oVirt Host Deploy

Alon Bar-Lev
Red Hat

What is “Host deployment”?

- The process of preparing an operating system environment suitable to host virtual machines and to be managed by the ovirt back-end.
  - VDSM packages are installed.
  - Clock is 'soft' synchronized.
  - Management bridge created.
  - Firewall rules applied.
  - SSH trust obtained.
  - PKI trust obtained.
  - VDSM certificate issued.
  - Services' boot state set.
  - Host tuned for virtualization.

oVirt Host Deploy Artifacts

- Back-end side
  - Package: ovirt-host-deploy
  - Bundle: /usr/share/ovirt-host-deploy/interface-3
  - Cache: /var/cache/ovirt-engine/ovirt-host-deploy.tar
  - Logs: /var/log/ovirt-engine/engine.log
  - Logs: /var/log/ovirt-engine/host-deploy/*.log
- Host/ovirt-node side
  - /tmp/ovirt-host-deploy-*.log
- ovirt-node side
  - /var/log/vdsm-reg/*.log

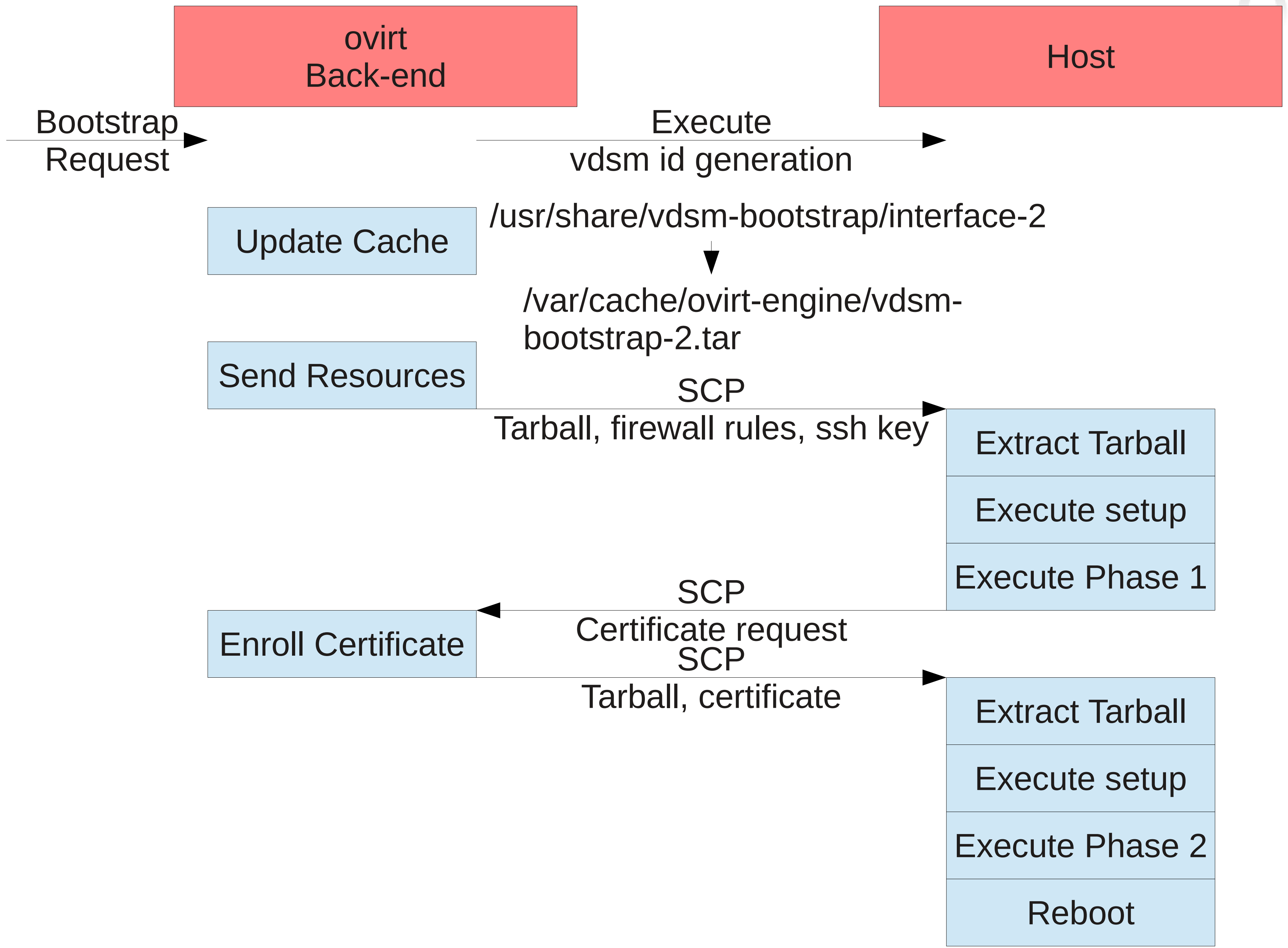
Parameters

- ServerRebootTimeout – Delay before communicating with VDSM.
- SSHInactivityTimeoutSeconds – Inactivity timeout.
- SSHInactivityHardTimeoutSeconds – Max bootstrap time.
- IPTablesConfig – iptables rules.
- BootstrapCacheRefreshInterval – as-is.

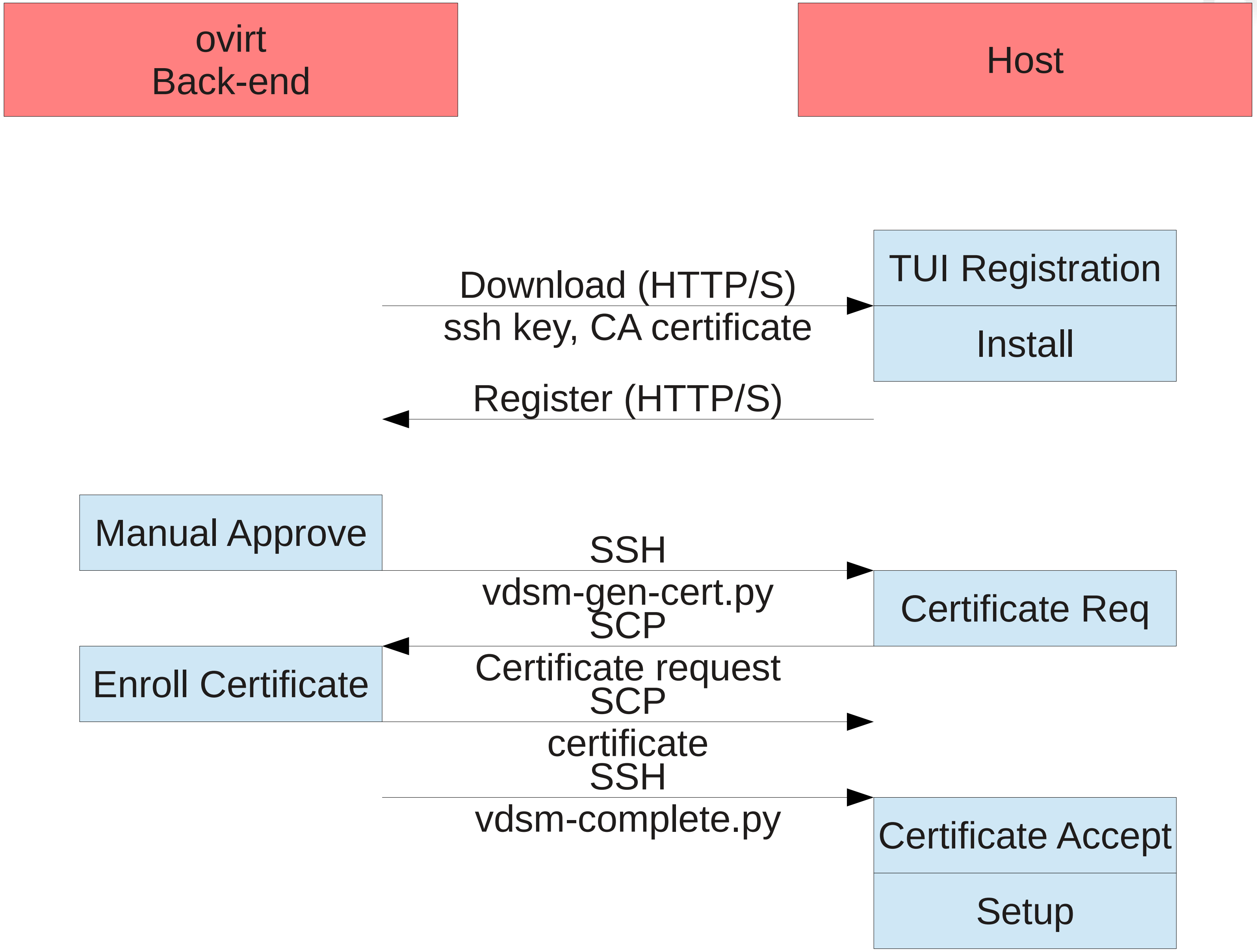
Legacy vdsm-bootstrap

- Types of “Bootstraps”:
  - Standard bootstrap
    - Handle standard RHEL/Fedora host.
  - ovirt-node registration
    - Handle ovirt-node when it is the initiator.
  - ovirt-node bootstrap
    - Handle ovirt-node when ovirt back-end is the initiator.
  - ovirt-node upgrade

