The new Check-API

CHECKMK CONFERENCE #5 – MUNICH, APRIL 30, 2019

tribe29

Agenda

1. **WHY – THE RATIONALE TO BUILD A CHECK-API**
2. WHAT AND HOW – THE PRINCIPLES BEHIND THE CHECK-API
3. NEXT STEPS FOR THE CHECK-API

tribe29

First, a look at a simple check...

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

OK, inventory. But didn't we read something about discovery somewhere?

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

Now we check something. Ok, good.

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1’,
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

3rd block. Wtf is a check_info? And where does it come from?

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None
}
```

We got Lambdas!!!

snmp_info check

4? Why shouldn't we try 3 or 5?

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                           is not None,
}
```

Hello! It’s here.

snmp_info check

The check item. You know?

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

snmp_info check

No discovered parameters. Got me?

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

But the checking is understandable?

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, para
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

These 4 elements again!

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info
    'check_function':
    'inventory_function':     inventory_snmp_info,
    'service_description':    'SNMP Info',
    'snmp_info':              ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':     lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

We got 3 now!

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "                a retrieved")

    'inventory_function':     inventory_snmp_info,
    'service_description':    'SNMP Info',
    'snmp_info':              ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':     lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

And here is a 0. Is it better than 3?

snmp_info check

```
def inventory_snmp_info(info):
    if len(info[0]) >= 4:
        return [ (None, None) ]

def check_snmp_info(checktype, params, info):
    if len(info[0]) >= 4:
        return (0, ', '.join([ info[0][i] for i in [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


check_info["snmp_info"] = {
    'check_function':          check_snmp_info,
    'inventory_function':      inventory_snmp_info,
    'service_description':     'SNMP Info',
    'snmp_info':               ('.1.3.6.1.2.1.1',
                                                    ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':      lambda oid: oid(".1.3.6.1.2.1.1.1.0")
                                                    is not None,
}
```

["a", "c", "d", "b"] turns into "a, b, c, d"

From few to 1700 plugins, API hasn't changed

Past

30 plugins + **Check-API v1**

Present

1700 plugins + 3rd party checks + **Check-API v1** + some more functions

A lot of room for improvement

Current Issues

- What should be handled by a check, discovery or parse function?
- A lot of redundant code
- What is included? Where are imports? Which python modules can I use?
- Where does check_info come from?
- A lot of things happen implicitly
- Poor API documentation, changes communicated by werks

Results

- Understanding the structure of a plugin is not straight-forward
- Learning happens by looking at existing plugins
- Writing plugins is error prone and takes longer than it should

Agenda

1. WHY – THE RATIONALE TO BUILD A CHECK-API
2. **WHAT AND HOW – THE PRINCIPLES BEHIND THE CHECK-API**
3. NEXT STEPS FOR THE CHECK-API

tribe29

Overarching goal: A new way to write plugins

Restructure Check Plugins

Goal

- ✔ Check plugins are modules
- ✔ Plugins import a versioned, well-defined API

```
--def inventory_snmp_info(info):
    if len(info[0]) >= 4:
--      return [ (None, None) ]

--def check_snmp_info(checktype, params, info):
--    if len(info[0]) >= 4:
--        return (0, ', '.join([ info[0][i] for i in
--                        [0, 2, 3, 1] if info[0][i] ]))
    else:
        return (3, "No data retrieved")


  # This check works on all SNMP hosts
>>check_info["snmp_info"] = {
    'check_function':        check_snmp_info,
    'inventory_function':    inventory_snmp_info,
    'service_description':   'SNMP Info',
    'snmp_info':             ('.1.3.6.1.2.1.1',
                              ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':    lambda oid:
                             oid(".1.3.6.1.2.1.1.1.0") is not None,
  }
  ~
  ~
  ~
  ~
  ~
  ~
  ~
  ~
  ~
NORMAL  check_before        4% ☰   1/25 ln : 17  W:12(L1)  E:1(L17)
```

```
--def inventory_snmp_info(info):
--    if len(info[0]) >= 4:
--        return [ (None, None) ]
--
--def check_snmp_info(checktype, params, info):
--    if len(info[0]) >= 4:
--        return (0, ', '.join([ info[0][i] for i in
--                               [0, 2, 3, 1] if info[0][i] ]))
--    else:
--        return (3, "No data retrieved")


# This check works on all SNMP hosts
>>check_info["snmp_info"] = {
    'check_function':        check_snmp_info,
    'inventory_function':    inventory_snmp_info,
    'service_description':   'SNMP Info',
    'snmp_info':             ('.1.3.6.1.2.1.1',
                              ['1.0', '4.0', '5.0', '6.0']),
    'snmp_scan_function':    lambda oid:
                             oid(".1.3.6.1.2.1.1.1.0") is not None,
}
~
~
~
~
~
~
~
~
~
~
NORMAL check_before  4% ☰  1/25 ln : 17  W:12(L1)  E:1(L17)
```

```
from cmk.check_api.v1 import (
    SNMPSection,
    Check,
    Output,
    check_api,
)


class SectionSNMPInfo(SNMPSection):
    name = "snmp_info"

    def scan(self):
        return self.has_oid(".1.3.6.1.2.1.1.1.0")

    def snmp_oids(self):
        return ('.1.3.6.1.2.1.1', ['1.0', '4.0', '5.0', '6.0'])


@check_api.register(name="snmp_info")
class CheckSNMPInfo(Check):
    Sections = [SectionSNMPInfo]
    description = 'SNMP Info'

    @discover
    def discover(self, line):
        return len(line) >= 4

    def check(self, _no_item, info):
        for i in (0, 2, 3, 1):
            yield Output(info[0][i])
~
N… check_after.py[+]  pyt…  3% ☰  1/33 ln :  1
```

