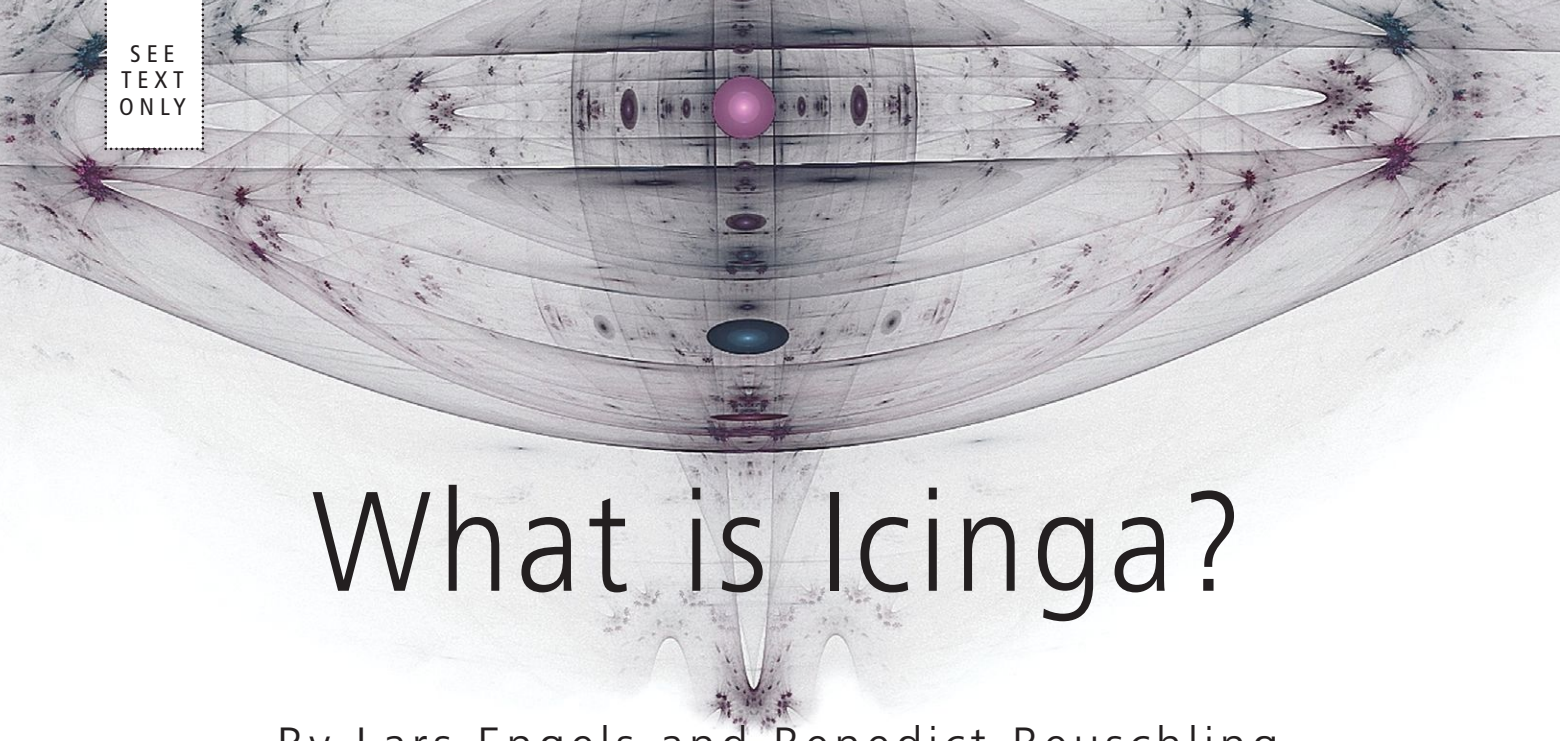

What is Icinga?

By Lars Engels and Benedict Reuschling

During the early ages of network monitoring, there was a monitoring software called Netsaint, which was later renamed to Nagios for legal reasons. While Nagios was very successful in the open-source world, patches from the community were often rejected or added very slowly. In 2009, members of the Nagios community created a fork that was named Icinga (Zulu for: it "examines"). Since the release of Icinga 2.0 in 2014, it is no longer a fork of Nagios, but a complete rewrite with new features like distributed monitoring, HA-Clustering, a REST API, and more. Icinga can reuse all Nagios monitoring plugins, while providing a modern web interface and powerful extensions to its core functionality.

Installation on FreeBSD/ZFS

In this article, we will walk through a setup of Icinga and Icinga Web 2 on FreeBSD to monitor hosts. The configuration data, events, and user authentication will be stored in a PostgreSQL database (MySQL is also available by default). NGINX will serve the web pages for Icinga Web 2. It is assumed that there is a FreeBSD base installation already installed and that it is connected to the network, able to download packages and reach other hosts via `ping` and `ssh`. This setup has been tested on a Raspberry Pi 3, as well as on a regular server without requiring any special hardware or tuning parameters.

