What’s new in Icinga 2

Věroš Kaplan
@verosk

http://inuits.eu/
http://veroskaplan.cz/
A lesson in Zulu: “Icinga”

by Amanda Mailer | Nov 3, 2010 | Misc | 0 comments

You say “Eee-Chingaa”, I say “Aye-Singa”... but how do the Zulus say it?

We consulted a couple Zulu speakers and even received a pronunciation guide from us all out. As she told us:

“The ‘c’ in Icinga is actually a Romanized representation of the Zulu click. The easiest way to reproduce the click is by putting your tongue behind your top teeth and pulling it down, kind of like the ‘q’ in ‘question’. The vowels on either side of the click are “eee” noises, like in ‘see’. There is no hard ‘g’ sound in Icinga – the ‘ng’ is part of ‘sing’. Finally, the ‘a’ is pronounced like the “a” in ‘father’.

History

May 2009 - fork from Nagios
https://www.icinga.org/2009/05/06/announcing-icinga/

Icinga 1 / Icinga Web

Icinga 2 - June 2014
https://www.icinga.org/2014/06/16/icinga-2-0-has-arrived/

IcingaWeb2 - beta November 2014

Icinga 2.4.0 - in a week
Icinga Web 2 (beta)
Icinga Web 2.0.0

October 2, 2015

Tactical Overview

15 Hosts DOWN

Services
- 40 CRITICAL
- 40 WARNING
- 10 OK

36 Hosts UP
10 Hosts PENDING

Services
- 5 WARNING
- 1 Acknowledged
- 40 UNKNOWN
- 175 OK
- 40 PENDING

Monitoring Features
- Flap Detection
  - 61 Hosts Disabled
  - 351 Services Disabled
- Notifications
- All Hosts Enabled
- All Services Enabled
- Event Handlers

Host and Service Checks
- Hosts
  - 51 Active
  - 10 Passive
- Services
  - 311 Active
  - 40 Passive

Overview
- Dashboard
- Problems
- Test Random

History
- Reporting
- Documentation
- System
- Configuration

icingaadmin
Logout
Vagrant playground

# completely redesigned at July of 2015
# # see https://github.com/Icinga/icinga-vagrant

# clone repository
git clone https://github.com/Icinga/icinga-vagrant

# choose your version (standalone one?)
cd icinga2x

# start the box
vagrant up
Vagrant playground

```ruby
--> icinga2: Notice: /Stage[main]/Icinga2/Icinga2::Feature[livestatus]/Exec[icinga2-feature-livestatus]/returns: executed successfully
--> icinga2: Notice: /Stage[main]/Icingaweb2/Icingaweb2::Module[monitoring]/File[/etc/icingaweb2/enabledModules/monitoring]/ensure: created
--> icinga2: Notice: /Stage[main]/Icinga2/Service[icinga2]/ensure: ensure changed 'stopped' to 'running'
--> icinga2: Notice: /Stage[main]/Apache::Mod::Dav_fs/File[dav_fs.conf]/ensure: defined content as '{md5}899a57534f3d84efa81887ec93c90c9b'
--> icinga2: Notice: /Stage[main]/Apache::Mod::Cgi/Apache::Mod[cgi]/File[cgi.load]/ensure: defined content as '{md5}ac20c5c5779b37ab06b480d6485a0881'
--> icinga2: Notice: /Stage[main]/Main/Apache::Vhost[vagrant-demo.icinga.org]/Concat[5-vagrant-demo.icinga.org.conf]/File[5-vagrant-demo.icinga.org.conf]/ensure: defined content as '{md5}d999d0fd63b6ba5f3eaf8b17b6848107d'
--> icinga2: Notice: /Stage[main]/Php::Mod_php5/File[/etc/httpd/conf.d/php.conf]/content: content changed '{md5}99e38fbe7fa8dd9066f79cfbe9aaa18d' to '{md5}9f8788e6315fc8b275862031eb64be64'
--> icinga2: Notice: /Stage[main]/Apache::Service/Service[httpd]/ensure: ensure changed 'stopped' to 'running'
--> icinga2: Notice: Finished catalog run in 333.33 seconds
--> icinga2: Running provisioner: shell...
  icinga2: Running: /tmp/vagrant-shell120151104-12258-us6dta.sh
--> icinga2: The Icinga 2 Vagrant VM has finished installing. See http://192.168.33.5 for more details.
--> icinga2: The Icinga 2 Vagrant VM has finished installing.
--> icinga2: Navigate to http://192.168.33.5
--> icinga2: and log into Icinga Web 2 using icingaadmin/icinga as credentials.
```
Modules & features

icinga2 feature list
icinga2 feature enable

IDOUtils, livestatus,
API, cluster,
graphite, GELF writer,
Yet another

New programming language

Objects, attributes, variable, types, function, operators, ....

