# Introduction to Socket Programming

**By Juan Natera**
**OC Perl Mongers - Jan 2014**

# Outline

- What are Sockets?
- Brief History of Sockets.
- Basic Concepts.
- Examples.
- Demo.
- References.

# What are Sockets?

IPC mechanism developed to allow data exchange between applications, running either on the same host or on different hosts connected to a network.

# Brief History of Sockets

- Follows other IPC mechanisms, such as Pipes, FIFOs, etc.
- API Introduced with 4.2BSD in 1983, current interface based on 4.4BSD
- Soon ported to most UNIX implementations.
- Now present in most other Operating Systems.
- Basic API shared by most OSes, though many have introduced variations (affecting portability).
- Formally specified in SUSv3.

# ATTENTION!

There are many great resources that discuss these topics thoroughly, from books to the man pages in your favorite UNIX flavor.

This talk focuses on the simpler, HIGH level API for Perl socket programming, but keep in mind the LOW level API, which mirror what's available to other languages such as C/C++, etc., it is also available. we will see a bit of that.

All code using "use warnings; use strict;"

# Basic Concepts

## What are Sockets?

Sockets are objects that define a communication endpoint. Just like file descriptors are used to access files, socket descriptors are used to access sockets.

In fact, you can use some of the same operations, like read, syswrite, close, etc.

```
use Socket qw(AF_INET SOCK_STREAM pack_sockaddr_in
inet_aton);

socket(my $socket, AF_INET, SOCK_STREAM, 0) or die $!;
```

# Basic Concepts

## Socket Domains

| Domain | Description |
|---|---|
| `AF_INET` | IPv4 internet domain |
| `AF_INET6` | IPv6 internet domain |
| `AF_UNIX` | UNIX Domain |
| `AF_UNSPEC` | Unspecified |

# Basic Concepts

## Socket Types

| Socket Types | Description |
|---|---|
| `SOCK_DGRAM` | Fixed-length messages, connectionless, unreliable, (UDP) |
| `SOCK_STREAM` | Sequenced, reliable, bidirectional, connection oriented streams, (TCP) |
| `SOCK_SEQPACKET` | Fixed length, sequenced, reliable, connection oriented streams (SCTP) |
| `SOCK_RAW` | bypasses UDP / TCP, application needs to manage headers, etc. |

# Basic Concepts

## Sample low level API program

```
use Socket qw(:DEFAULT :crlf);
use IO::Handle;

socket(my $socket, PF_INET, SOCK_STREAM, 0)
    or die "socket: $!”;

my $port = getservbyname "http", "tcp";

die “invalid hostname $ARGV[0]” unless ($ARGV[0] =~ /$\w+\.\w+$/);
connect($socket, pack_sockaddr_in($port, inet_aton($ARGV[0]))) or die $!;

$socket->autoflush(1);
$socket->print("GET / HTTP/1.1$CRLF”);
$socket->print(“Connection: close$CRLF”);
$socket->print("Host: $ARGV[0]$CRLF$CRLF”);
print $socket->getlines();

$socket->close() or die $!;
```

# Basic Concepts

## Some basic functions

- `socket($socket, $domain, $type, $protocol)`
- `getsockopt($socket, $level, $opt_name)`
- `getsockopt($socket, $level, $opt_name, $opt_value)`
- `connect($socket, $dst_addr)`
- `close($socket)`
- `shutdown($socket, $how)`
- `bind($socket, $my_addr)`
- `listen($socket, $max_queue)`
- `accept($session_socket, $socket)`
- Addressing functions, IO functions, etc

# Examples

## TCP Client

# Examples

## TCP Server

# DEMO TIME

# References


- Network Programming with Perl by Lincoln Stein
- Perl Cookbook 2e by Tom Christiansen and Nathan Torkington
- Advanced Programming in the UNIX Environment 3e by Richard Stevens and Stephen A. Rago.
- The Linux Programming Interface by Michael Kerrisk
- Socket Image: http://flickr.com/photos/9822107@N08/2778963222/