The Icinga documentation (https://docs.icinga.com) has an excellent description of the necessary Icinga installation steps and even provides FreeBSD-specific instructions (when paths are different, for example). A FreeBSD port/package is also available and already includes some steps that users of other Unix distributions must still perform.

The easiest way to install Icinga 2 on FreeBSD is to use the package net-mgmt/icinga2. This guide uses # in front of commands that should be performed by the root user and `$` for a non-root user. Alternatively, `sudo` can be used to temporarily elevate privileges and to execute commands as another user.

Installing Icinga 2

We begin by installing a couple of packages required for the setup (see Box 1). Icinga 2 will install the basic monitoring components. Icinga Web 2 is an optional component to display events and check results in a browser-based dashboard. PostgreSQL 9.5 is the database server being used

Note

This guide concentrates on getting Icinga 2 up and running, and won't focus on performance, SSL setup, and other security options (other than passwords). This is left as an exercise for the reader, once you have become more familiar with Icinga 2 and its features.

by the package at the time of this writing. Although the setup of NGINX is described here, Apache 2 users can run Icinga Web 2 just as well.

The packages ImageMagick-nox11 and pecl-imagick are installed separately, as they are not pulled in as dependencies by default. When we start to configure Icinga Web 2, it will complain about these missing components when they are not installed. They are needed to create graphs in the PDF output when generating reports. Without them, the report PDFs will have stats and metrics in textual form only. Other dependencies like PHP, which is used by Icinga Web 2, will be pulled in automatically by the packages listed here.

Box 1
```
# pkg install icinga2 icingaweb2 postgresql95-server nginx ImageMagick-nox11 pecl-imagick
```

After all components have been installed, we add entries using `sysrc(8)` (see Box 2) to `/etc/rc.conf` so that these components will be automatically started upon reboot.

Box 2
```
# sysrc icinga2_enable=yes
# sysrc postgresql_enable=yes
```

Setting up PostgreSQL

Before we begin setting up the PostgreSQL database and create users for Icinga, we create a ZFS dataset to hold the data (UFS users can just use `mkdir` to follow along). Replace `monitor` with the name of your pool in the following examples:

Box 3
```
# zfs create -o mountpoint=/usr/local/pgsql/data monitor/pgdata
# zfs set compression=lz4 monitor/pgdata
# zfs set recordsize=8k monitor/pgdata
# zfs set logbias=throughput monitor/pgdata
# zfs set redundant_metadata=most monitor/pgdata
# zfs set primarycache=metadata monitor/pgdata
# chown pgsql:pgsql /usr/local/pgsql/data
```

The more systems are being monitored, the more writes to the database will be performed, so we tune ZFS accordingly. With these settings, the database transactions will not only be compressed on disk, but will also honor the metadata settings the database uses, so ZFS will not assume to know any better than PostgreSQL when it comes to writes to disk.

Now it is time to log in as the `pgsql` user and create the database cluster by running `initdb` with the `data` directory (dataset) that we just created:

Box 4
```
# su pgsql
$ cd
$ initdb -D ./data -E UTF8
$ pg_ctl start -D ./data
```

We create a new database user `icinga`, a database with the same name, and assign the `icinga` user as the owner. Provide a password for the icinga user when asked. It is required later when we set up Icinga Web 2, so make sure to not forget it.

Box 5
```
$ createuser -dPrs icinga
$ createdb -O icinga -E UTF8 icinga
```

The `pg_hba.conf` file was generated when `initdb` was executed and controls which user can access the database. To allow the icinga user to connect only locally to the database, edit the `pg_hba.conf` file in `data` and add the following lines:

Box 6
```
local      icinga      icinga                                  md5
host       icinga      icinga          127.0.0.1/32            md5
host       icinga      icinga          ::1/128                 md5
```

The icinga database needs a couple of tables, indices, and references before it can record the monitoring data from the hosts. This schema is provided by the port/package and located in `/usr/local/share/icinga2-ido-pgsql/schema/pgsql.sql`. The `psql` command interpreter is used to execute the SQL-script directly:

Box 7
```
$ psql -U icinga -d icinga < /usr/local/share/icinga2-ido-pgsql/schema/pgsql.sql
```

After the script was executed successfully, the database-specific configuration steps are done. Log out of the `pgsql` user before continuing. Icinga connects to the database via IDO (`Icinga Data Out to Database`) and Icinga needs to know about this. Icinga 2 has a plugin-like interface that can enable and disable features. For IDO to work with the PostgreSQL database, we need to enable the `ido-pgsql` feature. Another feature that is needed is called `command`. Afterwards, Icinga 2 is started for the first time.