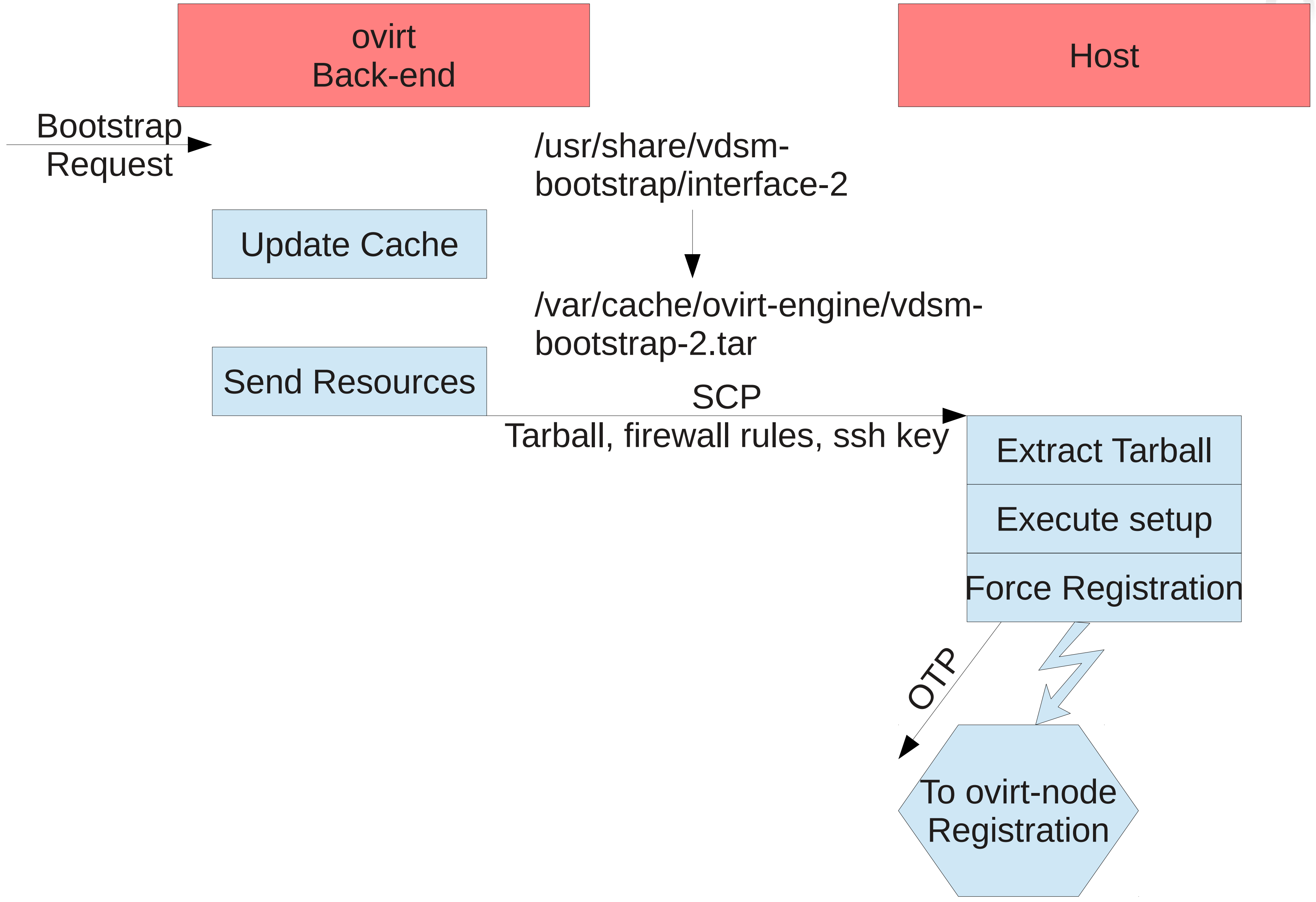
Standard Legacy Bootstrap Sequence



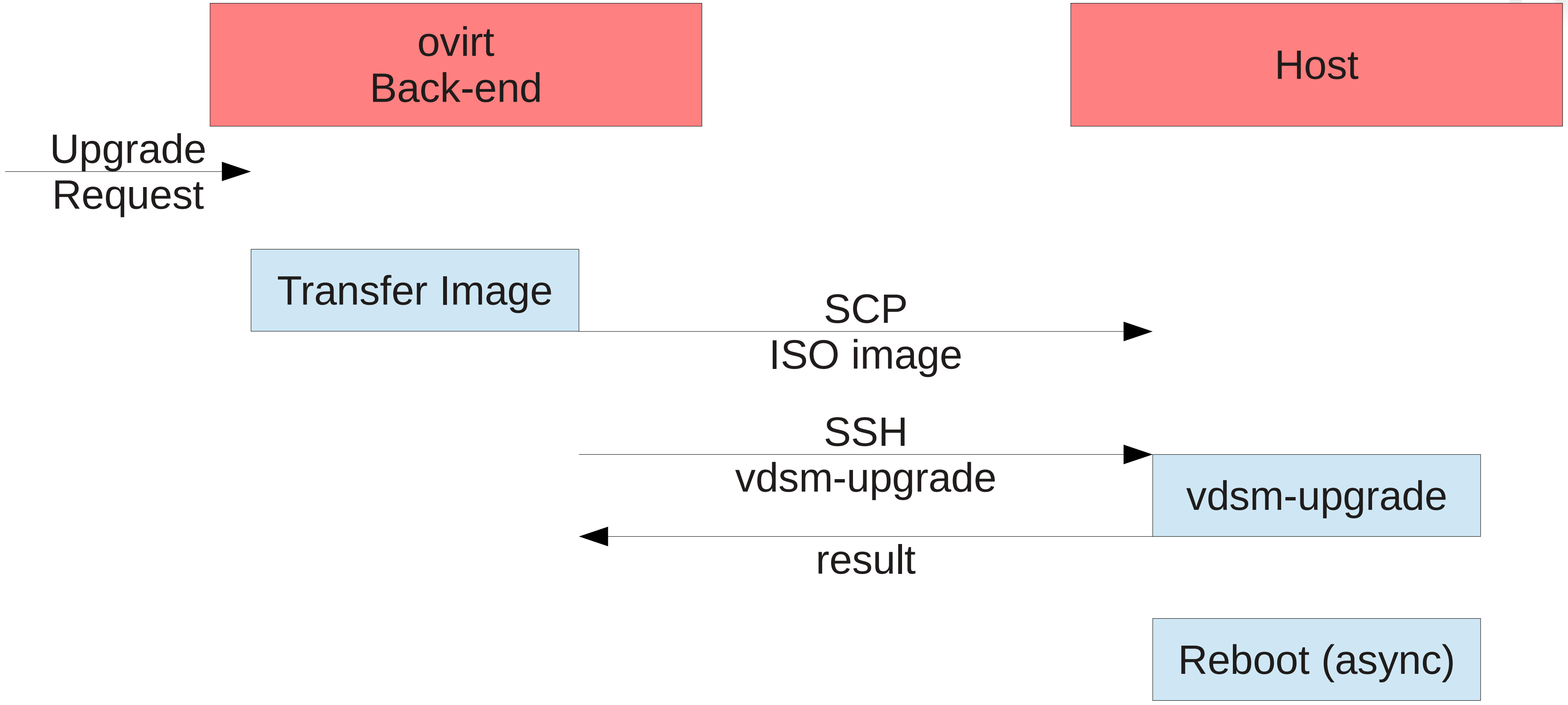
ovirt-node Legacy Registration Sequence



ovirt-node Legacy Bootstrap Sequence


ovirt Back-end
Host
Bootstrap Request
Update Cache
/usr/share/vdsm-bootstrap/interface-2
/var/cache/ovirt-engine/vdsm-bootstrap-2.tar
Send Resources
SCP
Tarball, firewall rules, ssh key
Extract Tarball
Execute setup
Force Registration
OTP
To ovirt-node Registration

