudo apt-add-repository ppa:ubuntu-mate-dev/ppa
sudo apt-add-repository ppa:ubuntu-mate-dev/trusty-mate
sudo apt-get update && sudo apt-get upgrade
sudo apt-get install --no-install-recommends ubuntu-mate-core ubuntu-mate-desktop
```

Check the service status

```
sudo service x2goserver status
```

Server side installation, Xubuntu and Ubuntu Mate 16.04

The installation follows the previous procedure

```
sudo add-apt-repository ppa:x2go/stable
sudo apt-get update
sudo apt-get install x2goserver x2goserver-xsession
sudo service x2goserver status
```

At this writing, Xubuntu and Ubuntu Mate 16.04 need one extra tweak (see. http://askubuntu.com/questions/763597/x2go-with-ubuntu-mate-xfce-16-04-fails-to-start). For xfce4, we need the following environment variables:

```
sudo vim /etc/profile.d/x2go_xfce4.sh

# Manually add some paths for x2go
export GSETTINGS_SCHEMA_DIR=/usr/share/xfce4:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
export XDG_DATA_DIRS=/usr/share/xfce4:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
```

The procedure is similar to Ubuntu Mate:

```
sudo vim /etc/profile.d/x2go_mate.sh

# Manually add some paths for x2go
export GSETTINGS_SCHEMA_DIR=/usr/share/mate:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
export XDG_DATA_DIRS=/usr/share/mate:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
```

Customize shortcut keys

http://wiki.x2go.org/doku.php/wiki:advanced:nx-keyboard-shortcuts

On the server side (i.e. at moondust), issue

```
sudo vi /etc/x2go/keystrokes.cfg
```

And change

```
<keystroke action="close_session" Control="1" AltMeta="1" key="t" />
```

into

```
<keystroke action="close_session" Control="1" AltMeta="1" key="0" />
```

To change session termination key from <Ctrl> + <Alt> + t (which is default for opening Terminal application in Ubuntu) into <Ctrl> + <Alt> + 0, which is not already reserved. Finally, log out from the current x2go-session to reload the new settings.

Backup System

Document Author: arho.virkki@tyks.fi

Automated Backups

Wiki, version control system, raw data, PostgreSQL data base and several other localtions are automatically backed up to NAS at */nas/backup*.The cron scheduler at *aku@moondust* is responsible for running the backup scrtips. For details, see `crontab -l`.

Backup scripts

The backup script are stored in *Common.git* repository and located at *Common/backup/*. Currently, the scripts include several shell and R scripts.

```
.
 data
ăă  backup_git.R
ăă  backup_raw_data.R
 KVM
ăă  backup_all_domains.sh
ăă  backup_quickboot_domain.sh
ăă  backup_reboot_domain.sh
ăă  virsh_shutdown_domain.sh
 postgres
     backup_pg.sh
```

Backing up PostgreSQL

There are several possibilites of making a full backup of a running PostgreSQL cluster. We use now Option 1 below since it was fastest.

Option 1: Run pg_dumpall at moondust.intra.net

First, install a matching version of the PostgreSQL *pg_dumpall* tool. (In this case postgresql94 - PostgreSQL client programs and libraries: http://yum.postgresql.org/9.4/redhat/rhel-7-x86_64/repoview/postgresql94.html) The development libraries are not strictly necessary (but are needed for R support, if even needed).

```
wget http://yum.postgresql.org/9.4/redhat/rhel-7-x86_64/postgresql94-9.4.8-1PGDG.rhel7.x86_64.rpm
wget
    http://yum.postgresql.org/9.4/redhat/rhel-7-x86_64/postgresql94-libs-9.4.8-1PGDG.rhel7.x86_64.rpm
wget
    http://yum.postgresql.org/9.4/redhat/rhel-7-x86_64/postgresql94-devel-9.4.8-1PGDG.rhel7.x86_64.rpm
```

Install the packages with

```
sudo rpm -ivh postgresql94-*
```

Ensure that the password for the PostgreSQL root user is set at the database machine. Then, create a *.pgpass* file under the *aku* user home directory. The generic format for the file is

```
hostname:port:database:username:password
```

In this case, we allow *postgres* user to access all databases (*) with

```
echo "dbdev.intra.net:5432:*:postgres:<passwd here>" > .pgpass
chmod og-rwx .pgpass
chmod 0600 .pgpass
```

where the password is set to *aku* (normal short) password. Try that the arrangement works with

```
psql -U postgres -h dbdev.intra.net -d postgres
```

Then, make a copy of the database with

```
today=$(date --iso-8601)
mkdir -p  "/var/local/backup/dbdev/$today"

time /usr/pgsql-9.4/bin/pg_dumpall \
-w -h dbdev.intra.net -U postgres -l postgres | lz4 | \
split -a 2 -b 1G - "/var/local/backup/dbdev/$today/pgdump.lz4_"
```

The execution took about 2 hours and 36 minutes as of this writing (2016-08-03)

```
real    156m44.363s
user    48m14.420s
sys     26m17.773s
```

Option 2: Run at moondust but call pg_dumpall at dbdev

Copy the public key of aku@moondust to aku@dbdev

```
scp .ssh/id_rsa.pub aku@dbdev.intra.net
```

Then, add the key to the postgres user

```
sudo sh -c "cat id_rsa.pub >> ~postgres/.ssh/authorized_keys"
```

Now we can backup the whole database from aku@moondust with the following script:

```
#!/bin/bash

# PostgreSQL cluster backup script

# Author(s) : Arho Virkki
# Copyright : VTT Technical Reseach Centre of Finland
# Date      : 2016-07-29

today=$(date --iso-8601)
mkdir -p  "/var/local/backup/dbdev/$today"

time ssh postgres@dbdev.intra.net "pg_dumpall | lz4 " | \
split -b 1G - "/var/local/backup/dbdev/$today/pgdump.lz4_"
```

The execution took about three hours (2016-08-03)

```
[aku@moondust postgres]$ ./backup_pg.sh

real    185m44.510s
user    5m32.734s
sys     4m3.767s
```

Option 3: Run the backup at dbdev.intra.net

aku@dbdev:~$ sudo apt-get install liblz4-tool

Allow

```
sudo su - postgres
ssh-keygen
ssh-copy-id -i .ssh/id_rsa.pub aku@moondust.intra.net
```

The execution took about 3 and half hours (2016-08-03)

```
time sudo -u postgres sh -c \
"pg_dumpall | lz4 | ssh aku@moondust.intra.net \
\"split -a 2 -b 1G - /nas/backup/dbdev/$(date --iso-8601).lz4_\""

real    211m47.591s
user    28m51.380s
sys     15m27.717s
```

Backing up KVM virtual machines

Shut down for maintenance

While it is possible to back up live machines with snapshots, it is safest to power off the machine to ensure a consistent state of the virtual disk. Otherwise, we need to make sure that e.g. no database transactions are running while the snapshot was taken for the backup.

Run the backup script

The backup scripts reside on the *Common* repository under `backup/KVM`, and an instance of *Common* should be found at `aku@moondust.intra.net:/home/ktp/Common`. There are also symbolic links at `/usr/bin` for *sudo* access.

Examples:

```
sudo backup_reboot_domain.sh gitbox
time sudo backup_reboot_domain.sh gitbox pbzip2
```

Typical output:

```
[aku@moondust images]$ time sudo backup_reboot_domain.sh ktptest pbzip2
Waiting for ktptest to shut off..
Backing up ktptest into
/nas/backup/images/2016-01-15_ktptest.xml
/nas/backup/images/2016-01-15_ktptest.qcow2.tar.bz2
Backup done
Domain ktptest started

real    8m6.144s
user    56m55.455s
sys 5m38.021s
```

Ensure that the machine responds to ACPI poweroff

On Ubuntu hosts, save the original script which responds to power button

```
cd /etc/acpi/
sudo mv powerbtn.sh powerbtn_orig.sh
```

and edit the *powerbtn.sh* to only contain the following line

```
#!/bin/shs
/sbin/poweroff
```

to disable any user interactivity required for poweroff (such as the "Would you like to…" prompts).

Why the backup is slow?

A "recent" discussion at *comp.unix.internals* (1990) explains that "…you cannot tell the difference between a hole and an equivalent number of nulls without reading raw blocks…". Hence tar needs to read the whole file, since it is a file-system independent tool (xfs, ext2/3/4, ntfs and nfs all work). For details, see: http://www.delorie.com/gnu/docs/tar/tar_118.html

Appendix A: Tar Performance with Different Compression Levels

For details, see e.g.

- http://www.gnu.org/software/tar/manual/tar.pdf
- http://serverfault.com/questions/66338/how-do-you-synchronise-huge-sparse-files-vm-disk-images-be

Tar with no compression

```
time tar -cSf /nas/backup/images/`date --iso-8601`_ktptest.qcow2.tar \
-C /var/lib/libvirt/images/ ktptest.qcow2

real    9m48.137s
user    2m5.844s
sys     4m35.557s

time tar -xvSf /nas/backup/images/2016-01-14_ktptest.qcow2.tar
ktptest.qcow2

real    2m42.673s
user    0m1.598s
sys     0m43.269s
```

Tar with gzip (I/O bound)

```
time tar -cSzf /nas/backup/images/`date --iso-8601`_ktptest.qcow2.tar.gz \
-C /var/lib/libvirt/images/ ktptest.qcow2

real    17m35.627s
user    13m28.720s
sys     4m21.880s

time tar -xvzf /nas/backup/images/2016-01-14_ktptest.qcow2.tar.gz

real    2m48.644s
user    2m9.246s
sys     1m3.905s
```

Tar with lz4

```
time tar -I lz4 \
-cSf /nas/backup/images/`date --iso-8601`_ktptest.qcow2.tar.lz4 \
-C /var/lib/libvirt/images/ ktptest.qcow2

real    8m8.091s
user    3m3.050s
sys     4m1.563s

time tar -I lz4 -xvf /nas/backup/images/2016-01-14_ktptest.qcow2.tar.lz4

real    2m44.864s
user    0m18.914s
sys     1m12.399s
```

Tar with pbzip2

```
time tar -I pbzip2 \
-cSf /nas/backup/images/`date --iso-8601`_ktptest.qcow2.tar.bz2 \
-C /var/lib/libvirt/images/ ktptest.qcow2

real    8m1.982s
user    54m55.448s
sys     5m28.025s

time tar -I pbzip2 -xvf /nas/backup/images/2016-01-14_ktptest.qcow2.tar.bz2

real    1m42.990s
user    15m34.306s
sys     1m52.933s
```

File size comparison

```
[arho@moondust images]$ ls -lha
-rw-r--r-- 1 arho wheel  14G 14.1. 16:04 2016-01-14_ktptest.qcow2.tar
-rw-r--r-- 1 arho wheel 5,7G 14.1. 22:45 2016-01-14_ktptest.qcow2.tar.bz2
-rw-r--r-- 1 arho wheel 6,0G 14.1. 22:24 2016-01-14_ktptest.qcow2.tar.gz
-rw-r--r-- 1 arho wheel 7,9G 14.1. 21:59 2016-01-14_ktptest.qcow2.tar.lz4
```

Appendix B: Typical Execution Times

Initial sized of the images

```
[aku@moondust images]$ ls -lhs
total 929G
 59G -rw-r--r-- 1 qemu qemu  2.1T Jan 16 11:17 woodpecker.qcow2
 17G -rw-r--r-- 1 qemu qemu  513G Jan 16 11:44 parrot.qcow2
 44G -rw-r--r-- 1 qemu qemu  2.1T Jan 16 11:17 gitbox.qcow2
523G -rw-r--r-- 1 qemu qemu   11T Jan 16 11:45 ktphadoop.qcow2
270G -rw-r--r-- 1 qemu qemu   11T Jan 16 11:17 dbdev.qcow2
 19G -rw-r--r-- 1 root root  257G Jan 15 15:01 ktptest.qcow2
```

Corresponding execution times

```
[aku@moondust KVM]$ sudo ./backup_all_domains.sh
This will take long, and automatically reboot all virtual
machines along the way!
Are you sure [y/n]: y
Backing up parrot into
/nas/backup/images/2016-01-15_parrot.xml
/nas/backup/images/2016-01-15_parrot.qcow2.tar.lz4
Backup done
Domain parrot started

real    14m28.701s
user    4m51.211s
sys     8m5.817s

Waiting for gitbox to shut off..
Backing up gitbox into
/nas/backup/images/2016-01-15_gitbox.xml
/nas/backup/images/2016-01-15_gitbox.qcow2.tar.lz4
Backup done
Domain gitbox started

real    49m24.771s
user    16m55.121s
sys     29m56.354s

Waiting for woodpecker to shut off...
Backing up woodpecker into
/nas/backup/images/2016-01-15_woodpecker.xml
/nas/backup/images/2016-01-15_woodpecker.qcow2.tar.lz4
Backup done
Domain woodpecker started

real    55m36.373s
user    18m6.019s
sys     31m46.295s

Waiting for dbdev to shut off...
Backing up dbdev into
/nas/backup/images/2016-01-15_dbdev.xml
/nas/backup/images/2016-01-15_dbdev.qcow2.tar.lz4
Backup done
Domain dbdev started

real    262m9.829s
user    89m34.183s
sys     160m33.088s

Waiting for ktphadoop to shut off...
ing up ktphadoop into
/backup/images/2016-01-16_ktphadoop.xml
/backup/images/2016-01-16_ktphadoop.qcow2.tar.lz4
Backup done
Domain ktphadoop started

    302m29.830s
    98m37.440s
    174m44.484s
```

Common tasks with SQL

Document author: arho.virkki@tyks.fi

The following examples illustrate both the common task and differences in SQL dialects between Oracle, PostgreSQL and Hadoop HQL.

PostgreSQL

Connecting

Use either pgAdmin III, Squirrel SQL Client or psql with the following command

```
psql -U ktp -h dbdev.intra.net
```

Change the database connection with psql

```
\connect <database_name>
```

Examples

Add primary key:

```
ALTER TABLE stage_oberon.diagnoosi ADD CONSTRAINT diagnoosi_pkey PRIMARY KEY ( dgn_numero );
```

List all tables in the stage schema:

```
\dt stage.*
```

List all tables starting with 'OSASTO'

```
\dt OSASTO*
```

Regexp matching and substitutions

Replace all newlines and tabs

```
select regexp_replace(regexp_replace(<column_name>, '\t', '|t|', 'g'), '\n', '|n|', 'g')
from <table_name>;
```

The last parameter 'g' means replace all matching substrings, not just the first one.

Hive SQL

Regexp_replace

```
SELECT regexp_replace(xml, '\r|\n','<br>') as xml,
edhreceiveddate
FROM loaddbqpati.qpativsshpraw_parquet
```

Oracle

Connecting

Use Squirrel SQL Client

Examples

Use date as a literal:

```
SELECT *
FROM "MDODS"."DIAGNOOSI_ODS"
WHERE potilasnumero = '597407000031511857'
AND paivityshetki_ods = TO_DATE( '2016-01-27 08:44:50', 'yyyy-mm-dd hh24:mi:ss' )
```

Hadoop

Connecting

Use either Hue on http://ktphadoop.intra.net:8888 or connect at command line with

```
ssh hive@ktphadoop "beeline -u jdbc:hive2://ktphadoop.intra.net:10000/ktp"
```

Examples

Creating a sub-table for testing ETL:

```
create table lab_part as select * from lab limit 100;
```

R Language

Document Author: arho.virkki@tyks.fi

Miscellaneous R scripts and tips

Installation and first steps

For the server environment, see the installation instructions for R Language. For a personal installation (e.g. for a laptop), download R from https://www.r-project.org/ and follow the instructions. If you have already installed R, Familiarize yourself with the original, official open source manuals:

- https://cran.r-project.org/manuals.html
- https://cran.r-project.org/doc/manuals/r-release/R-intro.html

Brief history

R is based on S-language, which was published in 1976, and t he first version of R appeared in 1993. The source code is under the GNU-license (like the Linux kernel), which means that the language is free and can be used in commercial settings, but not embedded in closed source products. This must just be taken into account, but it does not impose any practical limitations of R use.

There are version for Windows, OS X and Linux systems available at the R web site. R can be extended with custom functions (R scripts), packages from the comprehensive R archive network (CRAN), o wn custom written packages, and C/C++ or Java code, to name a few alternatives. For details, see: https://en.wikipedia.org/wiki/S_(programming_language)

Editors

RStudio R editor, or Rstudio Server
http://www.rstudio.org/

Notepad++, a must have editor for Windows
http://notepad-plus-plus.org/

Eclipse with StatET plug-in
http://www.eclipse.org/
http://www.walware.de/goto/statet/

Emacs with ESS (Emacs Speaks Statistics) plug-in
http://vgoulet.act.ulaval.ca/en/emacs/
http://ess.r-project.org/

Vim with optional Nvim-R extension
http://www.vim.org/download.php/
http://macvim-dev.github.io/macvim/
http://www.vim.org/scripts/script.php?script_id=2628

RStudio is the easiest to begin with, whereas Vim is hard to master, but really powerful.

rmarkdown templates

```
install.packages("rmarkdown")
```

Save templates into the rmarkdown folder. This may be, depending on your setup, e.g. either of the following (Windows):

```
C:/R-3.2.3/library/rmarkdown/rmarkdown/templates/
```

or (Linux:)

```
~/.RLibrary/rmarkdown/rmarkdown/templates/
```

Vim configuration

Example Vim configuration file for R (`~/.vimrc`);

```
" Configure indentation
set shiftwidth=2
set tabstop=2
" Expand all tabs into spaces
set expandtab

" Configure indentation to work with R
set nocompatible
syntax enable
filetype plugin on
filetype indent on

if has("gui_running")
  set lines=50 columns=80   " Set geometry
  set guioptions-=T         " Remove toolbar
  set guioptions-=r         " Remove right-hand scroll bar
  set guioptions-=L         " Remove left-hand scroll bar
  set guifont=Monospace\ 11 " Set font
endif
```

Pentaho Data Integration (= PDI = Kettle)

Document authors: arho.virkki@tyks.fi and anna.hammais@tyks.fi

You can install a copy of PDI on your local workstation or your local Biobank server or use the copy installed on FinBB stakeholder. For convenience, it is simplest to install PDI inside the network that houses the servers that you most often connect to. In biobank cases, it is recommended that biobanks install PDI locally to allow easy access to local databases and connect to stakeholder through an SSH tunnel.

PDI is already installed in stakeholder, where you can access the local PostgreSQL database without SSH tunnels by referring to localhost:5432 (server:port).

- [PDI setup and database access] (pentaho-setup)

ETL - Extract, transform and load data from source to target

- Reading data from a source and writing to a target. Sources and targets can be different file types, different relational databases, Hadoop, REST services, etc.

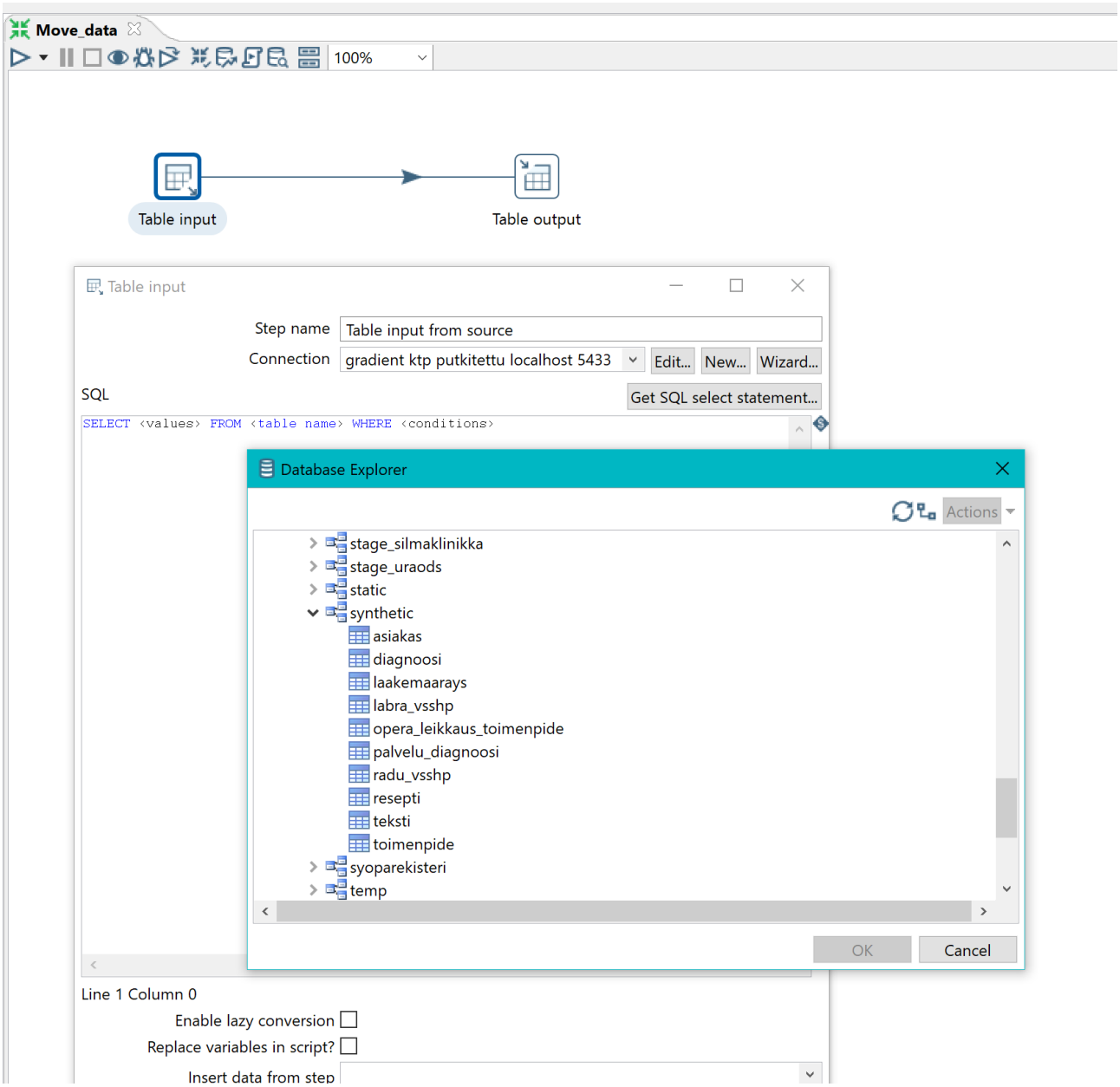


Figure Table input step. You can browse through the schemas and tables in the source database by clicking the *Get SQL select statement...* button in Table input.

- Transforming data: changing timestamp formats, replacing problematic encodings or characters, regular expression matching, adding or removing columns.
- Joining data from different sources into one process

PDI basic terminology

- transformation: A process that does operations on a group of data rows. Rows are processed in parallel.
- job: A process that usually contains of several steps that are run in sequence, not in parallel.
- step and hop: both transformations and jobs consist of *steps* that are connected in sequence by *hops*

Create transformations or jobs

- Select New –> Transformation/Job
- search for step names by typing in the left panel search box
- drag steps from left panel to main panel
- shift-drag to create hop between steps
- Job always starts with the Start step, Transformation starts with one or more data input steps that start running in parallel
- In a Transformation, rows flow from one step to the next. If it's necessary to relay rows from one transformation to another, this can be done with the following steps: Copy rows to result and Get rows from result

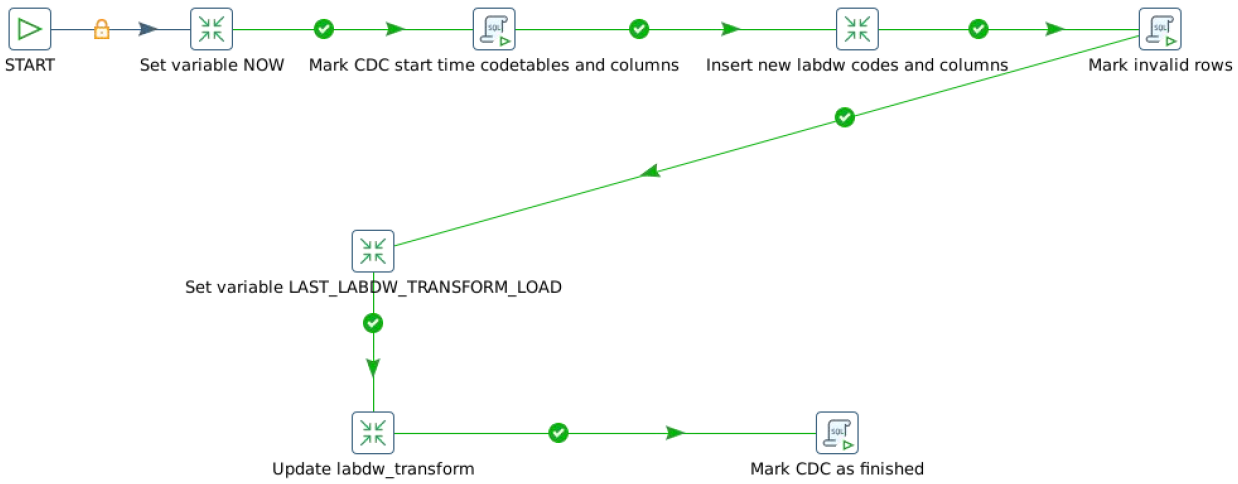


Figure Example of a Job in PDI. Green check mark symbol in a Hop means that the foillowing step will be run only if the previous step was successful. The lock sybol means that the following step will be run no matter what the result of the previous step was.

Add a database connection to PDI

In order to add a database connection, you need to download the JDBC driver for the database engine that you are connecting to. Pentaho maintains a list of download locations for the most common drivers here: https://help.pentaho.com/Documentation/8.0/Setup/JDBC_Drivers_Reference.

Save the driver jar file into the following location: yourPentahoFolder/data-integration/lib, (for example C:/Users/anna/Downloads/pdi-ce-7.1.0.0-12/data-integration/lib).

For connecting to stakeholder PostgreSQL, you need to dowload the at least the PostgreSQL driver at https://jdbc.postgresql.org/download.html.

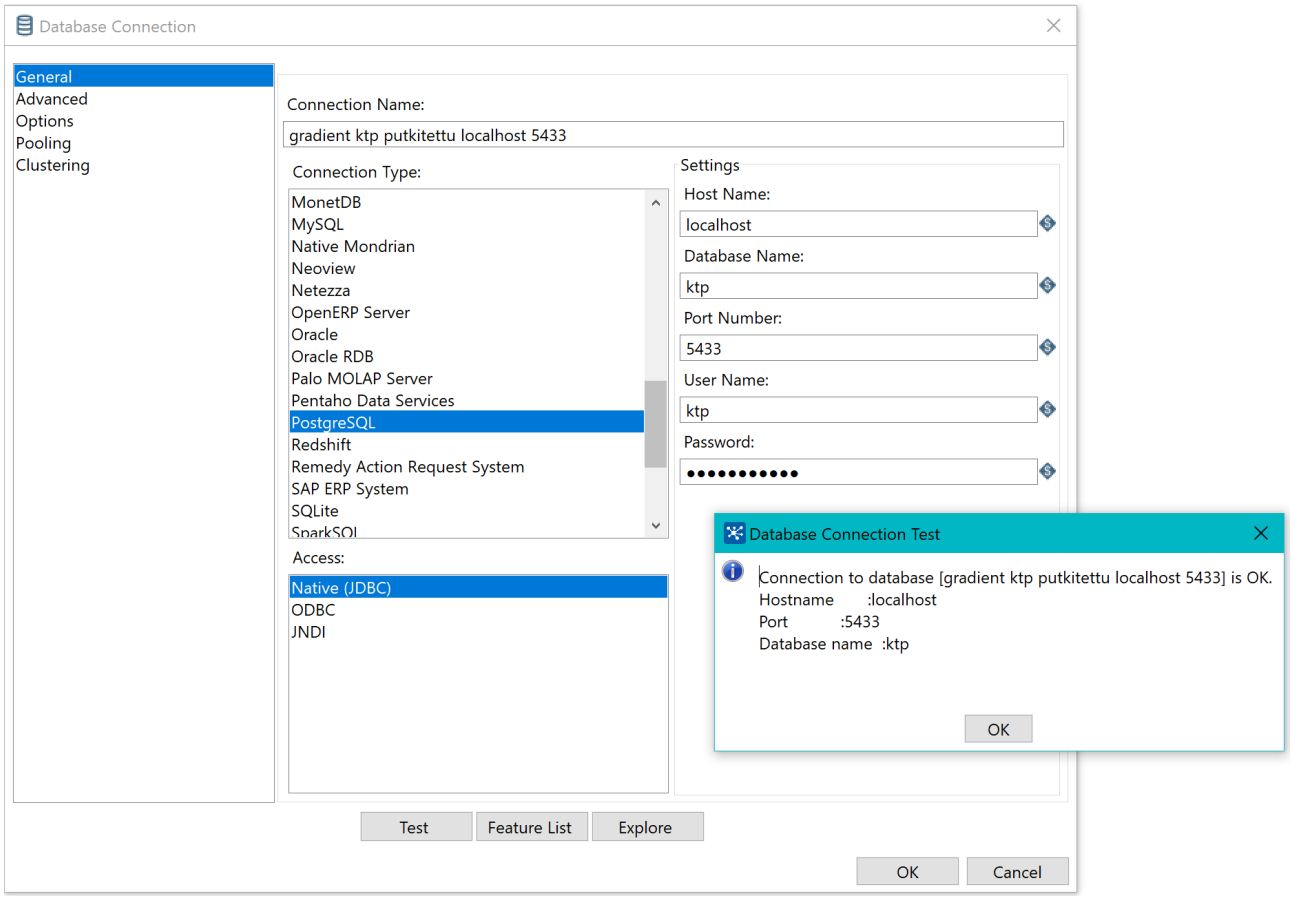


Figure Example of database connection in PDI. In this case, the connection is made through an SSH tunnel, and therefore the hostname is *localhost*. In cases where you're connecting to a server in your local network, write the server name it its usual form (of the type myserver.mynetwork.net).

You can add connections for all the databases that you need to read from or write to. If they are used through an SSH tunnel, remember to have the tunnel open also when your transformation is ready and you run it.

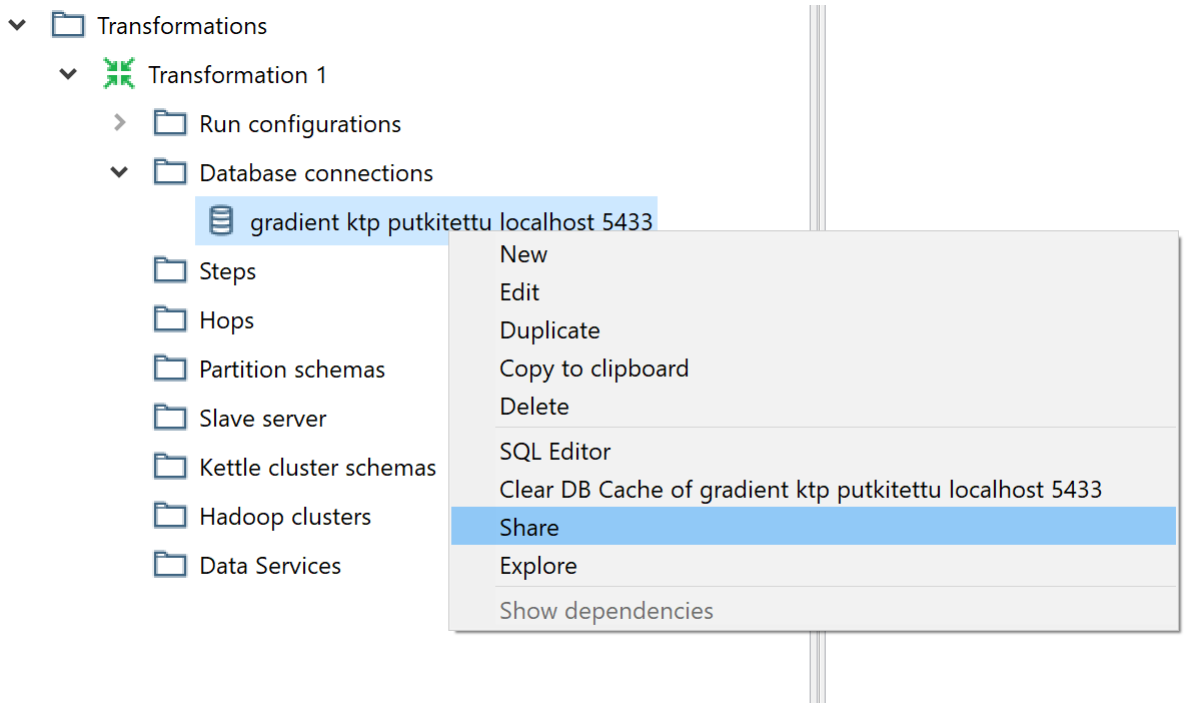


Figure In order to be able to use the same database connection in other transformations in the future, remember to *share* the connection.

Available steps and functionality

Documentation for all PDI steps can be found at https://wiki.pentaho.com/display/EAI/Pentaho+Data+Integration+Steps.

Input steps

- Table input: Read from database table
- Get data from XML: parse specific data fragments from XML file using XPath syntax
- Get system info: Get e.g. current system time (to add timestamp to each row)
- Get file names: get a list of file names by folder (to process each file in a loop)
- Text file input: Read from text file (tab/space/comma/etc-separated)
- Excel input: Read from excel file

Output steps

- Table output: Write (= insert) to database table. (Un-check the *Use batch update for inserts* options to avoid unnecessary complications.)
- Insert/output: Write (insert or update) to a database table based on whether the corresponding row already exists or not
- Text file output: write to text file (tab/space/comma/etc-separated)
- Excel file output: write to Excel file

Data processing

- Formula (Excel-like formulas): Do calculations on row values
- Regexp evaluation: Evaluate wheter value on row matches a regular expression, or separate matching substring from value (capture groups)
- Add constants: adds a column with a constant value to each row
- Concat fields: does string concatenation on columns

- Select values: Drop certain columns or change column metadata (e.g. string to timestamp)
- Unique rows (HashSet): Works like SQL distinct, i.e. removes duplicates based on defined columns.
- Filter rows: pass only rows that match a condition, or separate groups of rows to process them differently
- Switch/Case: similar to Filter rows, divides rows to different processing streams based on a field value

Logging and utilities

- Write to log: writes out selected fields of the row set to the console; useful for debugging
- Also, when running jobs and transformations, it is sometimes useful for debugging to select a different option instead of Basic logging in the Run popup window. Other options give more information about the rows being read.
- Copy rows to result and Get rows from result: relay rows from one Transformation to the next when they are subsequent steps inside a Job

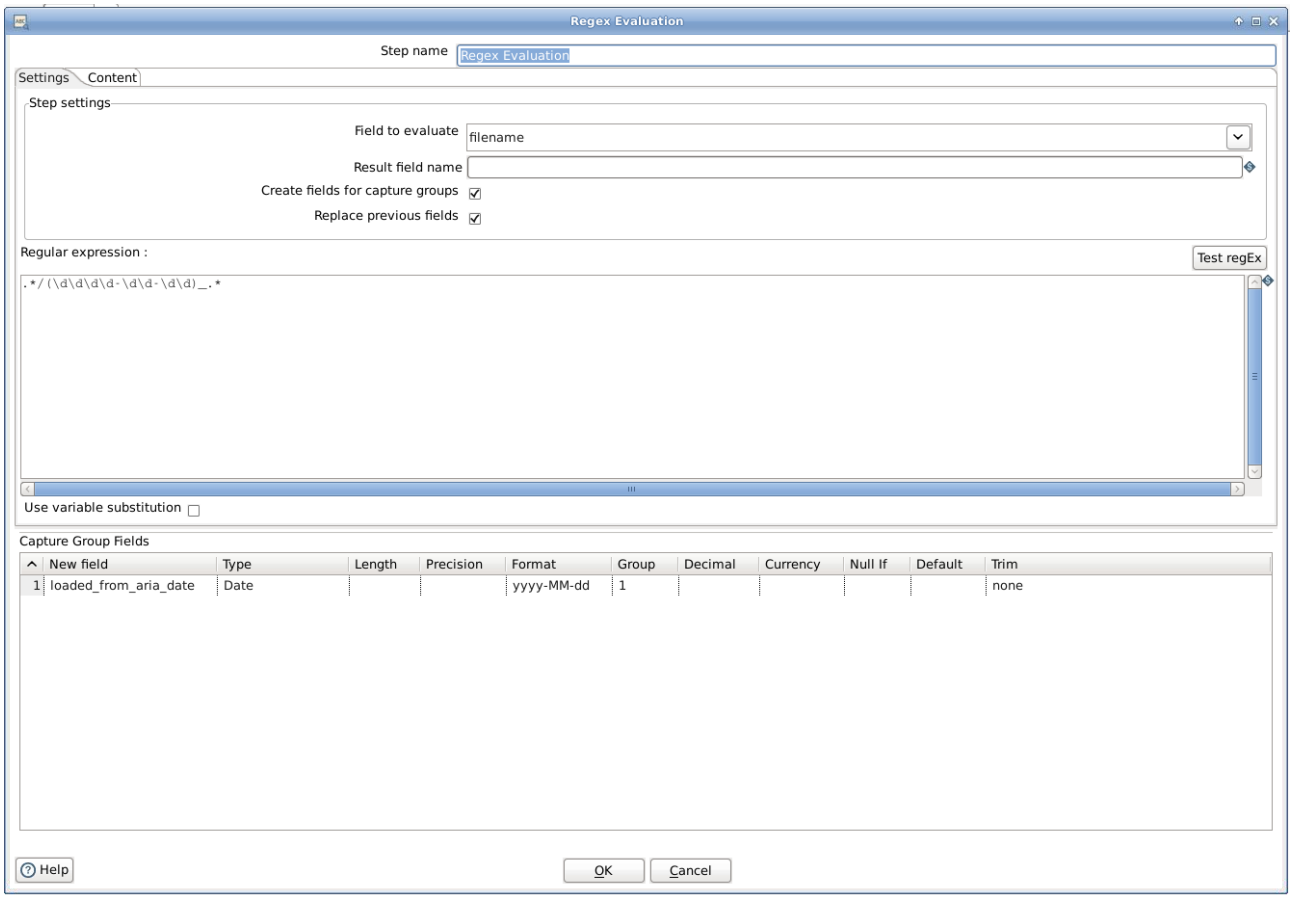


Figure Example of using a regular expression to extract a date from a string

Running Jobs and Transformations

There are different ways to run jobs and transformations.

- Spoon - the graphical user interface. Start with this!
- Kitchen - running jobs from command line after they have been developed in Spoon and tested
- Pan - running transformations from command line after they have been developed in Spoon and tested
- In Linux environments, scheduling jobs (with Kitchen) and transformations (with Pan) is easy with cron

Using parameters

Jobs and transformations can be parametrized, and parameter values are set each time the job/transformation is run. This makes jobs and transformations more generic.

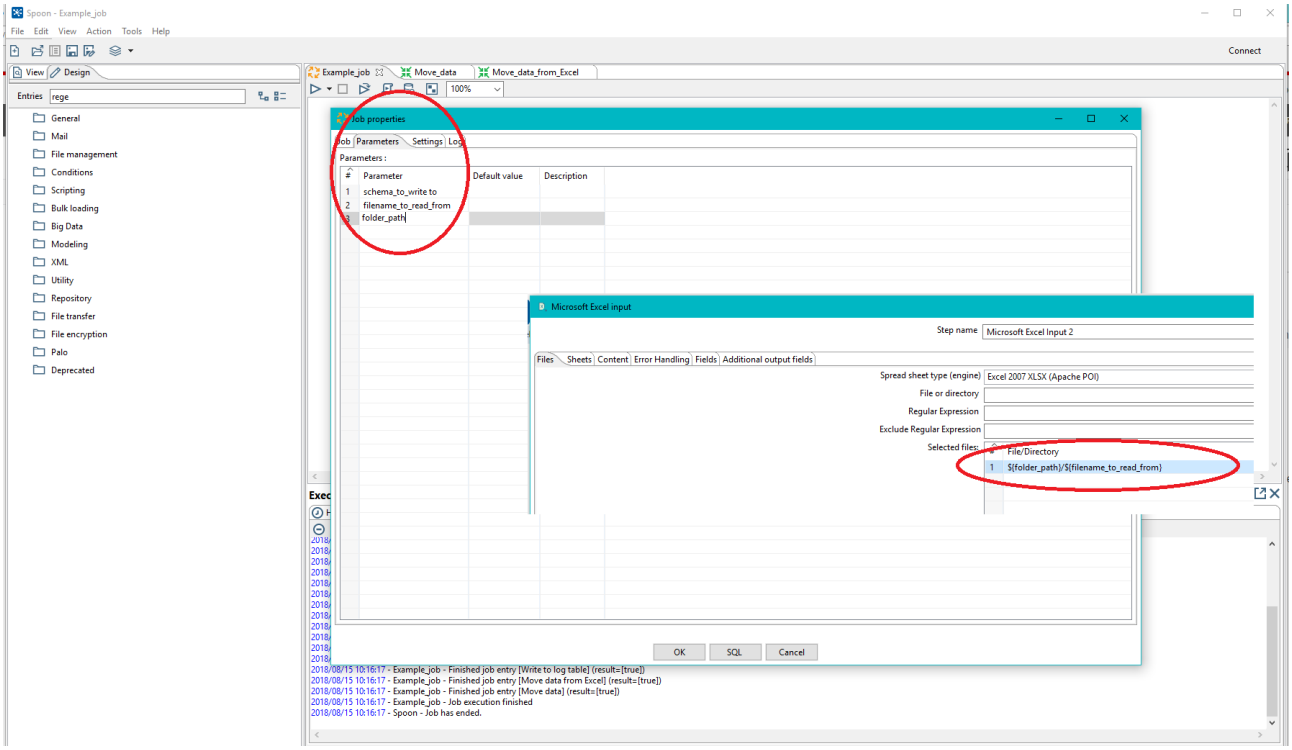


Figure 1:

Learning online

- Transformation tutorial for reading from text file: https://help.pentaho.com/Documentation/7.1/0J0/0C0/020
- Job tutorial: https://help.pentaho.com/Documentation/7.1/0J0/0C0/030

PostgreSQL Server and Database Administration

Document author: anna.hammais@tyks.fi and arho.virkki@tyks.fi

- PostgreSQL setup at Gradient
- Troubleshooting PostgreSQL at Gradient
- R Language (PL/R) and Shell (PL/sh) extensions
- Logins as specific user
- psql commands
- Manage databases and users
- Processes on PostgreSQL server
- Logging queries
- PostgreSQL server configuration
- Backup and restore
- SQL syntax
- JSON
- Adding auto-increment primary key after table creation
- Generating a date series
- Encrypt and Decrypt data with a symmetric AES key
- Vacuum PostgreSQL database manually
- Postgres_fdw (Foreign Data Wrapper) and dblink
- Setup per-user logging and import a log file into table
- How to hande view dependencies in PostgreSQL

Adding serial primary key to a table after table creation

Document author: anna.hammais@tyks.fi

Create a sequence:

