[Handout] Contributing to the Insights Core Framework

Sachin Patil, Vishwanath Jadhav*

DevConf.in 2019, Bangalore, India

This handout will guide you in developing and understanding the parser and the rule(plugin) using the **Insights Core Framework**. The rule uses the simple logic to analyze the sosreport and verify if the `root` login is enabled in `/etc/ssh/sshd_config` file. If the `root` is permitted to login to the SSH server, the rule will respond with the distro name(product) along with the resolution.

Start by installing the pre-requisites and creating the Python Virtual environment needed for developing & testing a rule. Once the virtual environment is ready, create an empty Python file with any name(say `my_plugin.py`) and start adding the code snippets marked as **CODE** in the python file to develop a complete rule. You can literally copy-paste the code snippets to get the working rule.

1 Prerequisites

- GNU/Linux
- Python 3.6
  - Fedora
    ```
    $ sudo dnf install python36
    ```
  - CentOS
    ```
    $ sudo yum install epel-release centos-release-scl
    $ sudo scl enable rh-python36 bash
    ```
  - Ubuntu
    ```
    $ sudo apt-get install python3-venv
    $ pip install wheel
    ```
- `git`[1]
- `sos-report`
  - Fedora
    ```
    $ sudo dnf install sos
    ```
  - CentOS/RedHat
    ```
    $ sudo yum install sos
    ```
  - Ubuntu
    ```
    $ sudo apt-get install sosreport
    ```

2 Setup

- Setting up a development environment

```
$ mkdir ~/insights
$ cd ~/insights
# Clone the Insights Core repo
$ git clone https://github.com/RedHatInsights/insights-core.git
$ python3.6 -m venv .
```

*psachin@redhat.com, vjadhav@redhat.com

[1] https://git-scm.com/book/en/v2

```
$ source bin/activate
$ pip install -e insights-core[develop]

# [Optional] Clone the plugin repo
$ git clone https://github.com/vishwanathjadhav/analysis-plugins.git
```

3 Sos-report[2]

- Generating an Sos archive

```
$ sudo sosreport

# Below command is recommended for this demo.
$ sudo sosreport -o ssh,systemd,release
```

4 Specs[3]

- Raw data in the form of file-content or command output.

```
# File
$ cat /etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04.2 LTS"

# Command output
$ uname -a
Linux foobar 5.0.17-200.fc29.x86_64 #1 SMP Mon May...2019 x86_64...GNU/Linux
```

- **CODE**: Define a Spec

```
"""The specs is where you define a path to the file(configuration, log etc.)
having the content or the command output within the sos-report. The valid path
can also be a file-system path such as ‘‘/var/log/messages‘‘.
"""
from insights.specs import Specs
from insights.core.spec_factory import simple_file


class SosSpecs(Specs):
    # sos-archive/etc/lsb-release.

    # You can safely skip the ’sos-archive/’ as the rule will be run against an
    # archive.
    lsb_release = simple_file("etc/lsb-release")
```

5 Parser[4]

- It structures the raw data(from specs) for further analysis.
- Example(structured data of `/etc/lsb-release` returned by the parser):

```
{
    ’product’: ’Ubuntu’,
    ’version’: ’18.04’
}
```

[2] https://github.com/sosreport/sos/wiki#for-users

[3] https://insights-core.readthedocs.io/en/latest/api.html#specification-factories

[4] https://insights-core.readthedocs.io/en/latest/api.html#parsers

- **CODE**: Define a parser

```
from insights import Parser, parser
from insights.parsers import split_kv_pairs


@parser(SosSpecs.lsb_release)
class LsbRelease(Parser):
    def parse_content(self, content):
        _content = split_kv_pairs(content)
        self.data = {
            ’product’: _content[’DISTRIB_ID’],
            ’version’: _content[’DISTRIB_RELEASE’]
        }

    @property
    def product(self):
        return self.data[’product’]

    @property
    def version(self):
        return self.data[’version’]
```

6 Plugin/Rule

6.1 Conditions[5]

- **CODE**: Analyzing the data from the parser

```
"""The (structured)data from the parsers is analyzed using the ‘‘@condition‘‘ &
‘‘@incident‘‘ decorators. The function decorated with the ‘‘@condition‘‘ should
return the value which will be used by the ‘‘@rule‘‘ to finally bind all the
rule logic.
"""
from insights.core.plugins import condition
from insights.parsers.ssh import SshDConfig
from insights.parsers.systemd.unitfiles import ListUnits
from insights.parsers.redhat_release import RedhatRelease

@condition(SshDConfig)
def check_permit_root_login(sshd):
    """Return True if ’PermitRootLogin yes’ in /etc/ssh/sshd_config.
    """
    if sshd.get(’permitrootlogin’):
        return sshd.get_values(’permitrootlogin’)[0] == ’yes’


@condition(ListUnits)
def is_sshd_running(units):
    """ Return True if ‘‘sshd.service‘‘ is running.
    """
    return units.is_running(’sshd.service’) or units.is_running(’ssh.service’)


@condition([RedhatRelease, LsbRelease])
def get_release(redhat_release, lsb_release):
    """Return the product name.

    RedhatRelease will parse the data from ‘‘/etc/redhat-release‘‘
```