But there are things to consider…

Restructure Check Plugins

- Support of legacy Check-API
- Documentation for transition
  (what features are not supported anymore)
- Migration of legacy plugins to new plugins
- Handling of identical legacy vs. new checks

So for now we will focus on...

Reduce boilerplate

Reduce technical debt

Improve usability

Restructure checks

Some examples of what are we doing

Goals

- **Reduce boilerplate**
- **Reduce technical debt**
- **Improve usability**

Some optimization examples

- Function to unpack parsed data
- Wrapping values in objects
- Handling of check plugin returns

And many more...

Some examples of what are we doing

Goals

- **Reduce boilerplate**
- **Reduce technical debt**
- **Improve usability**

Some optimization examples

- **Function to unpack parsed data**
- Wrapping values in objects
- Handling of check plugin returns

And many more...

Code Examples: Before...

```
def check_bla(item, params, parsed):
    item_data = parsed.get(item)
    if not item_data:
        return
    ...
```

... and after

```
def check_bla(item, params, parsed):
    item_data = parsed.get(item)
    if not item_data:
        return
    ...
```

```
@get_parsed_item_data
def check_bla(item, params, item_data):
    ...
```

Some examples of what are we doing

Goals

- **Reduce boilerplate**
- **Reduce technical debt**
- **Improve usability**

Some optimization examples

- Function to unpack parsed data
- **Wrapping values in objects**
- Handling of check plugin returns

And many more...

Code Examples: Before...

```
try:
    value = int(raw_value)
except ValueError:
    value = None

print '%s' % get_bytes_human_readable(value) # ‘3.00 MB'
```

... and after

```
try:
    value = int(raw_value)
except ValueError:
    value = None

print '%s' % get_bytes_human_readable(value) # ‘3.00 MB'
```

```
value = Bytes(raw_value)
print value # ‘3.00 MB'
```

Code Examples: Before...

```
disk_size_1 = int(raw_bytes)
disk_size_2 = int(raw_bits)

# … 100 lines later …

summary = disk_size_1 + disk_size_2
```

Two numbers?
Fine, lets continue...

... and after

```
disk_size_1 = int(raw_bytes)
disk_size_2 = int(raw_bits)

# ... 100 lines later ...

summary = disk_size_1 + disk_size_2
```

```
disk_size_1 = Bytes(raw_bytes)
disk_size_2 = Bits(raw_bits)

# ... 100 lines later ...

summary = disk_size_1 + disk_size_2
```

Bytes() + Bits()?
Ok, lets convert it first.

Code Examples: Before...

```
bytes = int(raw_bytes)
duration = int(duration)

summary = bytes + duration
```

Two numbers?
Fine, lets continue...

... and after

```
bytes = int(raw_bytes)
duration = int(duration)

summary = bytes + duration
```

```
bytes = Bytes(raw_bytes)
duration = Seconds(duration)

summary = bytes + duration
```

raises a ValueError()

Some examples of what are we doing

Goals

- **Reduce boilerplate**
- **Reduce technical debt**
- **Improve usability**

Some optimization examples

- Function to unpack parsed data
- Wrapping values in objects
- **Handling of check plugin returns**

And many more...

Code Examples: Before...

```
def check_bla(item, params, info):
    return 0, "bla", [("xxx", 0)]
```

... and after

```
def check_bla(item, params, info):
    return 0, "bla", [("xxx", 0)]
```

```
def check_bla(item, params, info):
    return CheckResult(
                        State.OK,
                        Output("bla"),
                        Metric(name="xxx", value=0)
            )
```

Agenda

1. WHY – THE RATIONALE TO BUILD A CHECK-API
2. WHAT AND HOW – THE PRINCIPLES BEHIND THE CHECK-API
3. **NEXT STEPS FOR THE CHECK-API**

tribe29

Next steps for the Check-API

Restructure Check Plugins

From check**mk** 1.7:

- ✔ API and plugins are being transformed into Python modules
- ✔ Existing plugins are being refactored
- ✔ Complete switch-over to Python 3

Any questions?

tribe29

Thank you!

tribe29 GmbH
Kellerstraße 29
81667 München
Deutschland

Web — tribe29.com
E-Mail — mail@tribe29.com