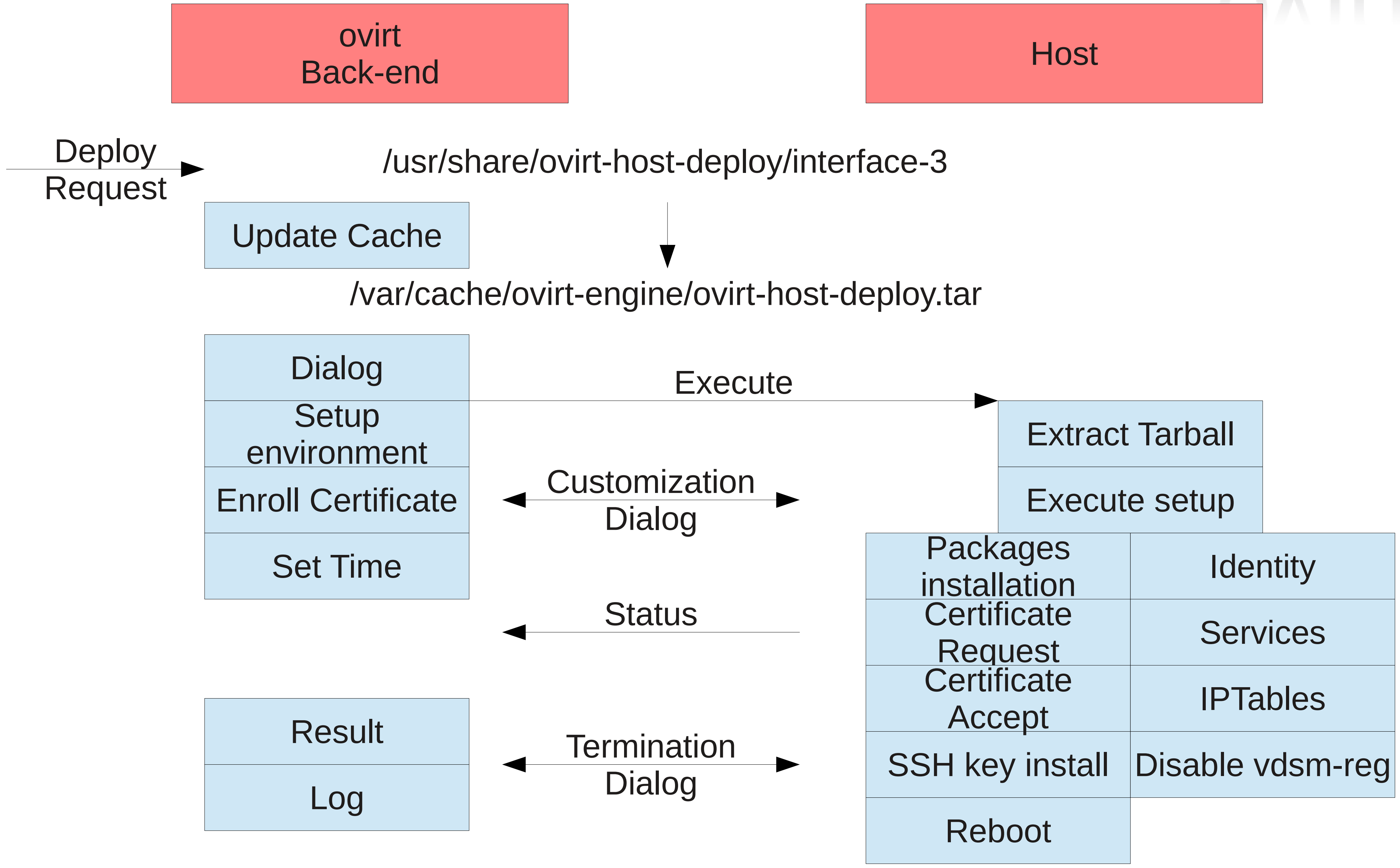
ovirt-node Upgrade Sequence



Legacy vdsm-bootstap - Issues

- Too many sequences.
- Too much complexity, can break at a lot of places.
- vdsm-bootstrap part of vdsm package, however it is back-end tool, hard to maintain the dependency and support multiple back-end versions.
- No manual invocation method.
- Logging is insufficient, example: stderr data absent from logs on host, but appear on back-end side.
- Monolithic design, hard to maintain.
- Interface is not formalized

ovirt-host-deploy Sequence



ovirt-host-deploy

- Standalone package.
- Two sequences
  - ovirt-host-deploy – standard and ovirt-node.
  - ovirt-node upgrade (legacy).
- Establish a bidirectional channel between back-end and host, no iterative commands nor file transfers.
- Manual invocation support.
- Better timeout management.
- Better logging.
- Pluggable implementation, easy to maintain and extend.