```
create sequence stage_labdw.labdw_transform_id_seq increment by 1;
```

Examine a sequence:

```
select * from stage_labdw.labdw_transform_id_seq;
```

Change a column to auto-increment:

```
alter table stage_labdw.labdw_transform rename to labdw_transform_old;

create table stage_labdw.labdw_transform as
select (row_number() over())::int labdw_transform_id,
l.*
from stage_labdw.labdw_transform_old as l;

select setval('stage_labdw.labdw_transform_id_seq',
(select max(labdw_transform_id) from stage_labdw.labdw_transform;

alter table stage_labdw.labdw_transform alter labdw_transform_id set not null;

alter table stage_labdw.labdw_transform alter labdw_transform_id
set default nextval('stage_labdw.labdw_transform_id_seq'::regclass)

alter table stage_labdw.labdw_transform ADD PRIMARY KEY (labdw_transform_id);
```

Backup and restore

Document author: anna.hammais@tyks.fi and juha-matti.varjonen@tyks.fi

Dump all databases into directory-format:

backup_pg.sh script is stored under Common/scripts/backup/postgres (in the Common Git repository)

This script creates a reperate dump folder for every database from Gradient using directory-format of pg_dump.

```
### create target data database
psql -h <host_name> -p 5432 -U postgres --command "CREATE DATABASE <database_name>;"

### restore globals (including user accounts etc.)
psql -h <host_name> -p 5432 -U postgres -d <database_name> < globals_only.backup

### restore database content
pg_restore -W -j 10 -Fd -h moondust-dev.intra.net -p 5432 -U postgres -d <database_name>
    <database_name>.dir
```

More information about pg_dump can be found from: https://www.postgresql.org/docs/9.6/static/app-pgdump.html

Dumping all tables in a specific schema into a plain-text file:

```
#!/bin/bash

# -n = name of schema
# ktp = name of database
# -f = name of output file
# date -I'date' = current timestamp in ISO format, date part only

pg_dump -n 'luovutusrekisteri' ktp -f '~/$(date -I'date')_moondust_luovutusrekisteri_backup.sql'
```

Restoring tables from the plain-text backup file:

```
psql database_name < backup_file.sql
```

Encrypt and Decrypt data with AES

Document author: juha-matti.varjonen@tyks.fi

Encryption and decryption routines in PostgreSQL aes and ascii

Encrypt from text into byte array

```
SELECT encrypt('secret','pass1','aes');
```

Convert to ascii

```
SELECT encode( encrypt('secret','pass1','aes'), 'base64');

           encode
--------------------------
  YJAGfjZ3NZliNc02TH+q+w==
```

Decode back

```
SELECT convert_from(decrypt( decode('YJAGfjZ3NZliNc02TH+q+w==','base64'), 'pass1', 'aes'),'utf8');
```

In summary:

```
SELECT convert_from(decrypt( decode(
  (SELECT encode( encrypt('tosi salainen juttu','pass1','aes'), 'base64')),
  'base64'), 'pass1', 'aes'),
  'utf8');
```

By using Encrypt fuction func.henkilotunnus_to_pseudonym_aes

```
select * from func.henkilotunnus_to_pseudonym_aes(_henkilotunnus := 'secret',_crypt := TRUE,_salt
    := 'pass1') as pseudonym_aes;
```

By using decrypt function func.pseudonym_to_henkilotunnus_aes

```
select * from func.pseudonym_to_henkilotunnus_aes(_pseudonym := 'YJAGfjZ3NZliNc02TH+q+w==', _salt
    := 'pass1') as henkilotunnus;
```

Install needed extensions

Install from Linux bash:

Note that you have to check the correct version to be installed

```
sudo yum install postgresql10-contrib

or

sudo apt-get install postgresql10-contrib
```

Open superadmin (postgres) connection to ktp database

Run SQL command

```
CREATE EXTENSION pgcrypto;
```

Generating a date series

Document author: anna.hammais@tyks.fi

Simple date series:

```
select
series + date '2003-01-01' as myDate
from generate_series(0,6000) as series;
```

Example query:

```
select md5(a.henkilotunnus),
p.palvelu_numero,
a.syntymaaika_pvm::date as birth,
a.kuolinaika_pvm::date as death,
alkuhetki_pvm::date as ward_start,
alkuhetki_pvm::date + date_series as ward_date,
loppuhetki_pvm::date as ward_end,
vo_toimipiste_koodi as unit_code,
yksikko_nimi as unit_name,
case when alkuhetki_pvm::date <= kuolinaika_pvm::date
  and kuolinaika_pvm::date <= loppuhetki_pvm::date
    then 1
    else 0 end
as died_on_this_ward

from stage_uraods.mv_palvelu as p
inner join stage_uraods.v_asiakas as a on p.henkilotunnus = a.henkilotunnus
, generate_series(0,6000) as date_series
where palvelumuoto = 'osastohoito'
-- include only dates that are between the ward start and end dates
and alkuhetki_pvm::date + date_series <= loppuhetki_pvm::date
and md5(a.henkilotunnus) = 'e1db13203ed29aa030b305745e7f223c'
order by p.palvelu_numero, ward_date;
```

PostgreSQL JSON

Document author: anna.hammais@tyks.fi

JSON data types

- **json**: stored as string, faster input, slower searches
- **jsonb**: stored as binary, faster searches, slower input, indexable

Logging queries in PostgreSQL

Document author: anna.hammais@tyks.fi

Setting PostgreSQL config options for session

```
load 'auto_explain';

set auto_explain.log_min_duration=0;

set auto_explain.log_analyze = true;

set auto_explain.log_nested_statements = ON;
```

Then, run your query in the session. Then, find the log file (on moondust, it's at /srv/pgsql/9.6/data/pg_log/).

Login as specific user

Document author: anna.hammais@tyks.fi, arho.virkki@tyks.fi

First ssh to dbdev.intra.net, then issue

```
psql -U <username> [--password]
```

Login as superuser

```
sudo su - postgres
psql
```

Automated login inside scripts

```
PGPASSWORD=<passwd here> psql -U <usernamer> -h <host> -d <database>
```

For example, list all schemas in ktp database to standard output, then quit

```
echo "\dn" | PGPASSWORD=<passwd here> psql -U ktp -h moondust
```

Manage PostgreSQL databases and users

Document author: anna.hammais@tyks.fi

Create databases

Create database

```
CREATE DATABASE <database_name> WITH OWNER = <role_name>;
```

To simplify privilege management, revoke all rights from the public role in the new database, so that they cannot be inherited by new roles that will be created in the future

```
REVOKE ALL ON DATABASE <database_name> FROM PUBLIC;
REVOKE ALL ON SCHEMA <database_name>.public FROM PUBLIC;
```

Create and manage users/roles

Login as superuser. Create a user:

```
CREATE ROLE <username> WITH LOGIN ENCRYPTED PASSWORD <pwd_in_single_quotes>;
```

The following are equivalent

```
CREATE ROLE username WITH LOGIN;
CREATE USER username;
```

Check users, roles and privileges

```
psql=> \du -- roles and users
psql=> \dp [<schema_name>.* | <table_name>]     -- user privileges for a specific schema or table
```

In effect, the latter only shows the privileges for a table that are not in place by default. For example, the table owner has all privileges by default, and those are not shown here.

Similarly, as a query:

```
select * from pg_roles;
select * from information_schema.role_table_grants where grantee = '<role_name>';
```

Note that the last query only works on tables, not views or materialized views.

Grant privileges

Grant privileges on a specific schema. First, allow the user to list the objects in the schema, and then assign privileges

```
GRANT CONNECT ON DATABASE <database_name> TO <role_name>;
GRANT USAGE ON SCHEMA <schema_name> TO <role_name>;
GRANT SELECT ON ALL TABLES IN SCHEMA <schema_name> TO <role_name>;  -- all tables
GRANT SELECT ON TABLE <table_name> TO <role_name>;                  -- one table
```

It is also possible grant privileges on objects that will be created in the future. However, this applies to tables and views but NOT materialized views. For materialized views, privileges have to be granted explicitly after a new view is created.

```
ALTER DEFAULT PRIVILEGES IN SCHEMA <schema_name> GRANT SELECT ON TABLES TO <role_name>;
```

Altered default privileges can be checked with psql command `\ddp`. They can be revoked (reset to default) by:

```
ALTER DEFAULT PRIVILEGES IN SCHEMA <schema_name> REVOKE SELECT ON TABLES FROM <role_name>;
```

According to PostgreSQL documentation: “If you wish to drop a role for which the default privileges have been altered, it is necessary to reverse the changes in its default privileges or use DROP OWNED BY to get rid of the default privileges entry for the role.”

Revoke privileges

```
revoke select on table <schema>.<table> from <user>;
```

PL/R extension in PostgreSQL 9.6

Document Author: arho.virkki@tyks.fi

For details, see the PL/R manual: http://www.joeconway.com/plr/doc/index.html

Prerequisites

One needs *super user* priviliges to ***create new R functions*** in PostgreSQL. Once the function is created, also non-privileged users can use it.

For deploying new functions, we set up a new **ktp_adm** user (having the same password as the ordinary *aku* user).

```
CREATE ROLE ktp_adm LOGIN SUPERUSER PASSWORD '<passwd_here>';
```

Examples

R code can contain comments (starting with #) and empty lines. If empty lines produce an error, check that the function definition is passed to PostgreSQL in one batch (as *psql* `\i <filename.sql>` command does by default).

Vector argument, double precision output

```
-- The function is created into 'test' schema. It accepts
-- a double precision postgresql vector 'a' and returns
-- a double precision scalar

CREATE OR REPLACE FUNCTION test.array2double( a float8[] )
RETURNS float8 AS
$$
median ( a )
$$ LANGUAGE plr;

SELECT test.array2double( '{1,2,5}'::float8[] );
```

Single row output

```
CREATE OR REPLACE FUNCTION test.get_table()
RETURNS setof int4 AS $$
array(1:10)
$$ LANGUAGE plr;

SELECT test.get_table();
```

Table output from a data.frame

```
-- One option it to define a table and return a row set of that type

CREATE TABLE IF NOT EXISTS test.integertable (
  val1 int4,
  val2 int4 );

CREATE OR REPLACE FUNCTION test.get_table(n int4 default 10, m text default 'foo')
RETURNS setof test.integertable AS
$$
data.frame(1:n, n:1)
$$ LANGUAGE plr;

SELECT * FROM test.get_table(12);
SELECT * FROM test.get_table(m:='fifi');
```

```
-- Other option is to build the output record format in the
-- function definition

CREATE OR REPLACE FUNCTION test.get_table2(
  IN n int4,
  OUT val1 int4,
  OUT val2 int4)
RETURNS setof record AS $$
data.frame(1:n, n:1)
$$ LANGUAGE plr;

SELECT * FROM test.get_table(25);
```

Read table into a data.frame

```
CREATE OR REPLACE FUNCTION test.table_reader( table_name text )
RETURNS text AS
$$
#
# The queries follow RPostgreSQL package syntax with the
# exception that the connection object is neglected (and hence NA here)
#
X <- dbGetQuery( NA, paste("SELECT * FROM", table_name))
infotext <- paste0("nrow: ", nrow(X), " ncol: ", ncol(X), " colnames: ",
                   paste( colnames(X), collapse=",") )
#
return( infotext )
$$ LANGUAGE plr;

SELECT * FROM test.table_reader( 'test.koodisto_ods' );
```

Working with the Date type

```
DROP FUNCTION IF EXISTS delays.event_delays() ;
CREATE FUNCTION delays.event_delays(
  OUT id int4,                          -- Observe how
  OUT pid text,                         -- the output table is
  OUT event_idx int4,                   -- defined here
  OUT event text,
  OUT event_type text,
  OUT event_info text,
  OUT event_date date,                  -- Date type is 'date' in PosgreSQL
  OUT decision_date date,
  OUT first_treatment bool,
  OUT treatment_delay int4 )
RETURNS setof record AS
$$
con <- NA

# Detect first treatment events with plain SQL
events <- dbGetQuery(con, "
SELECT
    <QUERY HERE>
")
                                    ## In the following, we need to cast
                                    ## dates explicitly into 'Date' (as
                                    ## they where read as 'character')

events$decision_date <- as.Date(events$decision_date)
events$event_date <- as.Date(events$event_date)

   <R COMPUTATIONS>
                                    ## Finally, the internal R 'Date' types
                                    ## are cast back to 'character' for
                                    ## PostgreSQL to parse it as 'date'

events$decision_date <- as.character(events$decision_date)
events$event_date <- as.character(events$event_date)

return( events )                    ## Finally, return a data frame that
                                    ## Matches the FUNCTION definition

$$ language plr;
```

Installation

For compiling the extension, PostgreSQL command line tools need to be in path:

```
# this is a good place to put own customizations
cd /usr/local/bin/
sudo ln -s /usr/pgsql-9.6/bin/* .
```

Download the latest release of PL/L from https://github.com/postgres-plr/plr/releases

```
wget https://github.com/postgres-plr/plr/archive/REL8_3_0_17.tar.gz
tar xvzf REL8_3_0_17.tar.gz
cd plr-REL8_3_0_17
USE_PGXS=1 make

sudo bash -c "PATH=$PATH:/usr/local/bin/ USE_PGXS=1 make install"
```

Now we can install the extension to the desired databases:

```
sudo su - postgres
psql
\c ktp
CREATE EXTENSION plr;
-- Optionally, to get rid of the extension, issue:
-- DROP EXTENSION plr;
```

Now test the extension

```
SELECT * FROM plr_environ();
SELECT load_r_typenames();
SELECT * FROM r_typenames();
SELECT plr_array_accum('{23,35}', 42);
```

Test the installation

Example of a function (under test schema):

```
--
-- Create this function with 'kpt_adm' user
--
CREATE OR REPLACE FUNCTION test.r_max (a integer, b integer)
RETURNS integer AS '
    if (a > b)
       return(a)
    else
       return(b)
' LANGUAGE 'plr';
```

Test the function

```
-- Some test data
CREATE TABLE test.maxtest (
  id SERIAL,
  i INTEGER,
  j INTEGER);
INSERT INTO test.maxtest ( i, j ) VALUES
  ( 1, 3),
  ( 5, 2),
  (-9,-1);

-- Ordinary users can now use the function
SELECT id, test.r_max(i,j) AS maximum FROM test.maxtest;
```

PL/sh extension in PostgreSQL 9.6

Document Author: arho.virkki@tyks.fi

The installation, requirements, and usage is similar to R Language extension (PL/R).

For details, see the PL/sh Github page: https://github.com/petere/plsh

Installation

```
wget https://github.com/petere/plsh/archive/master.zip
unzip master.zip
cd plsh-master/
make
make install
sudo bash -c "PATH=$PATH:/usr/local/bin/ make install"
```

In the database, enable the extension with

```
CREATE EXTENSION plsh;
```

Examples

```
CREATE OR REPLACE FUNCTION delays.update_event_delays () RETURNS text
LANGUAGE plsh
AS $$
#!/bin/sh
./R/compute_delays2.R 2>/dev/null
echo $?
$$;
```

Postgres_fdw (Foreign Data Wrapper) and dblink

Document author: juha-matti.varjonen@tyks.fi

Following instruction descripts how to set up postgres_fdw (Foreign Data Wrapper) and dblink between two database in moondust instance:

Open superadmin (postgres) connection to ktp database

```
CREATE EXTENSION postgres_fdw;
CREATE EXTENSION dblink;
CREATE FOREIGN DATA WRAPPER to_destination_fdw VALIDATOR postgresql_fdw_validator;
-- where 'research' is target database
CREATE SERVER destination_server FOREIGN DATA WRAPPER to_destination_fdw OPTIONS (hostaddr
    '127.0.0.1', port '5432', dbname 'research');
--where '*' is password for 'ktp' account
CREATE USER MAPPING FOR ktp SERVER destination_server OPTIONS (user 'ktp', password '*');
GRANT USAGE ON FOREIGN SERVER destination_server TO ktp;
```

Open superadmin (postgres) connection to research database

```
CREATE EXTENSION postgres_fdw;
-- where 'ktp' is source database
CREATE SERVER source_server FOREIGN DATA WRAPPER postgres_fdw  OPTIONS (hostaddr '127.0.0.1', port
    '5432', dbname 'ktp');
-- where '*' is password for 'ktp' account
CREATE USER MAPPING FOR ktp SERVER source_server OPTIONS (user 'ktp', password '*');
GRANT USAGE ON FOREIGN SERVER source_server TO ktp;
```

Verify that connection works between databases:

Open normal (ktp) connection to ktp database:

```
SELECT dblink_connect('connection', 'destination_server');
SELECT dblink_send_query('connection', 'CREATE SCHEMA IF NOT EXISTS tutkimus_1');
SELECT dblink_disconnect('connection');
SELECT dblink_connect('connection', 'destination_server');
SELECT dblink_send_query('connection', 'IMPORT FOREIGN SCHEMA func LIMIT TO (mv_table_info) FROM
    SERVER source_server INTO tutkimus_1');
SELECT dblink_disconnect('connection');
SELECT dblink_connect('connection', 'destination_server');
SELECT dblink_send_query('connection', 'CREATE TABLE tutkimus_1.mv_table_info_transferred as
    (select * from tutkimus_1.mv_table_info)');
SELECT dblink_disconnect('connection');
```

Installing PostgreSQL 9.6 on CentOS 7

Document author: arho.virkki@tyks.fi

Dowload the latest PostgreSQL YUM repository package from

```
https://yum.postgresql.org/
```

Adapt the following line to point to the latest release:

```
sudo rpm -Uvh
    https://download.postgresql.org/pub/repos/yum/9.6/redhat/rhel-7-x86_64/pgdg-centos96-9.6-3.noarch.rpm
sudo yum update
```

Install postgresql

```
sudo yum install postgresql96 postgresql96-server postgresql96-contrib \
postgresql96-devel postgresql96-libs
```

Initialize the database

```
sudo /usr/pgsql-9.6/bin/postgresql96-setup initdb
```

Start the service and turn it on also after reboot (see sudo chkconfig –list for all services)

```
sudo systemctl enable postgresql-9.6
sudo systemctl start postgresql-9.6
```

Try how the database works (postgres user can connect to the db using psql)

```
sudo su - postgres
psql
```

To install PostgreSQL Adminpack, enter the command in postgresql prompt:

```
CREATE EXTENSION adminpack;
```

Configure PostgreSQL-MD5 Authentication

By default, Posgresql uses ident authentication, so that the local system users can be granted access to databases own by them. If you want to set MD5 authentication to require users to enter passwords.

Open host-based authentication file:

```
sudo vim /var/lib/pgsql/9.6/data/pg_hba.conf
```

Add the following lines to the end of the file

```
# Connections from vsshp
host    all             all             10.150.0.0/16           md5
host    all             all             10.145.0.0/16           md5
```

Then, edit the file PostgreSQL conf file:

```
sudo vim /var/lib/pgsql/9.6/data/postgresql.conf
```

Change the line:

```
#listen_addresses = 'localhost'
#port = 5432
```

to

```
listen_addresses = '*'
port = 5432
```

and also change the line:

```
#password_encryption = on
```

to

```
password_encryption = on
```

Now, restart postgresql service to apply the changes:

```
sudo systemctl restart postgresql-9.6
```

Moving PostrgeSQL installation directory

Since CentOS systemctl script contain hard-coded paths to */var/lib/pgsql*, it is easiest to move the whole directy and create a symbolic link to point to the new location.

