

Zircon Software Product Suite

Version 3.1.0 for UNIX

Installation Guide

Zircon Computing LLC

Copyright © 2005–2010 Zircon Computing LLC. All rights reserved.

zNet is a registered trademark of Zircon Computing LLC.

Linux is a registered trademark of Linus Torvalds.

Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States and/or other countries Copyright © 2009 Microsoft Corporation. All rights reserved.

IBM is a registered trademark of International Business Machines Corporation Copyright © IBM Corporation 1994, 2009. All rights reserved.

Adobe, Acrobat, PDF, Portable Document Formats, and associated data structures and operators are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries Copyright © 2009 Adobe Systems Incorporated. All rights reserved.

All the other trademarks mentioned within this document are the property of their respective owners.

All trademarks are acknowledged.

Information in this document is subject to change without notice and is provided "as is" with no warranty. Zircon Computing LLC makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Zircon Computing LLC shall not be liable for errors contained herein or for any direct, indirect, special, incidental, or consequential damages in connection with the use of this material.

1. Getting Started

The Zircon Software Product Suite from Zircon Computing is an easy and affordable way to create high-performance distributed and parallel computing on demand. The following steps guide you through the installation of the Zircon software that contains all components (e.g., zEngines, zAdmin, zPluginBuilder, TCE, etc.) necessary to develop and run Zircon-enabled applications on UNIX. This software should be installed on *each* computer that will be involved in local or distributed processing, as well as on computers used to administer and monitor the processing. For more information on the Zircon software components, refer to the *Quick Start Guide* located in the /DOCS directory, which will appear after the installation is complete.

To install Zircon software on UNIX, your computer should be equipped with the following:

- Linux® or Solaris® 10 on x86_64 and/or x86_32
- GNU Make (3.80 or later)
- GCC (3.4.3 or later)
- A minimum of 75MB of available disk space. Additional disk space may be required, depending on the size of log files and plug-in libraries being generated.

2. Installing the Zircon Software

2.1 Unpack Installation Tar File

Use the following commands to unpack the Zircon software installation tar file and examine its contents.

```
[user@localhost ~]$ tar zxvf zproducts_Linux32.tar.gz
[user@localhost ~]$ tar zxvf zprojects_Linux32.tar.gz
[user@localhost ~]$ cd Zircon
[user@localhost Zircon]$ ls . znet/ zfunction/
.:
configs  setup.sh  envselect.sh  zfunction  znet  doc

zfunction/:
bin  common_make_include.gnu  include  lib

znet/:
bin  include  lib  Makefile.src  Rules.mk

[user@localhost Zircon]$ cd ../ZirconProjects
[user@localhost ZirconProjects]$ ls . znet/ zfunction/
.:
doc  examples.sh  license.txt  log  Makefile  zfunction  znet

zfunction/:
bin          cpp_examples  include  Makefile
c_examples  examples.sln  lib      README.txt

znet/:
advanced_STL  examples.sln  lib      README.txt
bin           include      Makefile  regular
```

As shown above, the directories contain the following files:

- `setup.sh`
Run this bash shell script after initial installation to create a config directory with an appropriate TCE server name or IP address and the location of the `TCE_STORAGE` directory. This script can also be used to create multiple config directories (as might be the case for developers).
- `envselect.sh`
This bash shell script accomplishes several tasks. First, it establishes the necessary environment variables. Second, it allows users to select one of potentially several different runtime configurations (only one is generated during install). Third, it installs a hidden shell script in the user's home directory that can be automatically sourced by `examples.sh` in `ZirconProjects` or a user shell script. Fourth, it uses the standard UNIX command `ps` to inform users of already running instances of key Zircon software executables, which is useful in scenarios where developers may want to start fresh and ensure there are no conflicting instances running.

Configured Environment Variables

- `PATH` – `znet/bin` and `zfunction/bin` added
 - `LD_LIBRARY_PATH` – `znet/lib` and `zfunction/lib` added
 - `ZNET_ROOT` – Points to root of the zNet installation
 - `ZFUNCTION_ROOT` – Points to root of zFunction installation
 - `ZCONFIGDIR` – Points to current configuration directory (use `setup.sh` to generate one or more configurations), as selected via `envselect.sh`. You may have multiple configuration settings organized in various directories and switch between them using this environment variable
 - `ZLOGDIR` – Points to desired location of various log files (default is the current directory at the time the executable is run)
- `zfunction/common_make_include.gnu`
A top level makefile used by zFunction's zPluginBuilder to make zPluginLibraries
 - `znet/Makefile.src`
A top level makefile used by zNet and zFunction examples
 - `znet/Rules.mk`
An additional makefile used by the zNet and zFunction examples.
 - `docs`
Subdirectory containing documentation
 - Subdirectories containing configuration files
 - `configs/localhost` Default configuration to run TCE and all binaries on local host
 - Subdirectories containing the zNet product files
 - `znet/bin` Binaries and configuration for zNet
 - `znet/include` Contains ACE and zNet header files specifying the zNet API
 - `znet/lib` The zNet runtime libraries
 - Subdirectories containing the zFunction product files
 - `zfunction/bin` Binaries and configuration for zFunction
 - `zfunction/include` Contains `Z_api.h` specifying the zFunction C API
 - `zfunction/lib` The zFunction runtime libraries

