

# hostinfo egg

---

Look up host, protocol, and service information  
Extension for Chicken Scheme  
Version 1.0

Zbigniew

---

# Table of Contents

<b>1</b>	<b>About this egg .....</b>	<b>1</b>
1.1	Version history .....	1
1.2	Requirements .....	1
1.3	Usage .....	1
<b>2</b>	<b>Documentation .....</b>	<b>2</b>
2.1	Short and sweet .....	2
2.2	Records .....	3
2.3	Record lookup .....	3
2.4	One-stop shops .....	4
2.5	Utility functions .....	4
<b>3</b>	<b>Bugs .....</b>	<b>5</b>
<b>4</b>	<b>Examples .....</b>	<b>6</b>
<b>5</b>	<b>License .....</b>	<b>7</b>
	<b>Index .....</b>	<b>8</b>

# 1 About this egg

## 1.1 Version history

1.0          Initial release

## 1.2 Requirements

This egg requires the following extensions:

`vector-lib`

## 1.3 Usage

Load this egg like so:

`(require-extension hostinfo)`

## 2 Documentation

This extension performs host, protocol and service information lookups via underlying calls to *gethostbyname(3)*, *getprotobyname(3)*, and *getservbyname(3)*. Depending on your system, this may consult DNS, NIS, */etc/hosts*, */etc/services*, */etc/protocols*, and so on.

A simple interface is provided for the most common queries. Also provided is a more comprehensive interface using records, which contain all data available in a lookup.

IP addresses are represented by 4 (IPv4) or 16 (IPv6) byte **u8vector**s. The interface requires, and returns, addresses in this format; functions are provided to convert between the string and **u8vector** representations. However, the "do what I want" procedures (e.g. **host-information**) will do the conversion for you.

### 2.1 Short and sweet

Quickly perform the most common lookups. Convenient and efficient for one-off use, but perform a new lookup each time. They return **#f** on failure.

```
hostname->ip [procedure]
  (hostname->ip HOSTNAME)
```

Look up string **HOSTNAME** and return IP address as **u8vector**.

```
ip->hostname [procedure]
  (ip->hostname IPADDR)
```

Look up **u8vector** **IPADDR** and return hostname as string.

```
protocol-name->number [procedure]
  (protocol-name->number PROTOCOL-NAME)
```

Look up string **PROTOCOL-NAME** and return protocol number.

```
protocol-number->name [procedure]
  (protocol-number->name PROTOCOL-NUMBER)
```

Look up **PROTOCOL-NUMBER** and return protocol name as string.

```
service-port->name [procedure]
  (service-port->name SERVICE-PORT [PROTO])
```

Look up **SERVICE-PORT** number and return service name as string. Optional **PROTO** argument, which must be a string, constrains lookup to that protocol.

```
service-name->port [procedure]
  (service-name->port SERVICE-NAME [PROTO])
```

Look up string **SERVICE-NAME** and return the canonical port for that service. Optional **PROTO** argument as above.

## 2.2 Records

Some lookups return a host, protocol, or service record. These records print nicely on the screen, for convenient interactive use.

**hostinfo-address** [procedure]

(hostinfo-address h)

Retrieves the **address** field of the **hostinfo** record **h**. Accessors are similar for other records and their fields.

**hostinfo** [record]

**name** Hostname

**addresses**

A vector of one or more u8vector IP addresses

**aliases** A vector of any alternate hostnames

**address** The first IP address (u8vector) in addresses

**type** 'AF\_INET (IPv4) or 'AF\_INET6 (IPv6)

**length** IP address length in bytes

**protoinfo** [record]

**name** Protocol name

**number** Protocol number

**aliases** Vector of alternate names for this protocol

**servinfo** [record]

**name** Service name

**number** Service number

**aliases** Vector of alternate names for this service

**protocol** Name of protocol this service uses

## 2.3 Record lookup

**hostname->hostinfo** [procedure]

**ip->hostinfo** [procedure]

**service-name->servinfo** [procedure]

**service-port->servinfo** [procedure]

**protocol-name->protoinfo** [procedure]

**protocol-number->protoinfo** [procedure]

(hostname->hostinfo NAME)

(ip->hostinfo IPADDR)

(service-name->servinfo NAME)

(service-port->servinfo NUM)

(protocol-name->protoinfo NAME)

(protocol-number->protoinfo NUM)

These lookups correspond to those described in *Short and sweet*, but return a full record. The entire record is filled in a single system call.