```
sudo systemctl stop postgresql-9.6

sudo mv /var/lib/pgsql /srv/
ln -s /srv/pgsql /var/lib/pgsql

sudo systemctl start postgresql-9.6
```

Adjust PostgreSQL configuration parameters

Check the PostgreSQL settings with

```
SHOW all;
SHOW effective_cache_size ;
```

Read about the recommended values (set in *postgresql.conf*) from https://www.postgresql.org/docs/9.6/static/runtime-config-resource.html and https://wiki.postgresql.org/wiki/Tuning_Your_PostgreSQL_Server

sudo vim /var/lib/pgsql/9.6/data/postgresql.conf

```
shared_buffers = 8192MB
temp_buffers = 32MB
maintenance_work_mem = 1024MB
```

Processes on the PostgreSQL server

Document author: anna.hammais@tyks.fi

Existing connections

List all database connections

```
select * from pg_stat_activity;
```

Table locks

A pending transaction may hold a lock on a table, preventing other transactions from accessing it. Checking the locks on a specific table:

```
select l.* from pg_locks l join pg_class t on l.relation = t.oid
where t.relkind = 'r' and t.relname = <your_table_name>;
```

This shows which processes are holding locks.

If you wish to know more about the process that is holding the lock, take the PID and see what the process is:

```
ps -f <your_PID>
```

You can kill it by:

```
sudo kill <your_PID>
```

Sometimes killing the process that is the original cause of the problem can remove all subsequent locks on the table as well.

RAM and CPU usage of Postgres processes

```
ssh aku@dbdev
ps -u postgres uf
```

Last column of output shows whether the process is idle or not.

Disk space report

```
df -h
```

Disk IO stats

```
sudo iotop -aoP
```

-a Will show accumulated output -o Will only output processes/threads doing IO -P Will only show processes instead of threads

PostgreSQL server configuration

Document author: anna.hammais@tyks.fi & juha-matti.varjonen@tyks.fi

Useful options to cofigure in the PostgreSQL server, according to https://wiki.postgresql.org/wiki/Tuning_Your_PostgreSQL_Server:

```
select * from pg_settings
where name in
('shared_buffers',
'checkpoint_segments',
'effective_cache_size',
'checkpoint_completion_target',
'maintenance_work_mem',
'autovacuum_work_mem',
'wal_buffers',
'config_file');
```

- shared_buffers – 25% of system memory (default very small)
- checkpoint_segments – default 3, set to 20
- effective_cache_size – 50% of system RAM (not hard allocation, just an estimate of max needs)
- checkpoint_completion_target – set to 0.9
- maintenance_work_mem – set to 64MB
- autovacuum_work_mem – -1, which means to use maintenance_work_mem
- wal_buffers – default 1/32 of shared buffers, which is ok

These changes should be entered in the *postgresql.conf* file, located by starting psql as superuser on the dbdev server:

```
ssh aku@dbdev.intra.net
sudo su -
psql

psql> show config_file;
```

The PostgreSQL server must be restarted for these changes to take effect. You can check this by re-running the above select query after the restart.

Following changes made to PostgreSQL configuration in moondust.intra.net 2018-04-22:

More detailed information of all configuration parameters can be found from PostgreSQL own wiki pages: https://wiki.postgresql.org/wiki/Tuning_Your_PostgreSQL_Server

```
ALTER SYSTEM SET
 -- Sets the maximum number of concurrent connections.
 max_connections = '100';
ALTER SYSTEM SET
-- Sets the number of shared memory buffers used by the server.
 shared_buffers = '20GB';
ALTER SYSTEM SET
-- Sets the planner's assumption about the size of the disk cache.
 effective_cache_size = '60GB';
ALTER SYSTEM SET
-- Sets the maximum memory to be used for maintenance operations.
 maintenance_work_mem = '2GB';
ALTER SYSTEM SET
-- Sets the minimum size to shrink the WAL to.
 min_wal_size = '4GB';
ALTER SYSTEM SET
-- Sets the WAL size that triggers a checkpoint.
 max_wal_size = '8GB';
ALTER SYSTEM SET
-- Time spent flushing dirty buffers during checkpoint, as fraction of checkpoint interval.
 checkpoint_completion_target = '0.9';
ALTER SYSTEM SET
-- Sets the number of disk-page buffers in shared memory for WAL.
 wal_buffers = '16MB
ALTER SYSTEM
-- Sets the default statistics target.
 default_statistics_target = '500';
```

```
ALTER SYSTEM SET
-- Sets the planner's estimate of the cost of a nonsequentially fetched disk page.
 random_page_cost = '4';
ALTER SYSTEM SET
-- Number of simultaneous requests that can be handled efficiently by the disk subsystem.
 effective_io_concurrency = '2';
ALTER SYSTEM SET
-- Maximum number of concurrent worker processes.
 max_worker_processes = '32';
ALTER SYSTEM SET
-- Sets the maximum number of parallel processes per executor node.
 max_parallel_workers_per_gather = '16';
ALTER SYSTEM SET
-- Sets the maximum memory to be used for query workspaces.
 work_mem = '6553kB';
```

psql commands

Document author: anna.hammais@tyks.fi

List psql commands

```
psql=> \?
```

See also http://postgresguide.com/utilities/psql.html for a brief introduction.

Show databases

```
psql=> \l+                          -- shows databases and their space usage
psql=> \c[onnect] <database_name> -- connects to specific database
psql=> \conninfo                  -- shows which database you are currently connected to, and as
    which user
```

PostgreSQL syntax

Document author: anna.hammais@tyks.fi

Regexp replace (‘g’ means replace all occurrences)

```
regexp_replace(<column_name>, '\n', '|n|', 'g')
```

Add primary key

```
ALTER TABLE stage_uraods.diagnoosi ADD CONSTRAINT diagnoosi_pkey PRIMARY KEY ( dgn_numero );
```

Remove Social ID number (HETU) from text

```
select
teksti,
regexp_replace(teksti, '([0-9]{6}[A-][0-9]{3}[a-zA-Z0-9_.-])', '<HETU>','g') as teksti_new
from text_mine.v_teksti
LIMIT 1000
```

Setup per-user logging and import a log file into table

Document author: juha-matti.varjonen@tyks.fi

This document descripes how to set up user level logging and import a log file in specific table created.

Edit postgresql.conf file

```
>sudo su - postgres
-bash-4.2$ nano <server_specific_folder>/postgresql.conf

 # In moondust_dev.intra.net, replace <server_specific_folder> with /srv/pgsql/10/data/
 # In moondust.intra.net, replace <server_specific_folder> with /var/lib/pgsql/9.6/data

 # In postgresql.conf, edit following lines:

log_destination = 'csvlog'
log_filename = 'postgresql-%Y-%m-%d.log'

 # Save changes and restart PostgreSQL server
 # In Gradient-Dev
-bash-4.2$ service postgresql-10 restart
 # In Gradient
-bash-4.2$ pg_ctl restart
```

The log files are store under /log or pg_log depending version number of PostgreSQL

Set up per-user logging

You have to log with postgres account in PostgreSQL server

Run command:

```
ALTER USER <username> SET log_statement TO 'all';

# remove logging from user
ALTER USER <username> SET log_statement TO 'none';
```

See more about log_statement parameter from https://www.postgresql.org/docs/current/static/runtime-config-logging.html

Import a log file into table

First you have to create a table in database, in this case we create a table ‘log.postgres_log’ in ‘ktp’ database and grant all for user ‘ktp’ in this table:

```
CREATE SCHEMA log
  AUTHORIZATION ktp;

GRANT ALL ON SCHEMA log TO ktp;

CREATE TABLE log.postgres_log
(
  log_time timestamp(3) with time zone,
  user_name text,
  database_name text,
  process_id integer,
  connection_from text,
  session_id text,
  session_line_num bigint,
  command_tag text,
  session_start_time timestamp with time zone,
  virtual_transaction_id text,
  transaction_id bigint,
  error_severity text,
  sql_state_code text,
```

```
  message text,
  detail text,
  hint text,
  internal_query text,
  internal_query_pos integer,
  context text,
  query text,
  query_pos integer,
  location text,
  application_name text,
  PRIMARY KEY (session_id, session_line_num)
);

ALTER TABLE log.postgres_log
  OWNER TO ktp;
GRANT ALL ON TABLE log.postgres_log TO ktp;
```

For importing a logfile in this table, use postgresql account and psql client in server where PostgreSQL server is running:

```
>sudo su - postgres
-bash-4.2$ psql -U postgres -h <server_name> ktp

 # In moondust_dev.intra.net, replace <server_name> with moondust_dev.intra.net
 # In moondust.intra.net, replace <server_name> with moondust.intra.net
```

Use following statement for importing csv file in table

```
COPY log.postgres_log FROM '<server_specific_folder>/<pg_>_log/postgresql-YYYY-MM-DD.csv' WITH csv;

  # In moondust_dev.intra.net, replace <server_specific_folder> with /srv/pgsql/10/data/
  # In moondust.intra.net, replace <server_specific_folder> with /var/lib/pgsql/9.6/data

# to see the imported content use command:

SELECT * FROM log.postgres_log;
```

Vacuum PostgreSQL database manually

Document author: juha-matti.varjonen@tyks.fi

There are two reason why the database tables should be Vacuumed and Analyzed.

VACUUM = Is to have free space available in tables (fsm = free space map).

ANALYZE = You need to analyze the database so that the query planner has table statistics it can use when deciding how to execute a query.

- You can setting up Auto Vacuum (by default it is on and shouldn't ever turned off)
- Or if you see that Auto Vacuum doesn't work perfectly e.g. tables n_dead_tup sizes increase, you can use Manual Vacuum Analyze script explained below.

The shell script is stored in Git repository:

```
gitbox.intra.net:/opt/git/Common.git
```

Under this repository there is subfoder ~/Common/script/shell where you can locate file vacuum_analyze_and_reindex_procedure.sh . The script must be executed by aku@moondust: (please note that reindex part is commented out and not tested as it might take so long time to execute):

```
~/Common/script/shell/vacuum_analyze_and_reindex_procedure.sh
```

This script check all tables from func.mv_table_info and compare this information to pg_stat_all_tables for getting the n_live_tupe and n_dead_tup information for those tables. The VACUUM ANALYZE script will be execute for certain table if relation of n_dead_tup (dead rows) and n_live_tup (row count) is more or equal to 4%. Otherwise table is skipped from VACUUM ANALYZE

How to handle view dependencies in PostgreSQL

Document author: juha-matti.varjonen@tyks.fi

This documents refers to instruction found from the Internet:

http://pretius.com/postgresql-stop-worrying-about-table-and-view-dependencies/

By using the instuction above, following tables and fuctions are created in the ktp database:

```
func.deps_saved_ddl
func.deps_save_and_drop_dependencies
func.deps_restore_dependencies
```

Fuction creation scripts are stored under refinery-scripts (aku@gitbox:/opt/git/refinery-scripts.git)

The usage of these fuctions is easy

```
select func.deps_save_and_drop_dependencies('p_schema_name', 'p_object_name');
```

You have to pass two arguments: the name of the schema and the name of the object in that schema. This object can be a table, a view or a materialized view. The function will drop all views and materialized views dependent on p_schema_name.p_object_name and save DDL which restores them in a helper table.

When you want to restore those dropped objects (for example when you are done modifying p_schema_name.p_object_name), you just need to make another simple call:

```
select func.deps_restore_dependencies('p_schema_name', 'p_object_name');
```

These functions take care of:

- dependencies hierarchy
- proper order of dropping and creating views/materialized views across hierarchy
- restoring comments and grants on views/materialized views

Flask installation

Document author : per-erik.gustafsson@tyks.fi

Create virtual environment for running Flask

```
First some tools:
sudo apt-get install python3-pip
sudo pip3 install --upgrade pip
sudo pip3 install virtualenv
```

Make directory for your environment Your application does not have to be in this environment, I think!

```
cd ~
mkdir myVenv
(sudo) virtualenv myVenv

Do not use: python3 -m venv ~/myVenv if you will run your app as a Apache/Nginx server app
  This does not create the needed file bin/activate_this
```

Activate virtual environment and install flask components

```
cd ~/myVenv/bin
source activate
pip3 install --upgrade pip
pip3 install flask
pip3 install flask_sqlalchemy
pip3 install psycopg2-binary
pip3 install flask_login
pip3 install flask_bootstrap
pip3 install flask_wtf
pip3 install flask_restful
pip3 install flask_babel
pip3 install flask_table
```

Clone the source code for your project from the git repo

```
cd ~/home/myVenv
mkdir urologia_laatu
git clone aku@gitbox.intra.net:/opt/git/urologia_laatu.git
```

Run Flask

```
cd ~/myVenv/urologia_laatu/app
cd ..
ls -l
export FLASK_APP=run.py
export FLASK_CONFIG=development
export FLASK_DEBUG=1
flask run
```

Deactivate virtual environment

```
deactivate
```

Modifications to application for deployment on http server

Instance folders

In your python code:

```
def create_app():
  # initialize Flask app
  app = Flask(__name__, instance_relative_config=True)
  app.config.from_pyfile('config.py')
  The config.py file is written in INI-style
  MY_PW='salasana'
```

In your code you reference values:
my_password = current_app.config[‘MY_PW’]

```
Some reading about Flask instance folders
http://flask.pocoo.org/docs/dev/config/
http://modwsgi.readthedocs.io/en/develop/user-guides/virtual-environments.html
http://modwsgi.readthedocs.io/en/develop/configuration-directives/WSGIDaemonProcess.html
```

Just do it

Put your secrets in config file
On server: $PREFIX/var/myapp-instance
This is under your virtual directory

On your local computer:
Directory /instance in your application root
Add this folder to .gitignore, and you can store your password in /instance/config.py! ### Compile mod_wsgi The Apache module mod_wsgi needs to be compiled for our Python version (3.x) and you need the apache2-dev package

```
First:
sudo apt-get install python3-dev ,in case it is not installed
sudo apt-get install libapache2-mod-wsgi before you compile. A convenient way to get the .conf and
    .load files
Download from https://github.com/GrahamDumpleton/mod_wsgi/releases
tar xvzf mod_wsgi-X.Y.tar.gz
cd mod_wsgi-X.Y
./configure --with-apxs=/usr/bin/apxs \--with-python=/usr/bin/python3.5
make
sudo make install



mkdir -p /var/www/pullot/venvFlask1
  Set owner, group
sudo virtualenv /var/www/pullot/venvFlask1
cd /var/www/pullot/venvFlask1
Aktivate the virtual environment:
source bin/activate
pip3 install flask
pip3 install ... needed modules
```

Apache config

```
Now testing without ssl, when moving data around, see that ssl (https) is used!
/etc/apache2/site-enabled/xxx.conf
<VirtualHost *:8000>
  ServerAdmin per-erik.gustafsson@tyks.fi
  DocumentRoot "/var/www/html"

  WSGIDaemonProcess uro
      python-path=/var/www/pullot/venvFlask1/bin:/var/www/pullot/venvFlask1/lib/python3.5/site-packages
      home=/var/www/pullot/venvFlask1/urologia_laatu python-home=/home/perre/wwwVenv user=www-data
      group=www-data
  WSGIScriptAlias / "/var/www/pullot/venvFlask1/urologia_laatu/wsgistart.wsgi"
    <Directory "/var/www/pullot/venvFlask1">
      WSGIProcessGroup uro
      WSGIApplicationGroup %{GLOBAL}
    </Directory>

    <Location "/">
      AuthType Basic
      AuthName "Flask test"
      AuthUserFile /etc/apache2/.htpasswd
      Require valid-user
    </Location>

</VirtualHost>
```

Neo4j installation and administration

Document author : per-erik.gustafsson@tyks.fi

Download from https://neo4j.com/download/

Online course https://neo4j.com/online_training/neo4j-in-production/

Lets start with the community version, standard with extensions installation mode

Installing

```
$ wget -O - https://debian.neo4j.org/neotechnology.gpg.key | sudo apt-key add -
$ echo 'deb https://debian.neo4j.org/repo stable/' | sudo tee /etc/apt/sources.list.d/neo4j.list
$ sudo apt-get update
$ sudo apt-get install neo4j
```

Settings in /etc/neo4j/neo4j.conf

```
\# Network connector configuration

\# With default configuration Neo4j only accepts local connections.
\# To accept non-local connections, uncomment this line:
#### dbms.connectors.default_listen_address=0.0.0.0