Implementation Details

- Common installation services
  - Package management
  - Service management
  - Transaction management
  - Sequence
  - Dialog

  Package: otopi
- oVirt specific deployment services
  - VDSM specific deployment.
  - oVirt specific deployment.

  Package: ovirt-host-deploy

Can also be used for other installation sequences, such as engine-setup, engine-upgrade and event openstack.

oVirt

otopi

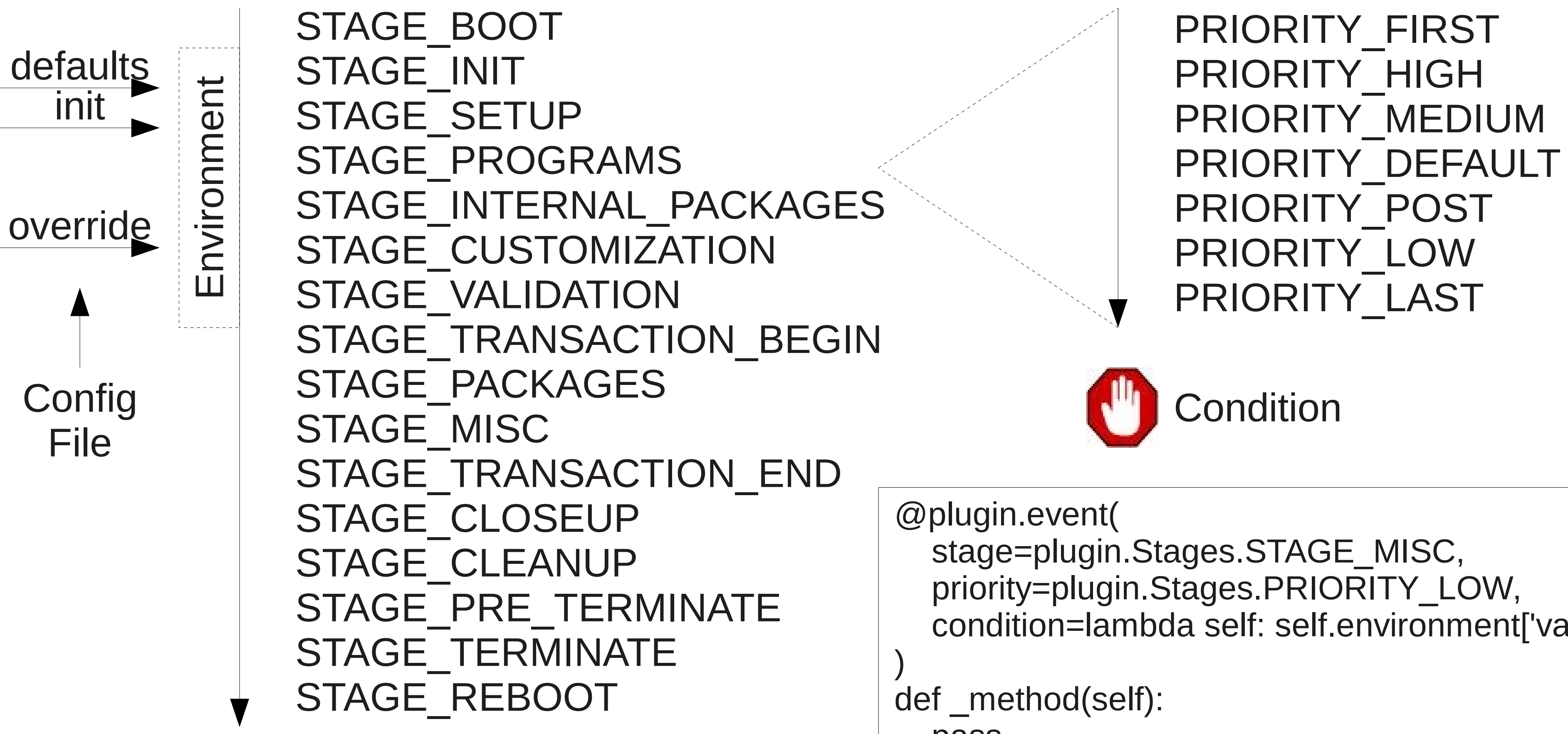
http://gerrit.ovirt.org/gitweb?p=otopi.git

otopi - oVirt Task Oriented Pluggable Installer/Implementation

- Standalone plugin based installation framework to be used to setup system components. The plugin nature provides simplicity to add new installation functionality without the complexity of the state and transaction management.
  - Modular, task oriented library implementation.
  - Supports pluggable manager dialog protocol, provides human and machine dialogs.
  - Localization support.
  - Local and remote execution modes are supported.
  - Distribution independent implementation (core).
  - Compatible with python-2.6, python-2.7, python-3.2

Installation Sequence

Developer provides plugins within plugin group.
Each plugin may register to events within stage by priority.
Plugin can provide condition callback.
State management is done via key/value environment.



