Managing servers with DSSH

29. 11. 2010

Juraj Bednár

Introduction

DIGMIA

- System administration and consulting company
  - Most of the TOP 20 web sites in Slovakia are our customers
- Supporters of open-source

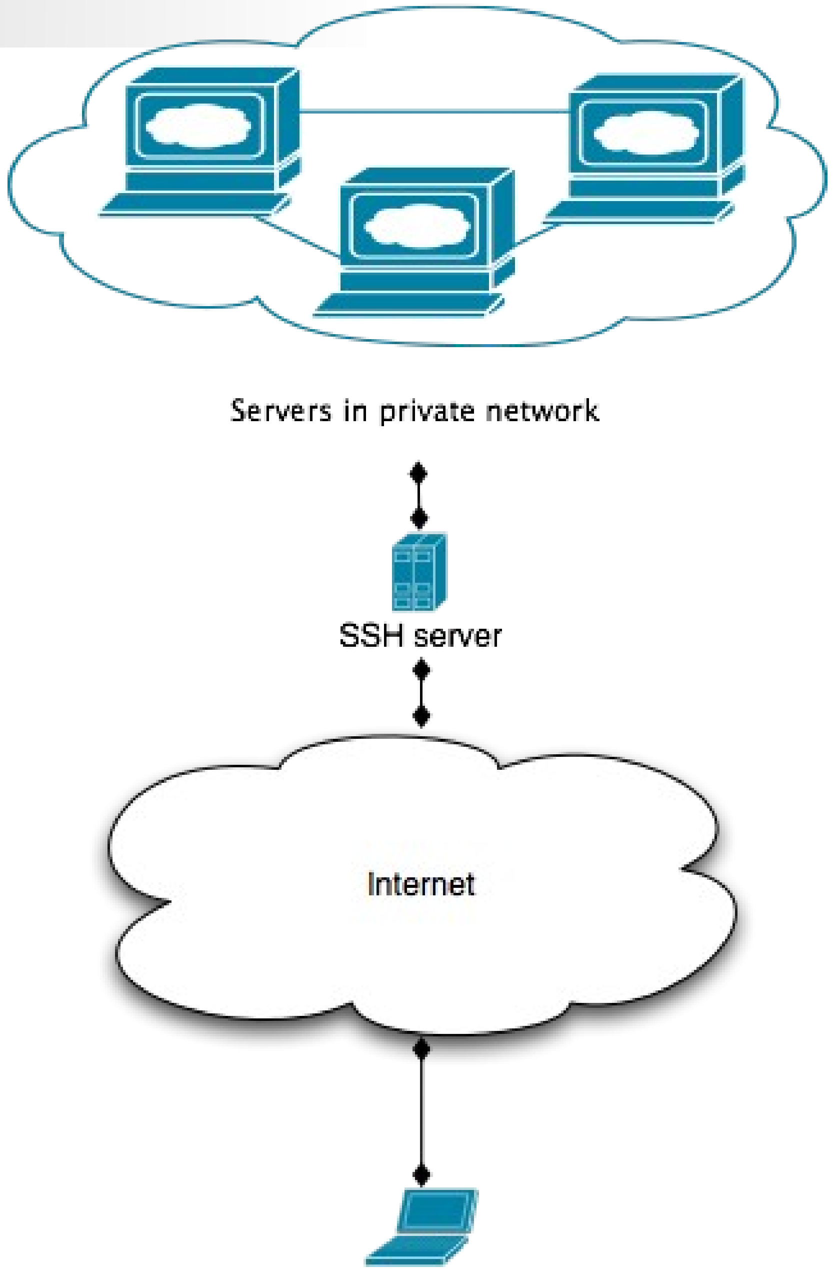
Me

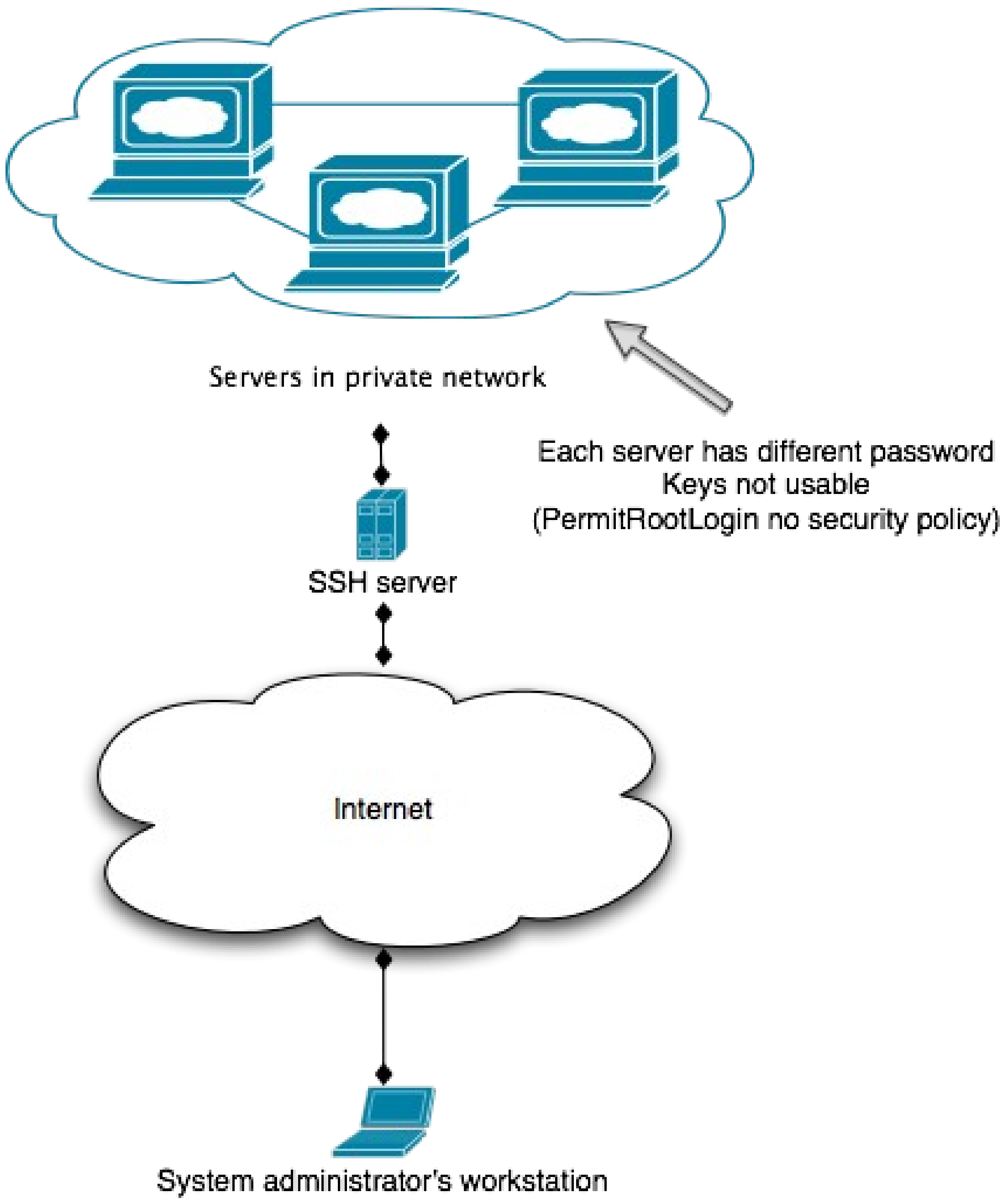
  - Co-founder of Progressbar.sk hackerspace
  - Member of Society for Open Information Technologies (soit.sk)

Who is this presentation for?

- At least 5 system administrators or
- At least 30 servers in heterogenous environment

Use case


Servers in private network
SSH server
Internet
System administrator's workstation


Servers in private network
Each server has different password
Keys not usable
(PermitRootLogin no security policy)
SSH server
Internet
System administrator's workstation

Wrong solutions

- Use ssh agent forwarding
  - Known to be insecure if you don't trust the server (which you should not – it's customer's, their security policy applies)
- Create VPN to office
  - Single point of failure
  - Difficult to manage if there are different customers with same network range (192.168.x.x)?