\# The address at which this server can be reached by its clients. This may be the server's IP
    address or DNS name, or
\# it may be the address of a reverse proxy which sits in front of the server. This setting may be
    overridden for
\# individual connectors below.
##### dbms.connectors.default_advertised_address=sonja.intra.net
```

Restart server
$ sudo neo4j restart
Getting warning:
Starting Neo4j. WARNING: Max 1024 open files allowed, minimum of 40000 recommended. See the Neo4j manual.
Anybody who knows where this setting is?

Some notes about setting up SAML with pySaml

Document Author: per-erik.gustafsson@tyks.fi

SAML is used world-wide as a method to log on using a thrid part identity provider

Check out:
https://pysaml2.readthedocs.io/en/latest/index.html
https://github.com/pitbulk/python3-saml
https://www.elastic.co/blog/how-to-enable-saml-authentication-in-kibana-and-elasticsearch

SAML, Security Assertion Markup Language, is a standard for exchanging authentication and authorization data between parties, in particular, between an identity provider and a service provider.
The base parts are Identity Provider and Service Provider.

Identity Provider

Install and set up

Downloading from https://github.com/IdentityPython/pysaml2
Following instructions on https://pysaml2.readthedocs.io/en/latest/index.html

Changed code in sp.py from http.cookies import SimpleCookie
from urllib.parse import parse_qs

in who.ini, added saml2 to s2repoze
use = saml2.s2repoze.plugins.sp:make_plugin [general] request_classifier = saml2.s2repoze.plugins.challenge_de

Service Provider

Install and set up

```
Dependences

    python 2.7 // python 3.3
    xmlsec Python bindings for the XML Security Library.
    isodate An ISO 8601 date/time/duration parser and formater
    (python3-pkconfig)


Make a virtual environmet, and install module python3-saml

We will nedd a X.509 certificate pair, creating it
openssl req -new -x509 -days 3652 -nodes -out sp.crt -keyout saml.key


Changed source code in index.py:
from urllib.parse import urlparse
```

Security warning

In production, the strict parameter MUST be set as “true”. Otherwise your environment is not secure and will be exposed to attacks.

Visualization Tools

Document Authors: arho.virkki@tyks.fi, anna.hammais@tyks.fi

Tools under development

- Timeline
- KTP portal
- Views under development (FIN)
- Views under development (ENG)

Reviews

- http://www.creativebloq.com/design-tools/data-visualization-712402/1
- http://inspire.blufra.me/big-data-visualization-review-of-the-20-best-tools/

Comparisons

- http://www.fusioncharts.com/javascript-charting-comparison/

Tools

- Serene Development

D3.js based (ordered by project activity)

1. https://d3js.org/ (Revised BSD, see D3 and BSD licenses)
2. http://emberjs.com/ (MIT license; see Ember and MIT and Revised BSD for D3.js)
3. http://raw.densitydesign.org/ (LGPLv3 and Revised BSD for D3.js)
4. http://nvd3.org/ (Apache 2.0 and Revised BSD for D3.js)
5. http://processingjs.org/ (MIT license like this and Revised BSD for D3.js)

Other JavaScript libraries

1. dygraphs
2. Flot

Commercial (but free-of-charge for non-profit and personal use)

- http://www.fusioncharts.com/
- http://www.highcharts.com/

PaaS

- https://plot.ly/feed/

Packages

- https://www.filamentgroup.com/lab/update-to-jquery-visualize-accessible-charts-with-html5-from-des
  html
- https://github.com/n3-charts
- http://sigmajs.org/

Creating custom visualizations

- Creating custom visualizations

Connecting Data Analysis Platform (DAP)

Document Author: arho.virkki@tyks.fi

DAP Concept

Turku Centre for Clinical Informatics at the Hospital District of Southwest Finland offers Data Analytics Platform (DAP) for data-driven research projects. This guide explains how to get credentials to DAP, and the optional step of installing an open-sourced X2Go software package for Linux, Apple or Windows for native remote desktop access.

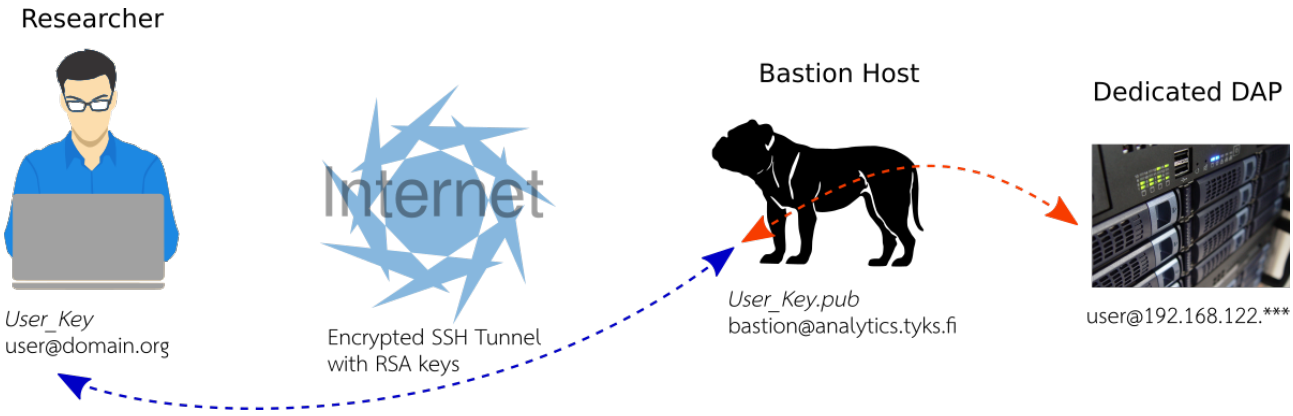


Figure 1:

Architecture. A Schematic illustarion of the Data Analytics Platform (DAP). API end points and web services like "R Studio Server" can be tunneled to researcher's computer through the Secure Shell (SSH) connection, which is a sptandard Internet protocol for encrypted communication.

1. Obtain Key Pair for Asymmetric Cryptography

Visit the CCI office, or generate the files yourself with the instructions below.

On Linux and Apple, the needed tools come bundled. On Windows 10, you need either to enable the Windows Sybsystem for Linux [1], or get a separate tool that installs OpenSSH. The easiest way for Windows users is to install Git [2], which comes with a bundled bash shell and ssh.

When ready, open Terminal.app, Gnome Terminal or Bash Launcher, and generate a key pair with the command

```
ssh-keygen -C "Your Name" -f Your_Name
```

The command generates two files, for example

```
John_Doe
John_Doe.pub
```

Email the public key (with the .pub extension) to aku@tyks.fi (cc: arho.virkki@tyks.fi). Keep the private key (with no extension) in a safe place. **Do not email the private key to anyone!**

Example. Generating keys public and private keys.

[1] http://www.howtogeek.com/249966/how-to-install-and-use-the-linux-bash-shell-on-windows-10/
[2] https://git-scm.com/

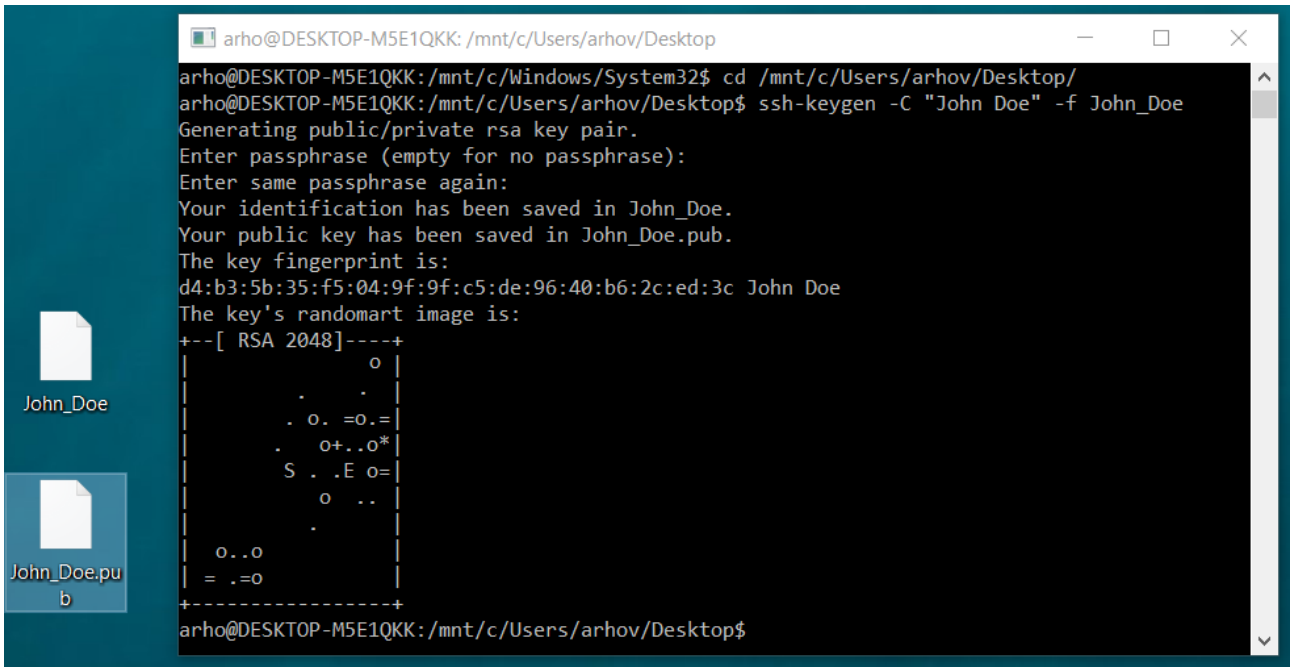


Figure 2:

2. Connect DAP via Command Line

Connection works only after CCI has obtained your public key and added it to the system.

Again, open Terminal.app, Gnome Terminal or Bash Launcher, and issue the command

```
ssh <user>@<machine>.scilake.net -o \
ProxyCommand="ssh bastion@supernova.public.net -W %h:%p -i <private_key>"
```

where “" denotes the continuation of line. For example,

```
ssh john@textmine.scilake.net \
  -o ProxyCommand="ssh bastion@supernova.public.net \
  -W %h:%p -i ~/c/Users/jdoe/John_Doe"
```

You should be able to log in with the correct password.

Using services on the DAP is now easy. For example, you can tunnel R Studio Server through the connection by adding its port as an option in the above command,

```
-L 8787:localhost:8787
```

Now, R Studio Server can be reached by pointing a browser to address http://localhost:8787, as shown in the image below.

Example. Using R Studio Server on DAP.

Opening the connection can be automated e.g. by modifying the following script template

```
#!/bin/bash

ssh john@medbox.scilake.net -L 5432:localhost:5432 \
  -o ProxyCommand="ssh -i /c/Users/john/.ssh/John_Doe bastion@supernova.public.net -W %h:%p" \
  "echo 'Tunnel open. Close with <Crtl> + C';
  while true
  do echo -n '.'; sleep 10;
  done"
```

Example. Clicking the script *open_tunnel.sh* on Windows will automatically launch the script (when Git is installed).

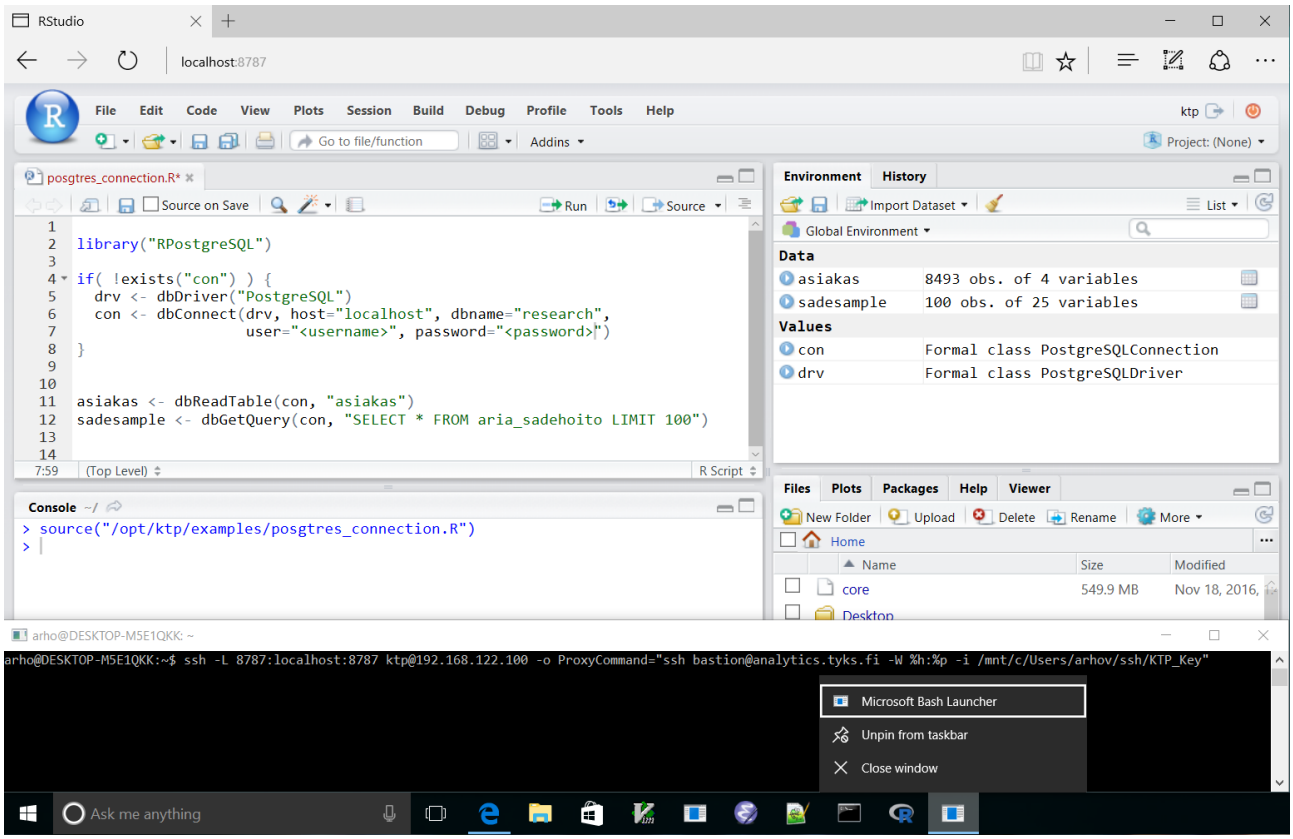


Figure 3:

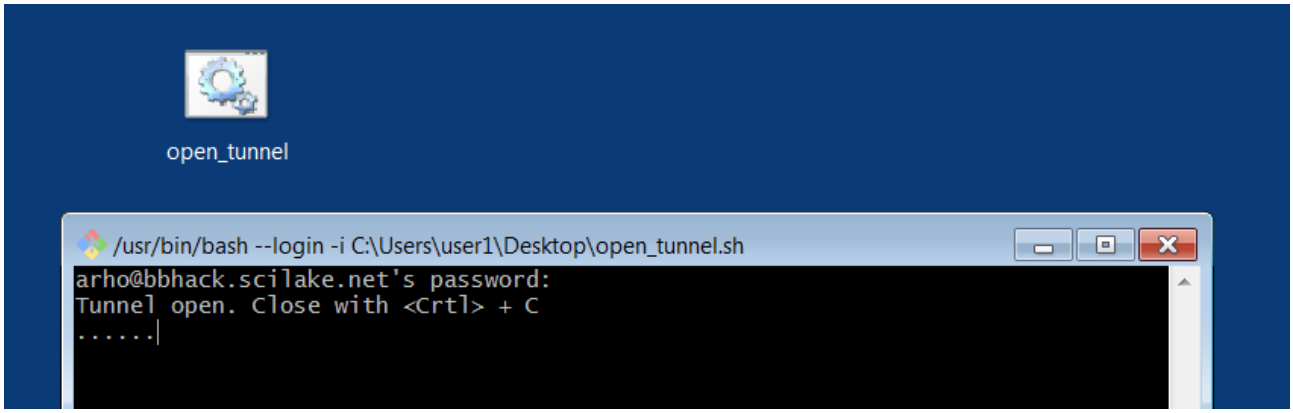


Figure 4:

Configuring SSH for Transparent Proxy

OpenSSH (on Linux, OS X, and with Git Bash on Windows) supports automated rules in *.ssh/config*. For example, the following lines

```
Host *.scilake.net
  ProxyCommand ssh bastion@supernova.public.net -W %h:%p

Host supernova.public.net
  IdentityFile /home/<username>/.ssh/<private key>
```

enable ssh to understand the command