STAGE_BOOT
STAGE_INIT
STAGE_SETUP
STAGE_PROGRAMS
STAGE_INTERNAL_PACKAGES
STAGE_CUSTOMIZATION
STAGE_VALIDATION
STAGE_TRANSACTION_BEGIN
STAGE_PACKAGES
STAGE_MISC
STAGE_TRANSACTION_END
STAGE_CLOSEUP
STAGE_CLEANUP
STAGE_PRE_TERMINATE
STAGE_TERMINATE
STAGE_REBOOT

PRIORITY_FIRST
PRIORITY_HIGH
PRIORITY_MEDIUM
PRIORITY_DEFAULT
PRIORITY_POST
PRIORITY_LOW
PRIORITY_LAST

```
@plugin.event(
    stage=plugin.Stages.STAGE_MISC,
    priority=plugin.Stages.PRIORITY_LOW,
    condition=lambda self: self.environment['var1'],
)
def _method(self):
    pass
```

Plugin Groups

- Plugins are groups by 'plugin group'.
- Groups can be enabled, so its plugins are loaded. Done via environment:

  APPEND:BASE/pluginGroups=str:group1

- File system structure:

```
/usr/share/otopi/plugins
    <plugin group>
        <plugin>
            Plugin sources
```

```
from otopi import util
from . import plugin1
@util.export
def createPlugins(context):
    plugin1.Plugin(context=context)
```

- Package management friendly.

Environment (partial)

BASE/aborted(bool)
Aborted by user.
Will also have error.

BASE/debug(int) [0]
Debug level.

BASE/error(bool)
Error during sequence.

BASE/pluginGroups(str)
Plugin groups to load. ':' separated.

CORE/logDir(str) [${TMPDIR}]
Log file directory.

CORE/logFileName(str)
Log file name.

CORE/configFileName(str) [/etc/otopi.conf]
Configration file name.

DIALOG/dialect(str) [human]
Dialect to use.

DIALOG/customization(bool) [False]
Enable customization

DIALOG/cliVersion
Command line interface version.

SYSTEM/clockSet(bool) [False]
Synchronize clock.

NETWORK/sshEnable(bool) [False]
Enable ssh key storage.

NETWORK/sshKey(str)
SSH public key.

NETWORK/sshUser(str)
SSH user or current.

NETWORK/iptablesEnable(bool) [False]
Enable set of iptables.

NETWORK/iptablesRules(multi-str)
iptables content.

PACKAGER/yumpackagerEnabled(bool) [True]
Enable yum packager.

PACKAGER/keepAliveInterval(int) [30]
Keep alive interval for status in seconds.

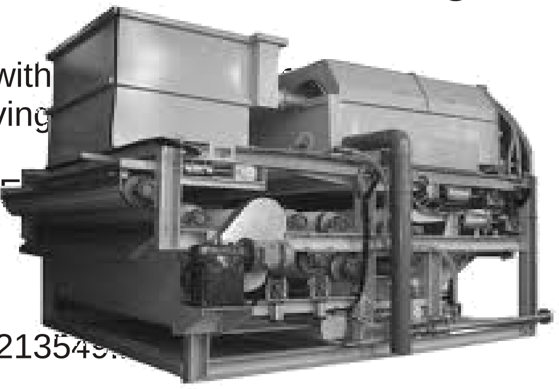
Dialog

- Interface between plugins and the outside world.
- Simple interaction:
  - Terminate
  - Note
  - Queries
    - String
    - Multi-string
    - Value
  - Display
    - Multi-String
    - Value
  - Confirm

Human Dialog

```
[ INFO  ] Stage: Initializing
          Continuing will configure this host for serving as hypervior.Are you sure
you want to continue? (yes/no) yes
[ INFO  ] Stage: Environment setup
          Log file: /tmp/ovirt-host-deploy-20121121213304.log
          Version: otopi-0.0.0
          Configuration files: ['/etc/ovirt-host-deploy.conf.d/50-offline-
packager.conf']
[ INFO  ] Stage: Installation packages setup
```