Box 8
```
# icinga2 feature enable ido-pgsql
# icinga2 feature enable command
# service icinga2 start
```

Icinga is basically running now and will perform checks for the local machine. Having a nice dashboard to get an overview of any reports and any incidents is much better, so we configure Icinga Web 2 and the NGINX webserver next.

Configuring NGINX for Icinga Web 2

Icinga Web 2 is running on PHP, and we will use PHP FPM to make it work with NGINX. First, we enable `php-fpm` and `nginx` to start when the system is rebooted:

Box 9
```
# sysrc php_fpm_enable=yes
# sysrc nginx_enable=yes
```

Then, we use a couple of `sed` commands to configure the `php-fpm` configuration file. The first line lets it listen to the local domain socket and the three lines below it uncomment owner, group, and mode options to use the ones provided by FreeBSD (i.e., the `www` user and the default permissions for the socket).

Box 10
```
# sed -i '' "s/listen\ =\ 127.0.0.1:9000/listen\ =\ \/var\/run\/php5-
fpm.sock/" /usr/local/etc/php-fpm.conf
# sed -i '' "s/;listen.owner/listen.owner/" /usr/local/etc/php-fpm.conf
# sed -i '' "s/;listen.group/listen.group/" /usr/local/etc/php-fpm.conf
# sed -i '' "s/;listen.mode/listen.mode/" /usr/local/etc/php-fpm.conf
```

Icinga Web 2 provides an example file for `nginx.conf` in `/usr/local/share/examples/icingaweb2/nginx/icingaweb2.conf`. We take the relevant portions and put them before the `location / {` line in `/usr/local/etc/nginx/nginx.conf`:

Box 11
```
location ~ ^/icingaweb2/index\.php(.*)$ {
  # fastcgi_pass 127.0.0.1:9000;
  fastcgi_pass unix:/var/run/php5-fpm.sock;
  fastcgi_index index.php;
  include fastcgi_params;
  fastcgi_param SCRIPT_FILENAME /usr/local/www/icingaweb2/public/index.php;
  fastcgi_param ICINGAWEB_CONFIGDIR /usr/local/etc/icingaweb2;
  fastcgi_param REMOTE_USER $remote_user;
}
location ~ ^/icingaweb2(.+)? {
  alias /usr/local/www/icingaweb2/public;
  index index.php;
  try_files $1 $uri $uri/ /icingaweb2/index.php$is_args$args;
}
```

PHP needs to be configured as well. Fortunately, FreeBSD provides a configuration file for PHP with reasonable settings for production use, aptly named `php.ini-production`. We copy this file to the location of the `php.ini` file:

Box 12
```
# cp /usr/local/etc/php.ini-production /usr/local/etc/php.ini
```

One line still needs to be added to let PHP know which time zone it is in. Substitute the example below with the time zone in which your monitoring server is located.

Box 13
```
# echo "date.timezone = Europe/Berlin" >> /usr/local/etc/php.ini
```

Time to start both PHP-FPM and the NGINX webserver:

Box 14
```
# service nginx start
# service php-fpm start
```

If everything works, open a browser and point it to the following URL to begin the Icinga Web 2 configuration: http://local.domain.or.ip/icingaweb2/setup.

If something went wrong, retrace the steps above and have a look at the log files located in `/var/log/icinga2` for any clues about what went wrong. Log rotation examples for newsyslog can be found under `/usr/local/share/examples/icinga2/newsyslog`.

Now, let's configure the Icinga Web 2 dashboard.

Configuring Icinga Web 2

The first welcome screen asks for the setup token. This is done to prevent someone from accidentally stumbling onto the freshly installed Icinga server and misconfiguring it (remember this could be the evil Internet). To create the necessary setup token, run the following commands:

Box 15
```
# /usr/local/www/icingaweb2/bin/icingacli setup token create \
  --config=/usr/local/etc/icingaweb2
# chown -R www:www /usr/local/etc/icingaweb2
```

Copy the setup token that is being echoed on the screen and enter it in the setup token field. Click Next to continue. The next screen will show what kind of modules the installation has detected. Make sure that at least "Monitoring" is checked, and click Next. To make sure the Icinga installation runs with all the required components (mostly PHP modules), Icinga will provide an overview page of what it detected in the local installation. When following this guide, most of these fields should be green, meaning that the module is present and can be used. The "Linux Platform" field can be safely ignored, as the software runs just as well on FreeBSD. Click Next once more.