```
ssh mymachine.scilake.net
```

directly, without explicitly writing every time the ProxyCommand clause. In addition to the ssh client itself, this also applies to the *scp* and *sftp* commands.

3. Connect DAP Using X2Go Remote Desktop

Download and install a copy of X2Go from http://wiki.x2go.org. This will provide you a full-blown graphical desktop to the remote server.

The settings must be chosen exactly as follows:

- Host: the IP of your machine, e.g. medrunner.scilake.net
- Login: You account name, e.g. johndoe
- [x] Use Proxy server for SSH connection
- Type: SSH
- Login: bastion
- Host: supernova.public.net
- RSA/DSA key: Full path to your *private* key
- Session type: XFCE

Example. Connection parameters for X2Go.

Known limitations

- On Windows, X2Go does not suppors switching between full screen and windowed mode during a single session, or does not support full screen mode at all.
- If the path to the RSA/DSA private key is wrong, the software will close with no error message.

Accessing the Data

In most cases, the data is stored in PosgreSQL database engine in a database named *research*. A ready-made ordinary user *analyst* (with the same default password) can access this database from the virtual machine's terminal with the command

```
PGPASSWORD=analyst psql -U analyst -d research -h localhost
```

The data resides in the *data* schema. The tables in this schema can be listed as follows:

```
research=> \dt data.
                List of relations
 Schema |             Name              | Type  | Owner
--------+-------------------------------+-------+-------
 data   | asiakas                       | table | ktp
 data   | diagnoosi                     | table | ktp
 data   | laake_maarays                 | table | ktp
 data   | labrat                        | table | ktp
 data   | leikkaus_opera                | table | ktp
```

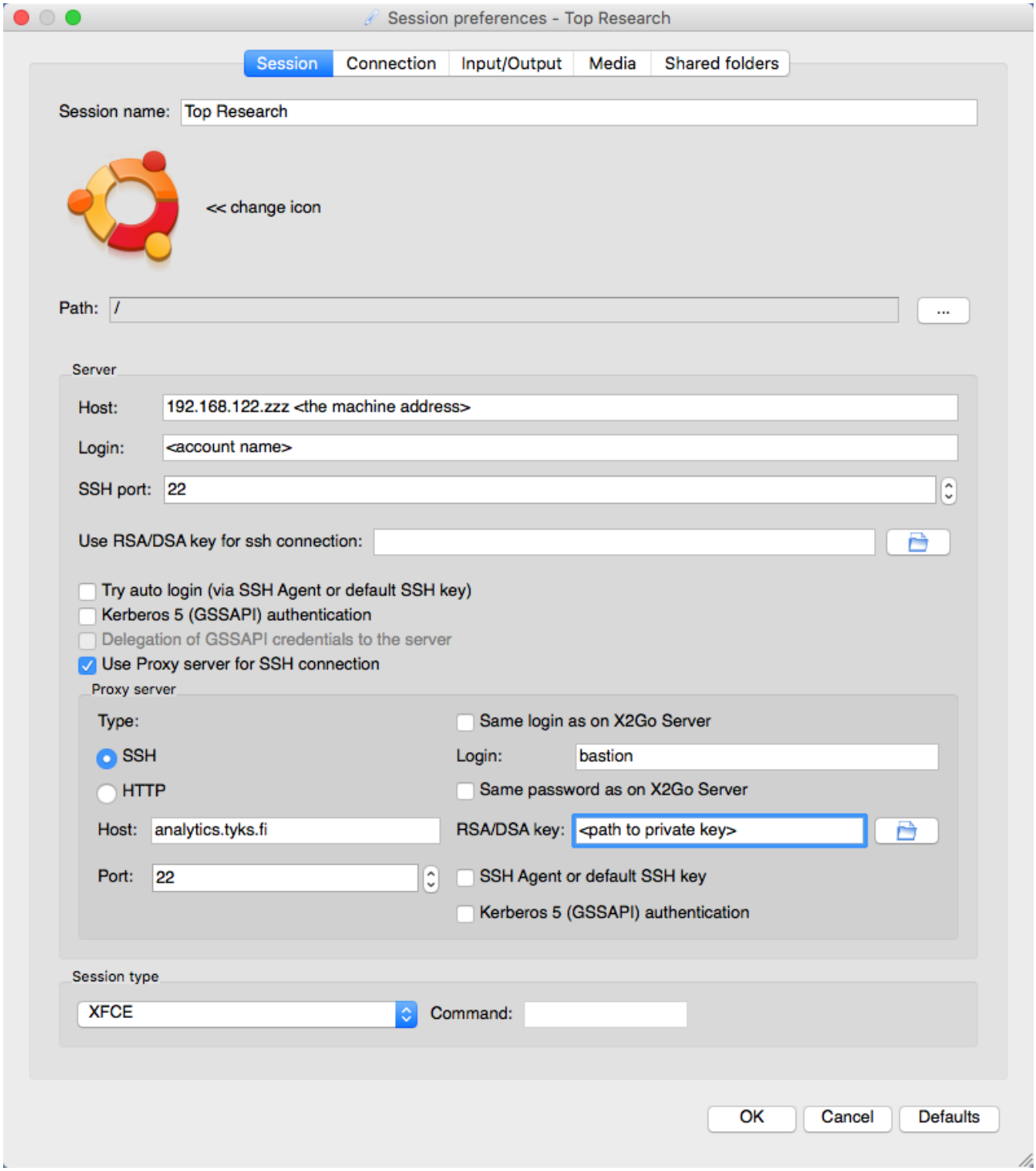

Session preferences - Top Research
Session
Connection
Input/Output
Media
Shared folders
Session name: Top Research
<< change icon
Path: /
...
Server
Host: 192.168.122.zzz <the machine address>
Login: <account name>
SSH port: 22
Use RSA/DSA key for ssh connection:
Try auto login (via SSH Agent or default SSH key)
Kerberos 5 (GSSAPI) authentication
Delegation of GSSAPI credentials to the server
Use Proxy server for SSH connection
Proxy server
Type:
SSH
HTTP
Host: analytics.tyks.fi
Port: 22
Same login as on X2Go Server
Login: bastion
Same password as on X2Go Server
RSA/DSA key: <path to private key>
SSH Agent or default SSH key
Kerberos 5 (GSSAPI) authentication
Session type
XFCE
Command:
OK
Cancel
Defaults

Figure 5:

```
 data   | leikkaus_toti                 | table | ktp
 data   | oberon_toimenpiteet           | table | ktp
 data   | palvelu                       | table | ktp
 data   | radut                         | table | ktp
 data   | reseptit                      | table | ktp
 data   | yhdistetty                    | table | ktp
(11 rows)
```

Uploading Own Data

Option 1.

Use *X2Go Shared folders* feature to copy data back and forth between the computing environment.

Option 2.

Use command line *scp* (With Linux, OS X, and Microsoft with bash installed)

Examples:

```
# Open a tunnel to the remote machine at local port 2222.
ssh -N -L 2222:<machine>.scilake.net:22 bastion@supernova.public.net -i <private_key_file> &

# Copy the file(s) to DAP Desktop folder
scp -P 2222 <my_files> <username>@localhost:~/Desktop/
```

Option 3.

Use dedicated secure copy software like WinSCP on Windows which can tunnel connections through the bastion host.

Installing WinCSP (for Windows)

Download a recent copy of WinSCP from https://winscp.net/eng/download.php and run the installation package. (In case you do not have administrator rights, choose portable executables. They are just launched directly and not installed.)

Configure site. Set host name to your DAP IP or host name (.scilake.net) and user name and password to your personal credentials. Then click the "Advanced…" button.

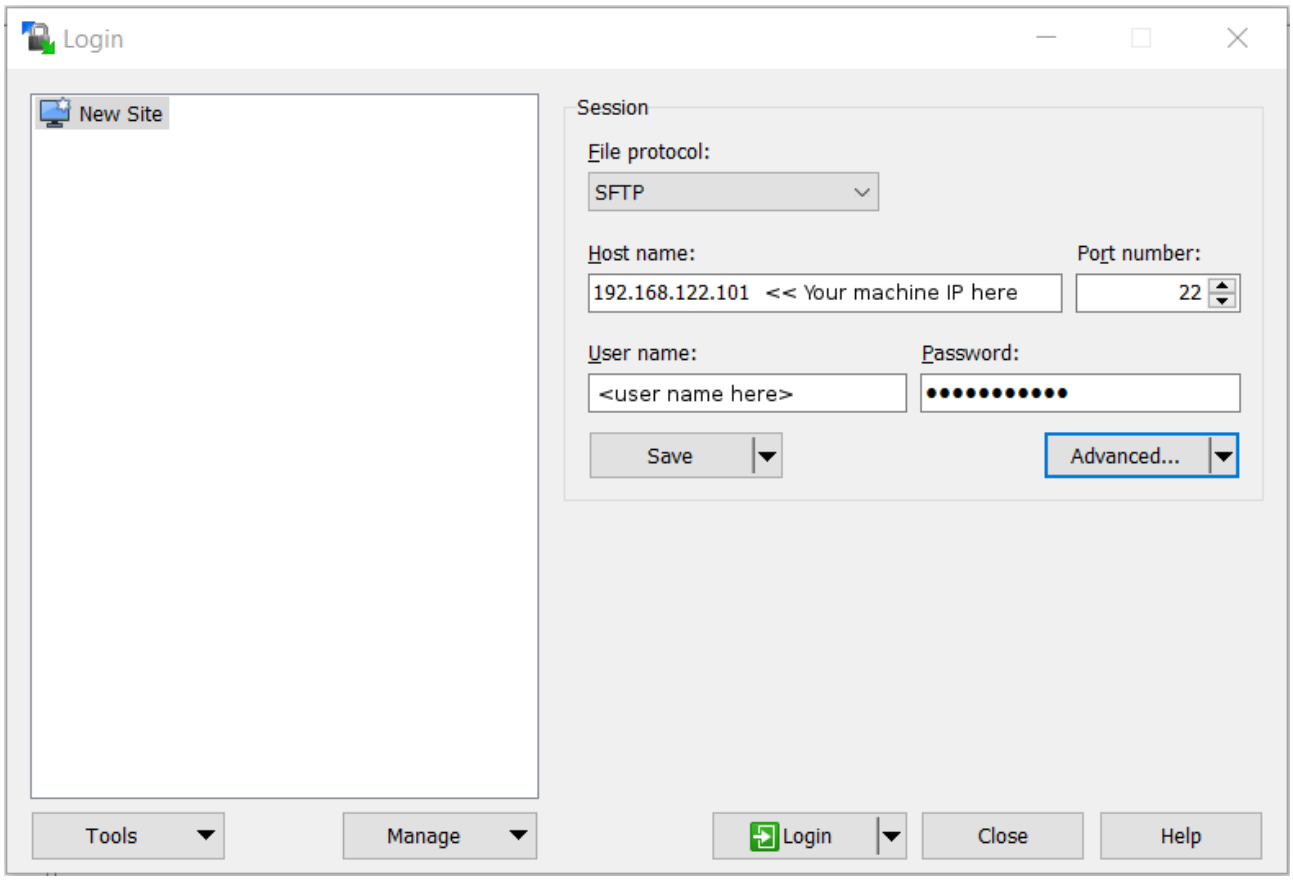
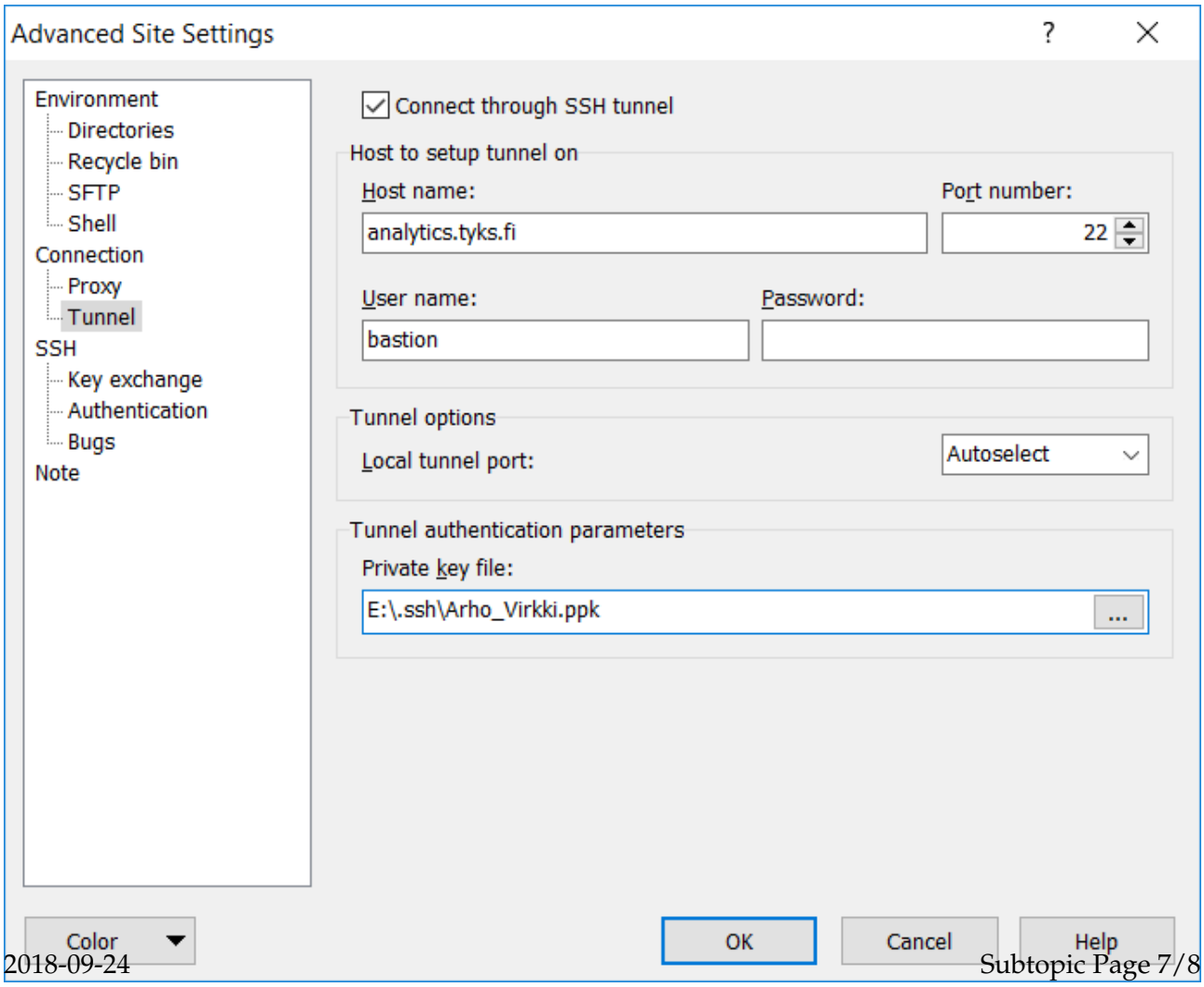

Login
New Site
Session
File protocol:
SFTP
Host name:
Port number:
192.168.122.101 << Your machine IP here
22
User name:
Password:
<user name here>
Save
Advanced...
Tools
Manage
Login
Close
Help

Figure 6:


Advanced Site Settings
Environment
Directories
Recycle bin
SFTP
Shell
Connection
Proxy
Tunnel
SSH
Key exchange
Authentication
Bugs
Note
Connect through SSH tunnel
Host to setup tunnel on
Host name:
Port number:
analytics.tyks.fi
22
User name:
Password:
bastion
Tunnel options
Local tunnel port:
Autoselect
Tunnel authentication parameters
Private key file:
E:\.ssh\Arho_Virkki.ppk
...
Color
OK
Cancel
Help

Set up tunnel. Select Connection -> Tunnel,and [x] Connect through SSH tunnel. Host name must be **supernova.public.net** and user **bastion**. Finally choose your private key file. When given an OpenSSH privater key, WinSCP offers to convert it into (Windows-specific) Putty format.

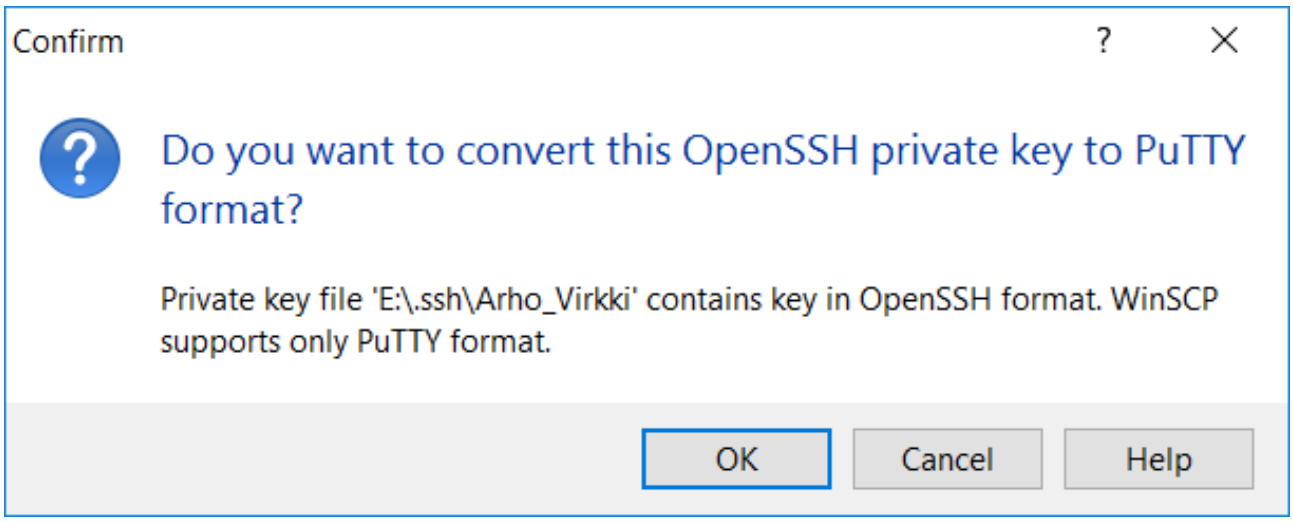


Figure 7:

Key conversion. OpenSSH private key can be converted into Putty format and saved.

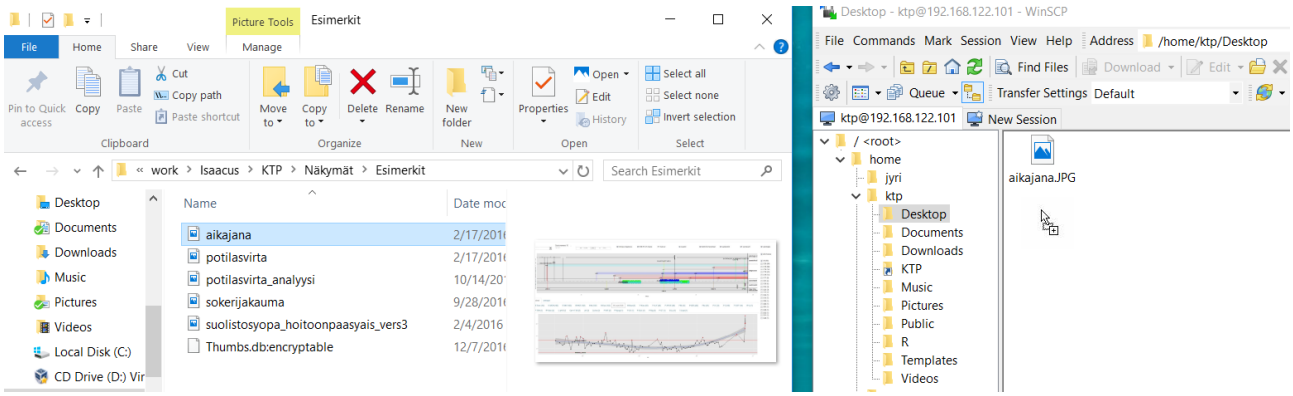


Figure 8:

File copying. Files can be copied with drag and drop.

Limited SFTP Access to supernova.public.net

Document Author: arho.virkki@tyks.fi

Adding a new SFTP user

The *NEWUSER* and *ID_PUB* environment variables should match the new user account name and the corresponding OpenSSH public key file.