Wrong solutions

- Cut & paste passwords
  - Clipboard not safe enough
  - You don't need to display passwords, just use them at just the right place (don't paste to chat...)

Not applicable for us

- pfexec, sudo, ...
  - Low auditability
    - sudo -s
    - copy file to server and then execute
  - Management hell
    - Can not create lots of unix accounts and manage them
    - LDAP not possible (different customers, different security policies)

Enter DSSH

- Custom scriptable SSH client
- Written in Java, using modified Trilead SSH library
  - Console initialization components written in JNI
  - Needs terminal emulator (such as xterm or Terminal.app)
- Scriptable in BeanShell
  - Used Groovy, but it was too slow (interactive start)

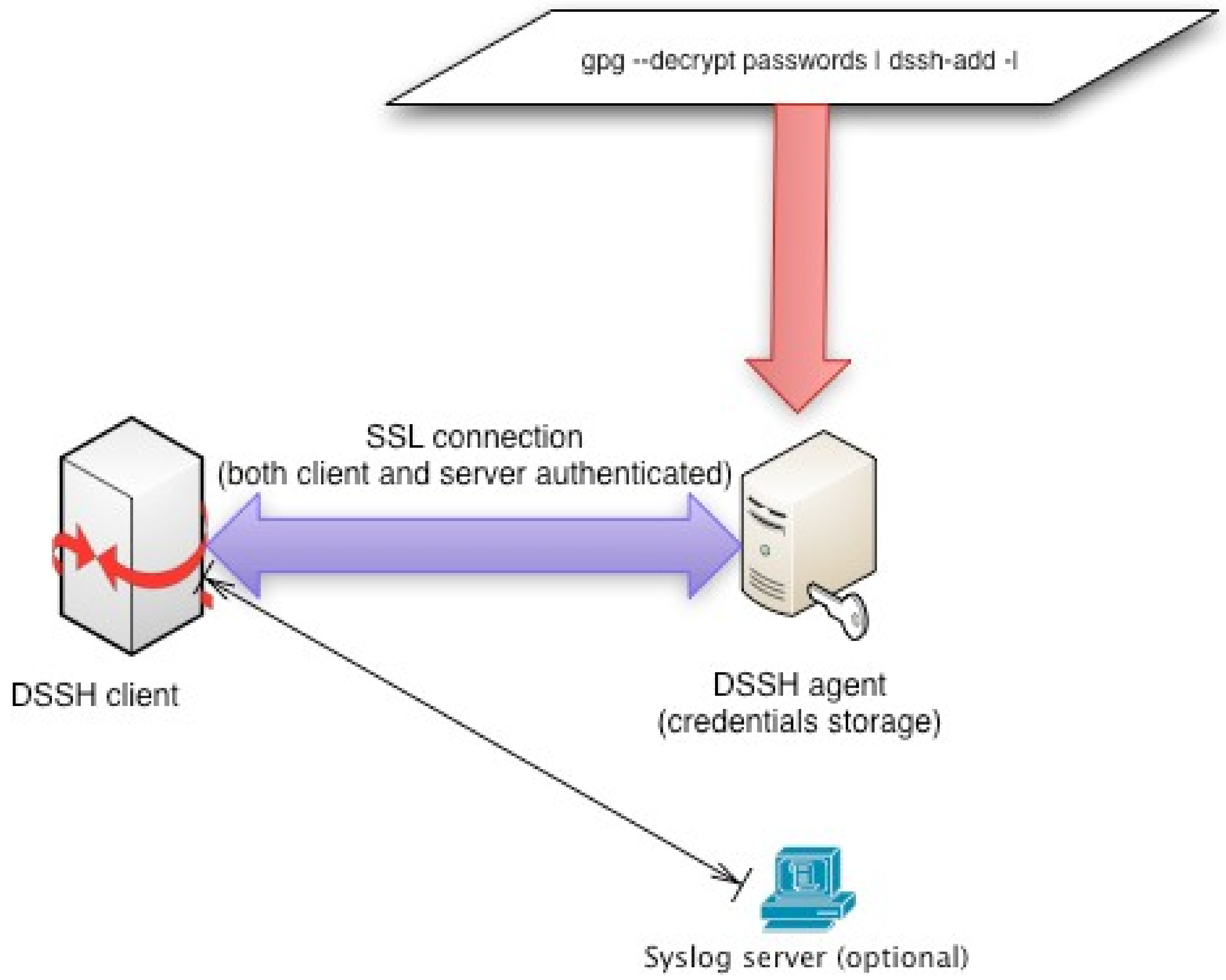
Features

- SSH in SSH tunneling
  - Hostnames can be interpreted by script to login you to target network
  - Possibility to change hostnames
- Possibility to login as root by using “su” or “ena”
- Limited scp support (sftp coming soon)
  - Not possible to scp using “su” or “ena” because of server lim.

Additional features

- Dynamic path selection (script can ping several entry point hosts)
- Logging support
- Credentials storage very lightweight
  - API does not support key retrieval (you can only use keys)
  - Supports password retrieval
  - Can be changed for any password storage solution easily

Architecture



Examples

- We don't want our admins to remember weird port numbers

```
if ((host.equals("weirdhost.customer1")) && (port == 22))
                port = 31337;
```

- Or IP addresses

```
          if (host.equals("weirdhost.customer2"))
               host = "192.146.122.211";
```

Examples

- Automatically use backup connection

```
  if (host.equals("weirdhost3.customer")) {
     InetAddress address = InetAddress.getByName(host);
       if (!address.isReachable(1500)) {
         if (verbose)
System.err.println("Unable to connect to weirdhost3.customer,
connecting to weirdhost3-1.customer instead");