This screen asks how users will authenticate against Icinga Web. We'll use the PostgreSQL database we set up earlier, but LDAP works just as well for corporate environments. After selecting "database," we'll continue to the next page where we'll enter our connection information for it. Make sure to select "PostgreSQL" as the database type; localhost and the port should already be correct. Enter "`icinga`" as the database name. The user and password is the same we provided in the postgresql setup above. Enter `UTF-8` as the encoding. You can decide whether you want a persistent connection to the database, which is usually a good idea when using the web interface often. You can validate the connection to see whether everything works before continuing on to the next page.

This page asks which authentication backend to use, and should already provide a default entry. More authentication backends can be added later in Icinga Web 2, so just continue here. Enter the credentials (username and password) for the administrative user account on Icinga Web 2 in the next screen. Make sure to remember the password before continuing.

The following screen deals with debugging (whether stacktraces should be created, which are user-visible), where settings are stored, and how logging should be done (logging type and log level). Logging to a separate file is a good idea if syslog shouldn't be cluttered with Icinga 2 messages. Run `mkdir /var/log/icingaweb2` followed by `chown www:www /var/log/icingaweb2` when switching to

"`file`" so that the web interface is able to write its logs there.

The next page summarizes all your settings, and when you are satisfied with these, go to the next page to configure to monitoring component. The backend is IDO, which we already used for the PostgreSQL database, so we just continue on from here. We want to store all monitoring information in PostgreSQL, so we have to enter the same connection information we used earlier. Confirm that the settings are correct by validating them before hitting the Next button.

The command transport is important when multiple instances of Icinga are being used, but in our case, we accept the local file transport settings and continue on. No need to make changes to the defaults provided in the variables to be protected. Another summary page follows before we can finally finish the setup for Icinga Web 2 and log in for the first time. Use the administrative account created earlier in the setup to get to the monitoring dashboard.

Monitoring Hosts and Services

Monitoring is a complex topic and Icinga has many different ways to monitor a remote system. The Icinga 2 documentation has plenty of examples available.

The easiest way is via simple ping checks, which we'll configure for an example host.

Open `/usr/local/etc/icinga2/conf.d/hosts.conf` and add an entry like this (Box 16):

Box 16

```
object Host "Example" {
  import "generic-host"
  address = "IP-ADDRESS"
}
```

Replace "`Example`" and "`IP-ADDRESS`" with the hostname and IP address, respectively. After saving and exiting the file, let Icinga 2 check its config (`# service icinga2 configcheck`) and restart the icinga2 service. The Icinga Web 2 interface should now display a new pending host in the dashboard, and a short time later, the host should be regularly checked by pings. In case the host is down, Icinga 2 will display a warning in the dashboard and remove it once the host can be reached again.

Icinga Can Do Even More...

Icinga can be extended with more functionality beyond monitoring machines and their services. For example, in complex environments, where there are a lot of objects being monitored in a hierarchical fashion, a business process module can help visualize them. This way, many thousands of cloud machines that form the basis for a service can be viewed from a high-level dashboard. In case there is an alert, the module allows you to drill down to the exact machine that is triggering the alert. [https://github.com/Icinga/icingaweb2-module-businessprocess]

Icinga Director makes it easy to handle Icinga 2 configurations. By allowing users the flexibility to create their own objects by "point & click," while at the same time completely automating their datacenter, sysadmins can be sure their monitoring solution grows with higher demands for their infrastructure. [https://github.com/Icinga/icingaweb2-module-director]

Another module adds a generic trouble ticket system (TTS) functionality to Icinga 2. It allows the replacement of ticket patterns in Icinga Web 2 with links to a trouble ticket system (TTS). It defines a ticket hook that can be used by the core monitoring module and others for acknowledgements, downtimes, and comments. [https://github.com/Icinga/icingaweb2-module-genericts]

Lastly, system administrators dealing with systems distributed in many geographical locations can get a better overview with the map module for Icinga Web 2. It uses OpenStreetMap data to display host objects and their status on a map. Multiple hosts at the same location are clustered together. There is a custom host action to locate a specific host on the map. Clicking on a host marker will open a pop-up that will display that host services' current status. [https://github.com/nbuchwitz/icingaweb2-module-map]

Also, there's a module that integrates Grafana graphs into Icinga Web 2 so you can display your check's performance data in an attractive way. [https://github.com/Mikesch-mp/icingaweb2-module-grafana]

All of these additional modules are available as FreeBSD ports/packages. Make sure to study the Icinga 2 documentation to monitor your hosts and services in an effective way. •

LARS ENGELS has been a FreeBSD ports committer for 10 years and is also loosely involved in the Icinga project.

BENEDICT REUSCHLING joined the FreeBSD Project in 2009. After receiving his full documentation commit bit in 2010, he began mentoring others to become FreeBSD committers. He is a proctor for the BSD Certification Group and is currently serving as vice president for the FreeBSD Foundation.