New VIM syntax highlighting

https://github.com/Icinga/icinga2/tree/master/tools/syntax/vim
Comments

// this is one-line comment
/* this is
   multi-line
   comment */

# this is comment too
#include <other_file> is also comment
Include

include <itl>
    # Icinga template library

include "not_so_magic_file.conf"
include "lot_of_files/*.conf"

include_recursive "directory"
    // only *.conf
include_recursive "directory" "*.cfg"
Constants

# /etc/icinga2/constants.conf

const PluginDir = "/usr/lib64/nagios/plugins"
Time granularity

object Service “is_it_friday” {
    check_interval = 1h
    retry_interval = 300s
    ...
}

Templates

template Host "linux-server" {
    check_command = "ping4"
    max_check_attempts = 3
    icon_image = "vendors/debian.png"
}

object Host "logstash" {
    import "linux-server"
    address = "192.168.4.110"
    icon_image = "vendors/debian.png"
}
object Host "logstash" {
    import "linux-server"
    address = "192.168.4.110"
    icon_image = "vendors/debian.png"
}

object Service "syslog-port" {
    host_name = "logstash"
    check_command = "check_syslog"
}
Apply services ...

```bash
apply Service "syslog-port" {
    check_command = "check_syslog"
    assign where host.name == "logstash"
}

# or even better

apply Service "syslog-port" {
    check_command = "check_syslog"
    assign where match("logstash*", host.name)
}
```
... works for other objects too

object ServiceGroup "time" {
    display_name = "time tests"
    assign where (service.name == "time")
    assign where (service.name == "ntp_time")
}

object HostGroup “logstash servers” {
    display_name = “Logstash servers”
    assign where match(“logstash*”, host.name)
    ignore where host.vars.test_server == true

    ... works for other objects too
}
Variables

template Host "linux-server" {
    ...
    vars.host_type = "linux-server"
    vars.distribution = "RedHat"
}

object Host "logstash" {
    import "linux-server"
    ...
    vars.distribution = "Debian"
}
Apply by variables

apply Service “updates” {
    check_command = “check_yum”
    assign where ((host.vars.host_type == “linux-server”) &&
                  (host.vars.distribution == “RedHat”))
}

apply Service “updates” {
    check_command = “check_apt”
    assign where ((host.vars.host_type == “linux-server”) &&
                  (host.vars.distribution == “Debian”))
}
List variables

object Host "windows-server" {
  ...

  vars.disks = [ "C", "D", "E" ]
}

apply Service "disk-" for (disk in host.vars.disks) {
  ...
  ...
  check_command = "check_disk"

  vars.disk_name = disk
}
Dictionary variables

object Host "oracle-001" {
    ...

    vars.mounts["/oradata"] = {
        disk_warning = 60
    }
    vars.mounts["/archive"] = {
        disk_warning = 80
    }
}
object Host "oracle-001" {
    ...
    vars.mounts["/oradata"] = {
        disk_warning = 60
    }
}
apply Service for (mount_name => config in host.vars.mounts) {
    check_command = "disk_free"
    ...
    vars.disk_warning = 15
    vars.disk_critical = 10
    vars.mount_name = mount_name
    vars += config
}
object CheckCommand "disk_free" {
    command = [ PluginDir + "\check_disk\" ]
    args = {
        