                                host = "weirdhost3-1.customer";
         }}
```

Examples

- Use jumpstation (SSH in SSH tunelling)

```
  if (host.equals("weirdhost5.customer")) {
      parent = getAuthenticatedSSHConnection(myuser,
“gw.customer”, 22, parent, auth);
        }
```

- Additionaly you can create “virtual hostnames” by adding

```
 host = “192.168.2.3”;
```

Examples

- Security policy denies direct root logins
- In getAuthenticatedSSHConnection()

```
    if (host.equals("weirdhost.customer4") && user.equals("root"))
                    user = "digmia";
```

- In getInteractiveSession()

```
if (host.equals("weirdhost.customer4") && user.equals("root")
   return new InteractiveSuSession(conn.openSession(), host,
username, pass);
```

Examples

- Collect configurations from Cisco routers

```
for i in `cat dsshhostlist-cisco`
        do
                echo "Downloading configuration from $i"
                echo term len 0 $'\n' sh run $'\n' exit |
/usr/local/bin/dssh -k cisco/known_hosts ena@$i | sed -n
'/^[!]/,/^end/p' > cisco/$i
        done
```

Documentation and license

- Currently GPLv2
  - We consider to switching to less strict license (BSD)
- Documentation with examples available online
- Download at http://opensource.digmia.com/

Future

- Creating “DSSH server”
  - Standalone mode still possible
  - Lightweight client (possibly in Python)
    - Faster start-up times
    - Very little actual functionality
      - Possible to quickly steal terminal emulation from Putty and allow “xterm”-less Windows client
    - SSL + certificates

Future

  - Server allows additional features
    - Client never sees credentials
    - Server can check ACLs
    - Revoke certificates (no need to change all passwords, although this is scriptable thanks to DSSH :)
  - DSSH server side auditing
    - DSSH server sees into connections (even forwarded ones)
    - Logging of file transfers, port forwards, console, ...
- Server can be clustered (very little state information, this is really easy)