Machine Dialog



```
***L:INFO Stage: Initializing
***CONFIRM DEPLOY_PROCEED Proceed with
### Continuing will configure this host for serving
you want to continue? (yes/no)
### Response is CONFIRM DEPLOY_PROCE
DEPLOY_PROCEED
CONFIRM DEPLOY_PROCEED=yes
***L:INFO Stage: Environment setup
### Log file: /tmp/ovirt-host-deploy-2012112121354
### Version: otopi-0.0.0
### Configuration files: ['/etc/ovirt-host-deploy.conf.d/50-offline-packager.conf']
***L:INFO Stage: Installation packages setup
```

Customization and termination dialog

- Enabled using DIALOG/customization=bool:True.
- A simple command-line will be available during customization and pre-terminate stages.
- Mainly used to manipulate environment.
- Available commands:

```
abort - Abort process
env-get - Get environment variable
env-query - Query environment variable
env-query-multi - Get multi string environment variable
env-set - Set environment variable
env-show - Display environment
exception-show - show exception information
help - Display available commands
install - Install software
log - Retrieve log file
noop - No operation
quit - Quit
```

Generic Plugins

- Services
  - systemd
  - rhel
  - openrc
- Network
  - hostname
  - ssh
  - iptables
- System
  - clock
  - command
  - info
  - reboot
- Packagers
  - yumpackager

otopi Bundle

- A bundle is a set of files prepared to be archived and sent to remote host.
- Except of python, no other software is required at destination host.
- Bundle structure:

```
.bundled
otopi -> ../../../sbin/otopi
otopi-plugins
    otopi -> ../../../otopi/plugins/otopi
pythonlib
    otopi -> ../../../../lib64/python3.2/site-packages/otopi
```

Resources

- Documentation
  - README
  - README.API
  - README.dialog
  - README.environment
- Packages
  - otopi – base package.
  - otopi-java – java constants and dialog parser.
  - otopi-devel – development tools.
- Java Artifacts
  - org.ovirt@oss.sonatype.org

ovirt-host-deploy

http://gerrit.ovirt.org/gitweb?p=ovirt-host-deploy.git

ovirt-host-deploy

- Implemented over otopi.
- A set of otopi plugins.
- Anything that is specific to ovirt host deployment.

ovirt-host-deploy plugins

- core/misc
  - General installer environment setup.
- core/offlinepackager
  - Offline packager used by ovirt-node and all-in-one.
- vdsm/bridge
  - Management bridge setup.
- vdsm/config
  - vdsm.conf manipulation.
- vdsm/hardware
  - Hardware prerequisites.

ovirt-host-deploy plugins

- vdsm/packages
  - Packages installation.
- vdsm/pki
  - PKI setup.
- vdsm/software
  - Software prerequisites.
- vdsm/tuned
  - Tuned setup.
- vdsm/vdsmid
  - VDSM id generation.
- vdsmhooks/hooks
  - Installation of vdsm hooks.

ovirt-host-deploy plugins

- node/detect
  - ovirt-node detection.
- node/persist
  - ovirt-node file persistence.
- node/vdsm_reg
  - ovirt-node's vdsm-reg handling.
- gluster/packages
  - Gluster package installation.

Resources

- Documentation
  - README
  - README.environment
  - ChangeLog
- Packages
  - ovirt-host-deploy – base package.
  - ovirt-host-deploy-java – java constants.
  - ovirt-host-deploy-offline – offline dependencies and setup.
- Java Artifacts
  - org.ovirt@oss.sonatype.org

Engine Side

Sources

- org.ovirt.engine.core.utils.ssh.**SSHDialog**
  - New class to handle interactive SSH dialog.
- org.ovirt.engine.core.utils.ssh.**EngineSSHDialog**
  - Extends SSHDialog with engine defaults.
- org.ovirt.engine.core.bll.**OVirtNodeUpgrade**
  - Rewrite of the legacy OVirtUpgrader using SSHDialog.
- org.ovirt.engine.core.bll.**VdsDeploy**
  - Rewrite of OVirtInstaller, VdsInstaller, VdsInstallerSSH.
  - Implementation using SSHDialog, otopi and ovirt-host-deploy.
  - Uses **MachineDialogParser** from otopi.

oVirt

Demonstration

oVirt

Summary

Summary

- ovirt-host-deply package is a complete rewrite of the legacy vdsm-bootstrap package.
  - Modular.
  - Pluggable.
  - Coherent.
  - Single sequence.
- otopi project spin out to enable reuse within other projects.
- Expectation: lower cost of maintenance, easy to deploy more features.

otopi/ovirt-host-deploy

- Questions?