    }
    env = {
        MOUNT_NAME = "$mount_name$"
    }
}
Exception occurred while checking 'stredisko-049-ostrava/uptime': /build/icinga2-pfgHLy/icinga2-2.1.1/doc/icinga/macroprocessor.cpp(170): Throw in function static icinga::String
icinga::MacroProcessor::InternalResolveMacros(const icinga::String&, const ResolverList&, const Ptr<, icinga::String*, const
EscapeCallback&>, int)
Dynamic exception type: boost::exception_detail::clone_impl::clone_exception_detail::error_info_injector<std::runtime_error >
std::exception::what: Closing $ not found in macro format string.
[icinga::StackTrace] =
(0) libbase.so: void boost::throw_exception<boost::exception_detail::error_info_injector<std::runtime_error >>(boost::exception_detail::error_info_injector<std::runtime_error const&>) (+0x4a) [0x2b892f8fe2a]
(1) libbase.so: void boost::throw_exception_(std::runtime_error const&)(std::runtime_error const& char const*, char const*, int) (+0x54) [0x2b892f8feda4]
(2) libicinga.so: icinga::MacroProcessor::InternalResolveMacros(icinga::String const&, std::vector<std::pair<icinga::String,
boost::shared_ptr<icinga::Object>>, std::allocator<std::pair<icinga::String, boost::shared_ptr<icinga::Object>>>>) const &
boost::shared_ptr<icinga::CheckResult> const&, icinga::String*, boost::function<icinga::String(icinga::String const&)> const&
<int> (+0x447) [0x2b893407aac7]
(3) libicinga.so: icinga::MacroProcessor::ResolveMacros(icinga::Value const&, std::vector<std::pair<icinga::String,
boost::shared_ptr<icinga::Object>>, std::allocator<std::pair<icinga::String, boost::shared_ptr<icinga::Object>>>>) const&
boost::shared_ptr<icinga::CheckResult> const&
boost::function<icinga::String(icinga::String const&)> const&
<int> (+0x1b77) [0x2b892fda68f7]
(4) libbase.so: icinga::DynamicObject::InvokeMethod(icinga::String const&, std::vector<icinga::Value, std::allocator<icinga::Value> >
const&) (+0x177) [0x2b8934640cd]
(5) libbase.so: icinga::ScriptFunction::Invoke(std::vector<icinga::Value, std::allocator<icinga::Value> > const&)
(std::vector<icinga::Value, std::allocator<icinga::Value> > const&
boost::function<icinga::String(icinga::String const&)> const&) (+0x1c1) [0x2b8934644c4]
(6) libbase.so: icinga::DynamicObject::InvokeMethod(icinga::String const&, std::vector<icinga::Value, std::allocator<icinga::Value> >
const&) (+0x177) [0x2b892fda68f7]
(7) libicinga.so: icinga::CheckCommand::Execute(boost::shared_ptr<icinga::Checkable> const&,
boost::shared_ptr<icinga::CheckResult> const&) (+0x16e) [0x2b89340f9ce]
(8) libicinga.so: icinga::Checkable::ExecuteCheck() (+0x1be) [0x2b89340a22ce]
(9) libchecker.so: icinga::CheckerComponent::ExecuteCheckerHelper(boost::shared_ptr<icinga::Checkable> const&)
(boost::shared_ptr<icinga::Checkable> const&
std::vector<icinga::Value> > const&) (+0x33) [0x2b89348d0433]
(10) libbase.so: icinga::ThreadPool::WorkerThread::ThreadProc(icinga::ThreadPool::Queue&) (+0x2d8) [0x2b892fdeb728]
(11) libboost_thread.so.1.49.0: <unknown function> (+0x10629) [0x2b892f71c629]
(12) libpthread.so.0: <unknown function> (+0x6b50) [0x2b892f90db50]
(13) libc.so.6: clone (+0x6d) [0x2b8931a4d7bd]
Debug commands?

object CheckCommand "disk_free" {
    command = [ "/bin/echo", PluginDir + "check_disk" ]
    args = {
        "-H" = "$address$"
        "-t" = "$mount_name$"
        "-w" = "$disk_warning$"
        "-c" = "$disk_critical$"
        ...
    }
    env = {
        MOUNT_NAME = "$mount_name$"
    }
}
define CheckCommand "ssh" { ... }
define NotificationCommand "hipchat" { ... }
define EventCommand "ipmi-restart" { ... }
Notifications

apply Notification "mail-icingaadmin" to Host {
  import "mail-host-notification"

  user_groups = host.vars.notification.mail.groups
  users = host.vars.notification.mail.users

  assign where host.vars.notification.mail
}
Dependencies

apply Dependency "router-ostrava" to Host {
    parent_host_name = "router-ostrava"
    assign where match("*-ostrava", host.name)

    disable_checks = true
    disableNotifications = true
}

apply Dependency "is-parent" to Host {
    parent_host_name = host.vars.parent
    assign where host.vars.parent
    ...
}

Best practices

- don’t write your configs by hand
- but if you need, use variables & apply
HA, Zones & API