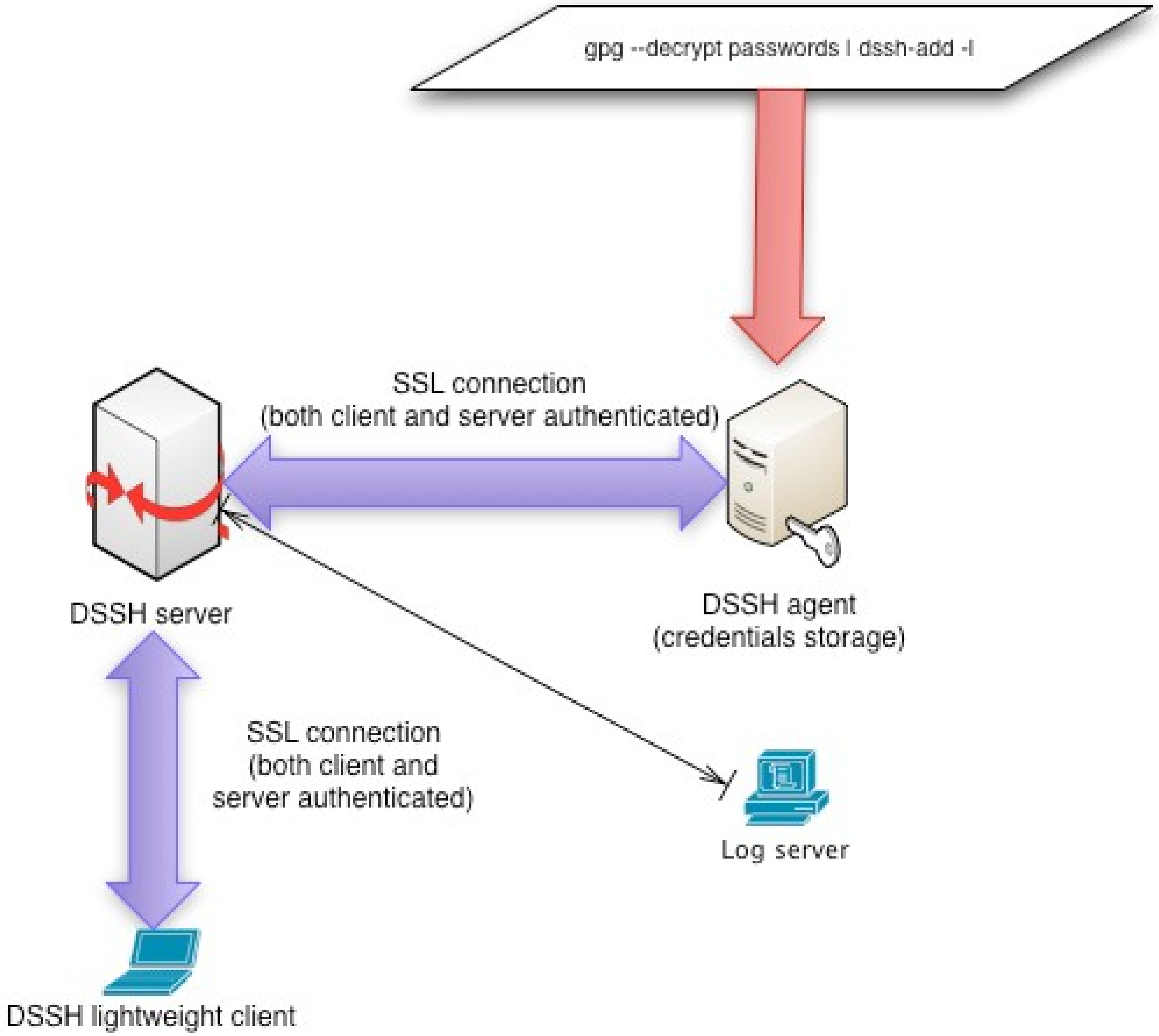
Future

- Admins never see credentials
  - No password leaks nor key leaks
  - SSL certificates can be revoked and exchanged easily
- Admins sometimes have to see credentials anyway (console root login to broken server)
  - Portal to request password and provide explanation
  - Automatically creates ticket to change password in trouble ticketing system
- SOCKS proxy –D (currently supports only –L a –R port forwarding)

Future

- Central server key authority
  - known_hosts idea is a main failure
  - Key never “just changes”, someone has to approve
  - Configurable policy, that does not ask the administrator, but just drops the connection

New architecture



Questions?

- Working on it right now
  - Q1-Q2 2011
  - You can help and shape the future of DSSH!
- Questions?
  - (and possibly some answers)