2.2 Linux Shell Environment

The shell scripts provided in the installation package are designed to take advantage of features in the BASH shell, which is typically the default shell for Linux installations. The following is a typical shell session immediately after unpacking the tar files:

```
[user@localhost ~]$ cd Zircon
[user@localhost ~]$ ./setup.sh
Enter hostname/IP of server running TCE [localhost]:
localhost
Enter writable location for TCE storage directory [TCE_STORAGE]:
/home/user/TCE_STORAGE
TCE_HOSTNAME=localhost
TCE_STORAGE_DIR=/home/user/TCE_STORAGE
Setup complete.
[user@localhost Zircon]$ source envselect.sh localhost
...
ZCONFIGDIR=/home/user/Zircon/configs/localhost
ZLOGDIR=/home/user/ZNET_LOGS
...
[user@localhost Zircon]$ cd ../ZirconProjects
[user@localhost ZirconProjects]$ source examples.sh
... [No output]
```

If the situation arises where BASH is not the currently running shell, there are several options.

- Change the default shell of the current user account to BASH. See your Linux administrator or a Linux administration guide for information on how to do this.
- Switch to BASH while working with a Zircon software installation, e.g., by simply typing `bash` at the shell prompt. A typical session might look like the following:

```
[user@localhost Zircon]$ bash
[user@localhost Zircon]$ source envselect.sh
[user@localhost Zircon]$ cd ../ZirconProjects
[user@localhost ZirconProjects]$ source examples.sh
...
[user@localhost ZirconProjects]$ tce &
```

- Switch to BASH long enough to configure your environment, and then switch back to your preferred shell. For example, assuming your preferred shell is TCSH you can perform the following steps

```
[user@localhost Zircon]$ bash
[user@localhost Zircon]$ source envselect.sh
[user@localhost Zircon]$ cd ../ZirconProjects
[user@localhost ZirconProjects]$ source examples.sh
...
[user@localhost ZirconProjects]$ tcsh
[user@localhost ZirconProjects]$ tce &
...
```

2.3 Configure and Start License Server

The Configuration Environment (TCE) executable in `$ZNET_ROOT/bin/tce` is the license and configuration server for Zircon software products. The TCE daemon validates applications during startup and is provided for every operating system supported by Zircon software. This daemon manages a Zircon Software domain, which contains all compute servers, clients, and monitoring tools that share a network and are configured to communicate with each other. Multiple domains can be set up on the same network, but each domain must run a separate TCE daemon that maintains certain domain-global configuration settings (e.g., network multicast addresses) and authenticates all zNet applications in the domain. A TCE daemon is only required to launch applications in the domain. Any subsequent failure of a TCE daemon does not affect operation of applications that are already running.

It uses the `$ZCONFIGDIR/tce_common.cfg` configuration file to specify the address of the computer it is running on. All Zircon software services must use the same `tce_common.cfg` to access the license server correctly. The default setting points to the local host. If you plan to launch zEngines on the network, you will have to change the `TCE:host` resource in `tce_common.cfg` to the IP address of the computer running TCE. You can find the IP address of your computer by executing:

```
[user@localhost ~]$ /sbin/ifconfig -a
```

On Linux the primary interface is usually designated as `eth0` and on Solaris as `bnx0`. Please see your *OS Administrator Guide* for additional help in determining your IP address. For instance, if the primary network interface of the computer where TCE is installed returns 192.168.1.101, change the settings in `tce_common.cfg` to:

```
TCE:host          192.168.1.101
```

Systems that either reside on multiple networks or have more than one active network interface may run into communication problems between zNet nodes. These problems are typically caused by communication occurring on the wrong network interface. To ensure the correct interface is used, add the following line to your `tce_common.cfg` file, substituting the actual network number(s) where appropriate:

```
Z:IO:listen_ips  192.168.10.*;192.168.11.*
```

By default, each Zircon software executable looks for the config file located in `$ZCONFIGDIR` (and if this variable is not defined, then in `$PWD`) with the same name as the executable and with the extension `.cfg`. Zircon software executables produce log files. The location of the log files is determined as follows (first takes higher precedence):

- `ZLOGDIR` – If defined, logs are placed in `$ZLOGDIR/LOGS`
- `PWD` – Logs are placed in the current working directory at the time the executable is run.
- The `envselect.sh` script defaults `ZLOGDIR` to `~/ZNET_LOGS`

The settings for TCE are defined in `tce.cfg`, which includes two additional configurations files: `tce_common.cfg` and `license.cfg`. The second file, `license.cfg`, is only needed when running TCE and must be installed in `$ZCONFIGDIR` by the user. Here is a typical TCE run:

```
[user@localhost ZirconProjects]$ tce &
Application <tce> log directory: /home/user/ZNET_LOGS
Application <tce> config directory:
/home/user/Zircon/configs/localhost, config file: tce.cfg
```