[5] https://insights-core.readthedocs.io/en/latest/api_index.html?highlight=%40condition#insights.core.plugins.condition

```
    LsbRelease will parse the data from ‘‘/etc/lsb-release‘‘
    """
    if redhat_release:
        return redhat_release.product
    if lsb_release:
        return lsb_release.product
```

6.2 The response[6]

- **CODE**: The `@rule` will have a final logic which will decide the response of the plugin.

```
"""The function decorated with the ‘‘@rule‘‘ decorator is where all the logic to
detect an issue exist. The logic for this rule is as follows:

Logic:
1. check_permit_root_login
2. is_sshd_running
3. get_release

if (2 & 3):
    if 1:
        # The root user login is permitted
    # The root user login is disabled

"""
from insights.core.plugins import make_fail, make_pass, rule


@rule(check_permit_root_login, is_sshd_running, get_release)
def report(root_login, sshd, release):
    if sshd and release:
        if root_login:
            # The issue is detected.
            return make_fail(’SSHD_ROOT_LOGIN_PERMITTED’,
                             os=release)
        # The issue does not exist.
        return make_pass(’SSHD_ROOT_LOGIN_DISABLED’,
                         os=release)
```

6.3 Embedded content[7]

- **CODE** Use the `CONTENT` attribute to render the response

```
fail_message = """
The root user can login on this {{os}} host because the ’PermitRootLogin’ is set
to ’yes’ in /etc/ssh/sshd_config.

It is recommended to set ’PermitRootLogin’ to ’prohibit-password’,
’forced-commands-only’ or ’no’.

Please refer the manpage of SSHD_CONFIG for more info:
$ man 5 ssh_config
"""

pass_message = """
The root user cannot login on this {{os}} host.
"""

CONTENT = {
```

[6] https://insights-core.readthedocs.io/en/latest/api.html?highlight=%40rule#rule-plugins

[7] https://insights-core.readthedocs.io/en/latest/embedded_content.html

```
    'SSHD_ROOT_LOGIN_PERMITTED': fail_message,
    'SSHD_ROOT_LOGIN_DISABLED': pass_message
}
```

7 Investigating the sos-report[8]

- Run the plugin against the sos-report using:

  ```
  $ insights-run -p my_plugin.py /path/to/sos-report.tar.xz
  ```

- Sample output

```
$ insights-run -p check_ssh_root_login.py sosreport_fedora_sshd_root_login.tar.xz
---------
Progress:
---------
F

--------------
Rules Executed
--------------
[FAIL] check_ssh_root_login.report
---------------------------------

The root user can login on this Fedora host because the 'PermitRootLogin' is set
to 'yes' in /etc/ssh/sshd_config.

It is recommended to set 'PermitRootLogin' to 'prohibit-password',
'forced-commands-only' or 'no'.

Please refer the manpage of SSHD_CONFIG for more info:
$ man 5 ssh_config


----------------------
Rule Execution Summary
----------------------
Passed      : 0
Failed      : 1
Info        : 0
Missing Deps: 0
Fingerprint : 0
Metadata    : 0
Metadata Key: 0
Exceptions  : 0
```

8 [WIP]Debugging[9]

- Specs

```
$ insights-inspect insights.specs.default.DefaultSpecs.redhat_release sosreport_xxx.tar.xz

IPython Console Usage Info:

Enter 'redhat_release.' and tab to get a list of properties
Example:
In [1]: redhat_release.<property_name>
Out[1]: <property value>
```

[8] https://insights-core.readthedocs.io/en/latest/manpages/insights-run.html

[9] https://insights-core.readthedocs.io/en/latest/manpages/insights-inspect.html

```

To exit ipython enter 'exit' and hit enter or use 'CTL D'

Starting IPython Interpreter Now

In [1]: redhat_release.content
Out[1]: ['Fedora release 29 (Twenty Nine)']

In [2]: redhat_release.path
Out[2]: '/tmp/insights-rdhi53c6/sosreport_fedora_sshd_root_login/etc/redhat-release'

In [3]: redhat_release.file_name
Out[3]: 'redhat-release'
```

- Parser:

```
$ insights-inspect insights.parsers.redhat_release.RedhatRelease sosreport_xxx.tar.xz

IPython Console Usage Info:

Enter 'RedhatRelease.' and tab to get a list of properties
Example:
In [1]: RedhatRelease.<property_name>
Out[1]: <property value>

To exit ipython enter 'exit' and hit enter or use 'CTL D'

Starting IPython Interpreter Now

In [1]: RedhatRelease.raw
Out[1]: 'Fedora release 29 (Twenty Nine)'

In [2]: RedhatRelease.parsed
Out[2]: {'product': 'Fedora', 'version': '29', 'code_name': 'Twenty Nine'}

In [3]: RedhatRelease.product
Out[3]: 'Fedora'

In [4]: RedhatRelease.version
Out[4]: '29'
```

9 End[10], [11], [12]

[10] Made with Love, LaTeX and GNU Emacs.

[11] The code snippets are tested on *Fedora 29* & *Ubuntu 10.04*.

[12] For Education purpose only.