## 2.4 One-stop shops

These decipher your argument, call the appropriate lookup, and return an information record.

**host-information** [procedure]  
(host-information HOST)

Look up and return a `hostinfo` record, or `#f`. `HOST` is a string hostname, a string numeric IP address, or a `u8vector` IP address.

**protocol-information** [procedure]  
(protocol-information PROTO)

Look up and return a `protoinfo` record, or `#f`. `PROTO` is a protocol number or string name.

**service-information** [procedure]  
(service-information SERVICE [PROTO])

Look up and return a `servinfo` record, or `#f`. `SERVICE` is a service number or string name. `PROTO` is an optional protocol number or string name, which will constrain lookups to that particular protocol.

**NOTE:** if the protocol number is illegal, an error is thrown, since this was probably unintentional.

## 2.5 Utility functions

**string->ip** [procedure]  
(string->ip IP-STRING)

Convert an IPv4 or IPv6 address string in canonical format to a `u8vector`, which can be considered an "IP address object". Returns `#f` on failure.

**ip->string** [procedure]  
(ip->string IPADDR)

Convert a 4 (IPv4) or 16 (IPv6) element `u8vector` to a string in canonical format. Throws an error if the `u8vector` is not 4 or 16 bytes long. This call should only fail on system error, in which case it will return `#f` (perhaps not the best behaviour).

### 3 Bugs

IPv6 lookup is not yet supported. However, IPv6<->string conversion works fine.

System errors return failure (**#f**) and so are indistinguishable from failed lookups. They should probably signal an error or an exception.

## 4 Examples

```
(host-information "www.call-with-current-continuation.org")
(host-information '#u8(194 97 107 133))
(host-information "194.97.107.133")
  ; => #,(hostinfo name: "www003.lifemedien.de"
  ;      addresses: #(#u8(194 97 107 133))
  ;      aliases: #("www.call-with-current-continuation.org"))
(ip->hostname '#u8(194 97 107 133)) ; "www003.lifemedien.de"
(string->ip "0708:0901")           ; #u8(7 8 0 0 0 0 0 0 0 0 0 0 0 9 1)
(ip->string '#u8(127 0 0 1))        ; "127.0.0.1"
(hostinfo-aliases
  (hostname->hostinfo
    (ip->hostname
      (hostname->ip
        (hostinfo-name
          (host-information "www.call-with-current-continuation.org"))))))
  ; => #("www.call-with-current-continuation.org")

(protocol-information 17)
  ; => #,(protoinfo name: "udp" number: 17 aliases: #("UDP"))
(protoinfo-name (protocol-information 2)) ; => "igmp"
(protoinfo-aliases (protocol-name->protoinfo
  (protocol-number->name
    (protoinfo-number
      (protocol-information "ospf"))))) ; => #("OSPFIGP")
(protocol-name->number "OSPFIGP")      ; 89 (you can look up aliases, too)

(servinfo-protocol (service-name->servinfo
  (service-port->name
    (servinfo-port (service-information "ssh")))))
  ; => "udp" (yes, really)
(service-information "ssh" "tcp")
  ; => #,(servinfo name: "ssh" port: 22 aliases: #() protocol: "tcp")
(service-information "ssh" "tco") ; => #f
(service-information 512 "tcp")
  ; #,(servinfo name: "exec" port: 512 aliases: #() protocol: "tcp")
(service-information 512 "udp")
  ; #,(servinfo name: "comsat" port: 512 aliases: #("biff") protocol: "udp")
(service-information 512 17)      ; same as previous
(service-information 512 170000)
  ; Error: (service-information) illegal protocol number: 170000
```



## 5 License

Copyright (c) 2005, 2006 Jim "Zb" Ursetto. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of the author nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# Index

## H

host-information .....	4
hostinfo .....	3
hostinfo-address .....	3
hostname->hostinfo .....	3
hostname->ip .....	2

## I

ip->hostinfo .....	3
ip->hostname .....	2
ip->string .....	4

## P

protocol-information .....	4
----------------------------	---

protocol-name->number .....	2
protocol-name->protoinfo .....	3
protocol-number->name .....	2
protocol-number->protoinfo .....	3
protoinfo .....	3

## S

service-information .....	4
service-name->port .....	2
service-name->servinfo .....	3
service-port->name .....	2
service-port->servinfo .....	3
servinfo .....	3
string->ip .....	4