```
# Create user and add it to the 'sftp-only' group
NEWUSER=test1

sudo mkdir /var/sftp/$NEWUSER
sudo useradd $NEWUSER -d /$NEWUSER -M
sudo usermod $NEWUSER -aG sftp-only

# Copy a public key to .ssh/authorized_keys and set its permissions
ID_PUB=Key_Holder_Name.pub

sudo mkdir -p /var/sftp/$NEWUSER/.ssh
sudo cp $ID_PUB /var/sftp/$NEWUSER/.ssh/authorized_keys
sudo chmod og-rwx /var/sftp/$NEWUSER/.ssh
sudo chown $NEWUSER: -R /var/sftp/$NEWUSER
```

Prerequisites for the Setup

Adapted from: https://access.redhat.com/solutions/2399571

Add group for SFTP Users

```
sudo groupadd sftp-only
```

Create a home folder for the sftp users

```
sudo mkdir /var/sftp
```

Now modify */etc/ssh/sshd_config*. Add *sftp-only* to allowed groups

```
AllowGroups <leave previous entries here and add => sftp-only
```

and add a Chroot rule

```
Match Group sftp-only
          ChrootDirectory /var/sftp/
          X11Forwarding no
          AllowTcpForwarding no
          ForceCommand internal-sftp -l VERBOSE
          AuthorizedKeysFile /var/sftp/%u/.ssh/authorized_keys
```

Finally, restart the ssh server

```
sudo systemctl restart sshd
```

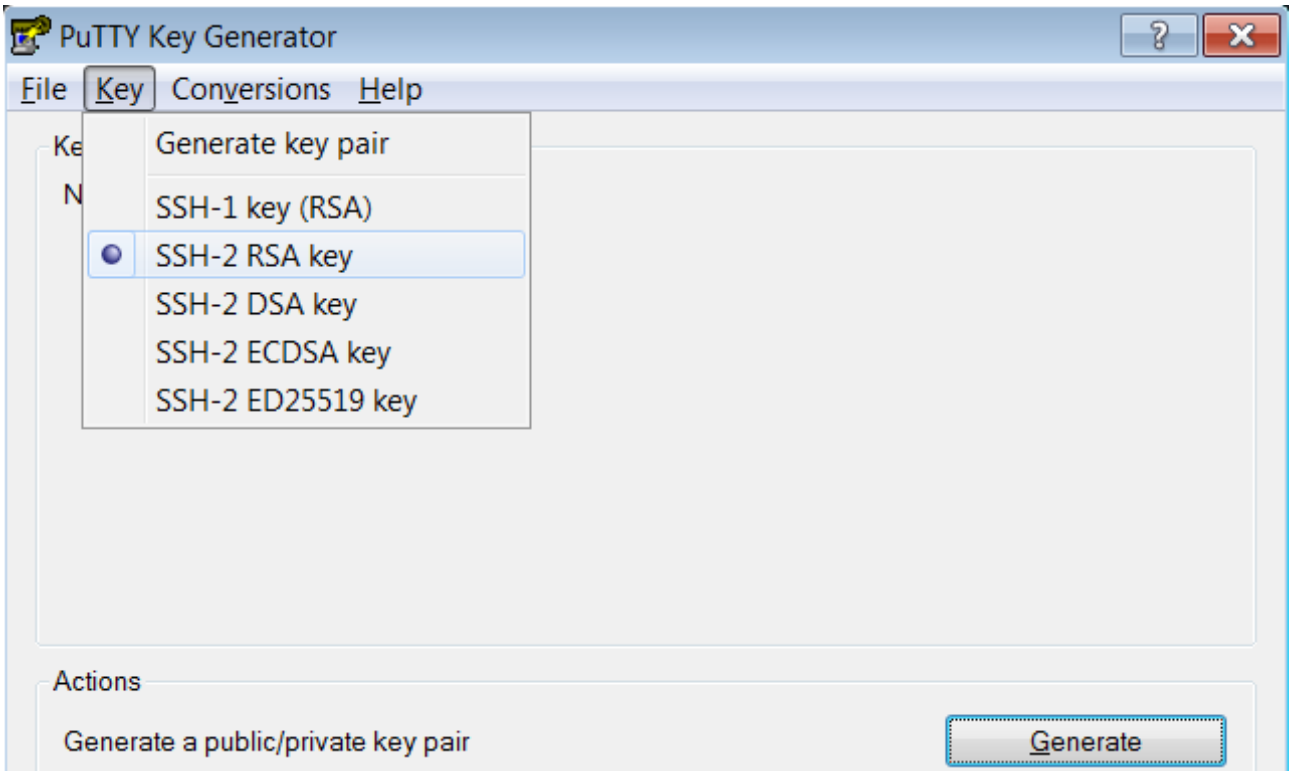
Appendix: Useful Putty Commands

Convert Putty public key into OpenSSH format (ignoring the comment tag, unfortunately):

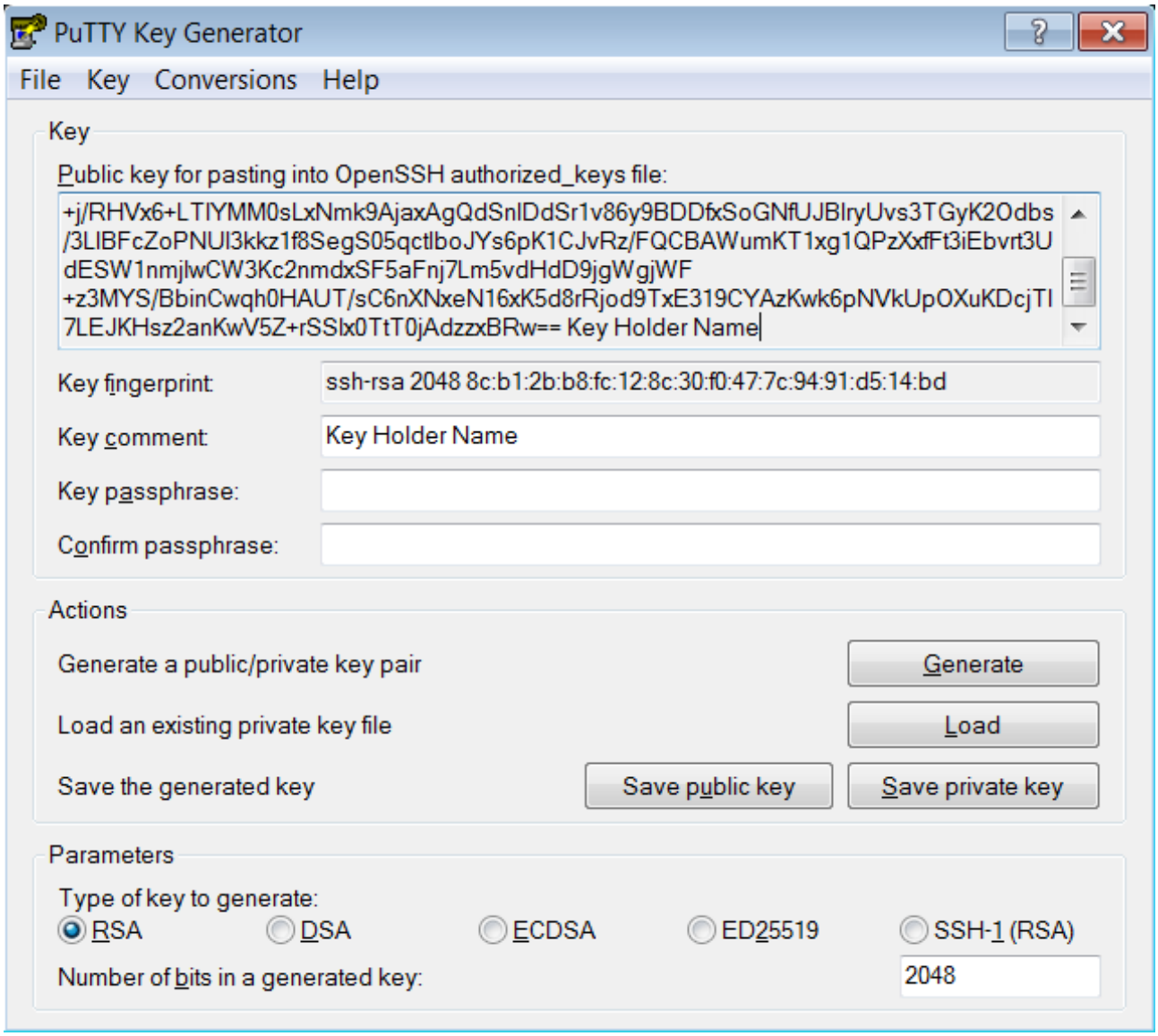
```
ssh-keygen -i -f Key_Holder_Name.puttypub > Key_Holder_Name.pub
```

Export OpenSSH public and private keys from Putty ppk file

```
puttygen Key_Holder_Name.ppk -O public-openssh
puttygen Key_Holder_Name.ppk -O private-openssh -o Key_Holder_Name
```



Generating keys in Putty.



Exporting Putty-generated OpenSSH keys.

Useful virsh commands

Document Author: arho.virkki@tyks.fi

```
# List running and all machines
sudo virsh list
sudo virsh list --all
```

```
# How much memory is allocated
sudo virsh domstats --balloon

# Only running machines
sudo virsh domstats --balloon | grep -e current -e Domain

# Only numbers (in KiB)
sudo virsh domstats --balloon | grep -e current | cut -d "=" -f 2

# Sum of all memory consumed (in KiB)
sudo virsh domstats --balloon | grep -e current | cut -d "=" -f 2 | paste -s -d+ | bc

# The same in (GiB)
sudo virsh domstats --balloon | grep -e current | cut -d "=" -f 2 |\
paste -s -d+ | bc | sed 's/$/\/1024\/1024/' | bc
```

Summary of Open Source Software

Operating Systems

- Physical servers: RHEL / CentOS 7 https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/
- Hypervisors: KVM https://en.wikipedia.org/wiki/Kernel-based_Virtual_Machine
- Virtual Machines: Ubuntu Linux https://www.ubuntu.com/desktop

Databases, data transformation and analysis

- Pentaho Kettle ETL-tool http://www.pentaho.com/product/data-integration
- PostgreSQL https://www.postgresql.org/docs/9.6/static/index.html
- Relational data modelling http://software.sqlpower.ca/page/architect
- Cloudera Hadoop https://www.cloudera.com/documentation.html
- R language (for statistical computing) https://www.r-project.org/
- Python 3 (for scientific computing and scripting) https://www.python.org/
- Unix Shell (for process automation) https://www.gnu.org/software/bash/manual/
- Machine learning: Weka https://www.cs.waikato.ac.nz/ml/weka/

Version control, documentation and backups

- Version control: Git https://git-scm.com/
- Wiki pages & notes: Markdown https://en.wikipedia.org/wiki/Markdown
- Kanban board: Wekan https://wekan.github.io/
- Backups: Cron & rsync https://en.wikipedia.org/wiki/Cron, https://en.wikipedia.org/wiki/Rsync

Publishing and Office

- Vector graphics: Inkscape https://inkscape.org/en/
- Image manipulation: Gimp https://www.gimp.org/
- Scientific publishing: LaTeX https://en.wikipedia.org/wiki/LaTeX

Remote Access

- OpenSSH https://en.wikipedia.org/wiki/OpenSSH
- X2Go https://wiki.x2go.org/doku.php
- Apache Guacamole (to be installed...) https://guacamole.apache.org/

Web-based User Interfaces

- Front-end layout: Twitter Bootstrap https://getbootstrap.com/
- Front-end widgets: jQuery and jQuery UI https://jquery.com/
- Interactive graphics: D3.js https://d3js.org/
- Back-end logic: Python/Flask http://flask.pocoo.org/