2.4 Start zEngine(s)

The zEngine executable in `$ZNET_ROOT/bin/zengine` must be started on every computer intended for distributed computation. One way to start this executable is to install Zircon software in a shared (NFS mounted) directory. The zEngine executable is lightweight and does not consume any significant resources when it is not engaged in calculations. It need not be adjusted or changed in any way since zEngines can dynamically load any zPluginLibrary produced by Zircon software. We recommend launching one zEngine per core on each machine. Here is a typical zEngine run:

```
[user@localhost ZirconProjects]$ zengine &
Application <zengine> log directory: /home/user/ZNET_LOGS
Application <zengine> config directory:
/home/user/Zircon/configs/localhost, config file: zengine.cfg
```

2.5 Start zPluginBuilder

The zPluginBuilder executable in `$ZFUNCTION_ROOT/bin/zpluginbuilder` must be launched at least once on each platform that runs Zircon software. The zPluginBuilder is a specialized version of zEngine that can produce a zPluginLibrary given an XML description of the target function. The Zircon software installation contains zPluginBuilder executables for all supported platforms. On Linux and Solaris, the zPluginBuilder requires `gmake` and `gcc`. Here is a typical zPluginBuilder run:

```
[user@localhost ZirconProjects]$ zpluginbuilder &
Application <zpluginbuilder> log directory: /home/user/ZNET_LOGS
Application <zpluginbuilder> config directory:
/home/user/Zircon/configs/localhost, config file: zpluginbuilder.cfg
```

2.6 Start zAdmin

3. Launch the zAdmin executable in `$ZNET_ROOT/bin/zadmin` and verify the license by typing the letter 'I' and checking the number of available licenses and running services. Here is a typical zAdmin run:

```
[user@localhost ZirconProjects]$ zadmin
Application <zadmin> log directory: /home/user/ZNET_LOGS
Application <zadmin> config directory:
/home/user/Zircon/configs/localhost, config file: zadmin.cfg
```

Enter an action:

```
B      Build a plugin library
D      Deploy a plugin library
M      Manage servers, clients, and deployed libraries
I      Obtain licensing information for Zircon features
R      Reload license.cfg
V      Validate an XML Interface Description File
T      Show application event trace
X      Exit
```

```
I
EVAL:ZMON v3.1.0 e3 20100509_235900 1[12]
EVAL:ZNET v3.1.0 e3 20100509_235900 2[256]
```

Enter an action:

```
B      Build a plugin library
D      Deploy a plugin library
M      Manage servers, clients, and deployed libraries
I      Obtain licensing information for Zircon features
```

```
R      Reload license.cfg
V      Validate an XML Inteface Description File
T      Show application event trace
X      Exit
```

M

Actions:

```
A      get_servers
B      get_pools
C      get_clients
D      get_libs
E      get_deployments
F      get_plugins
G      get_connections
H      start_server
I      suspend_server
J      create_pool
K      remove_pool
L      move_server
X      exit
```

A

Server Details #1

=====

```
Server:      102
Name:        engine_rh5x64-u39.zlab.local_1
Pool:        public
Host:        rh5x64-u39.zlab.local
Platform:    Linux
Status:      active
Load:        0
Clients:     0
Libs:        0
```

Server Details #2

=====

```
Server:      103
Name:        zpluginbuilder_rh5x64-u39.zlab.local_1
Pool:        system
Host:        rh5x64-u39.zlab.local
Platform:    Linux
Status:      active
Load:        0
Clients:     0
Libs:        1
```

Actions:

```
A      get_servers
B      get_pools
C      get_clients
D      get_libs
E      get_deployments
F      get_plugins
G      get_connections
```

```

H      start_server
I      suspend_server
J      create_pool
K      remove_pool
L      move_server
X      exit

```

B

Pool	Servers	Suspended	Total Load	Load	Clients	Libs
====	=====	=====	=====	=====	=====	=====
public	1	0	0	0	0	0
system	1	0	0	0	1	0

Actions:

```

A      get_servers
B      get_pools
C      get_clients
D      get_libs
E      get_deployments
F      get_plugins
G      get_connections
H      start_server
I      suspend_server
J      create_pool
K      remove_pool
L      move_server
X      exit

```

C

Client Details #1

```

=====
Client:          104
Name:            zadmin_rh5x64-u39.zlab.local_1
Pool:            system
Host:            rh5x64-u39.zlab.local
Platform:        Linux
Number of Libs:  0
Number of Servers: 0
Library Names:
Number of Requests: 0
Number of Replies: 0

```

Actions:

```

A      get_servers
B      get_pools
C      get_clients
D      get_libs
E      get_deployments
F      get_plugins
G      get_connections
H      start_server
I      suspend_server
J      create_pool
K      remove_pool
L      move_server

```

4. Next Steps

You are now ready to try the examples and tutorials supplied by Zircon Computing and start building your own applications to harness the power of distributed and parallel processing on UNIX platforms. Please refer to the Zircon software *Quick Start Guide* and *Developer's Guide* for further information.