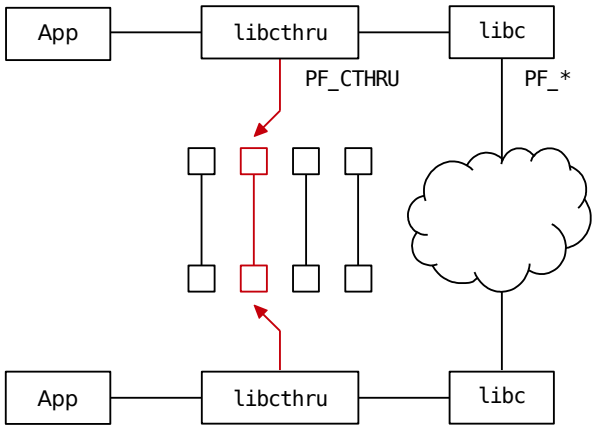
C-THRU™ Product Datasheet

$Rev: 3$

A communications software library, currently supporting POSIX systems; portable to other systems as required. Written in ISO standard C. Customer-facing interfaces are:

Interface

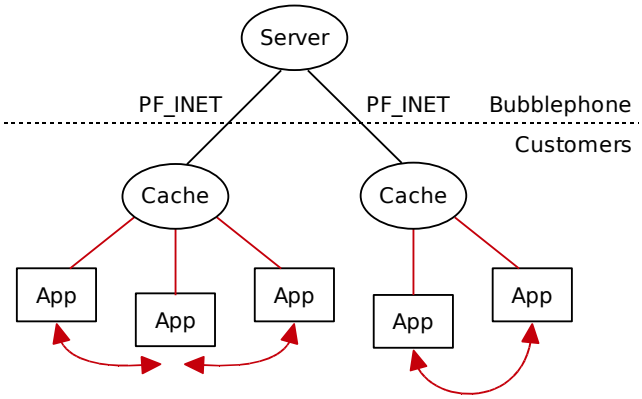
An *Application* links `libcthru` before `libc` to extend the OS's socket API to provide a new protocol family, PF_CTHRU. It abstracts multiple *Connection Methods* giving underlying transports as modular back-ends:



- Provides an end-to-end socket connection.
- Berkeley Socket API; can add other front-ends.
- Provides `SOCK_STREAM` and `SOCK_DGRAM`.
- Optional `AF_CTHRU` customer-designated addresses for ease of forming overlay networks.
- Transparently switches Method when necessary, whilst keeping the Application's socket open.
- Maintains a reserve pool of pre-connected Methods for fail-over throughout the lifetime of a socket.

Infrastructure Topology

Instances of `libcthru` are orchestrated by customer-hosted *Caches* (one per customer); these cache new Methods to update `libcthru`, and log usage statistics to help improve design of new Methods.

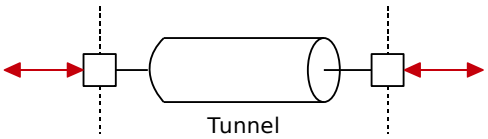


- Customer traffic is sent peer-to-peer for efficiency. This helps balance the load for bandwidth.
- The Server updates `libcthru` with new Methods as and when they become available. Method updates provide new connection techniques.
- The connection to the Cache is also `PF_CTHRU`.
- Caches and servers may be clusters of machines, rather than single nodes, for load balancing.

Internal to the library are the set of connection Methods, and the algorithm to select which are used:

Connection Methods

Methods abstract arbitrary transports providing a uniform API for I/O. These can be produced bespoke as per customer requirements. Methods may tunnel over other protocols of any network layer:

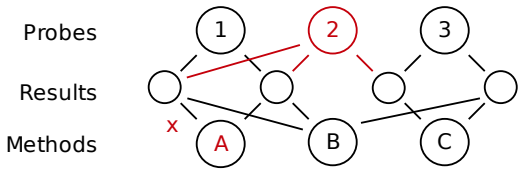


Examples:

- Simplest case is a direct IP connection.
- Tunnelling over HTTP to circumvent firewalls. Or more extreme, e.g. a tunnel over DNS.
- NAT traversal.
- Stenographic tunnels, and balancing traffic over multiple Methods for statistical frequency profiles.
- Non-IP transports, e.g. TETRA.
- Methods may or may not provide encryption.

Method Selection

Methods are selected by launching *Probes* to query the network to efficiently find which is most suitable. This selection is ongoing, and adjusts to fit the changing conditions of the network. Probes yield *Results;* Methods depend on sets of Results, forming a dependency graph:



Probes measure any quantifiable resource, such as latency, bandwidth, the ability to send certain packet types, or connect to ports. Probes are low overhead, as samples can often be taken from existing traffic.

Probe launching is minimal. E.g. if Probe 1's result for Method A (indicated “x”) makes Method A non-viable, `libcthru` can avoid needing to launch Probe 2.

Bubblephone Limited
Sussex Innovation Centre
Science Park Square
University of Sussex
Brighton BN1 9SB
Tel +44 (0) 1273 704 535
http://www.bubblephone.com/

Constraints and Requirements

- `libcthru` is threadsafe; the Application needn't be.
- Hosted ISO C90 or C99 environment. APIs are C90.
- Dynamic linking (ELF or Mach)
- Requires `pthreads`. Portable to alternative threading back-ends as required.
- `libcthru` maintains a single file, storing state, logs and other internal data.
- Currently unimplemented pending use-cases: `sendmsg(SCM_RIGHTS); AF_INET on PF_CTHRU; fcntl(F_GETFD, F_GETFL); fork()`.

Application Notes

C-THRU's functional features are broadly:

Making a connection in situations where it's difficult to connect
Keeping a connection open when roaming between networks
Maintaining quality as situations change

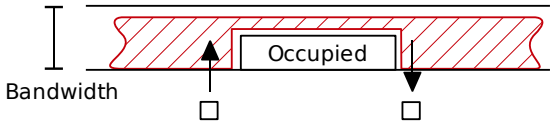
For quality maintenance, Results gathered during Method Selection may be fed back to Applications by passing a callback function through the `setsockopt()` interface; this function is called asynchronously when new Results are available. Applications vary widely in their use of this. Two examples from real use-cases:

VoIP or IPTV

A typical VoIP or IPTV Application streams media over a `SOCK_DGRAM` connection, permitting packet loss as the traffic is time-sensitive; resending lost packets would cause delays in reconstructing the media stream.

For situations with limited bandwidth (such as a residential ADSL connection), the capacity available for use may vary as unrelated traffic makes use of the same connection.

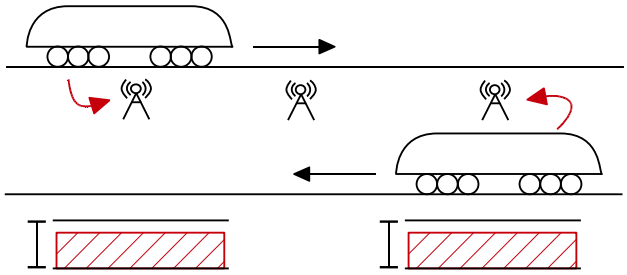
It is desirable to make the best use of the available bandwidth for the best quality stream. The bandwidth determines which codecs may be used for encoding the audio or video media. If more bandwidth is available, a higher quality codec may be used. Some codecs provide for variable compression.



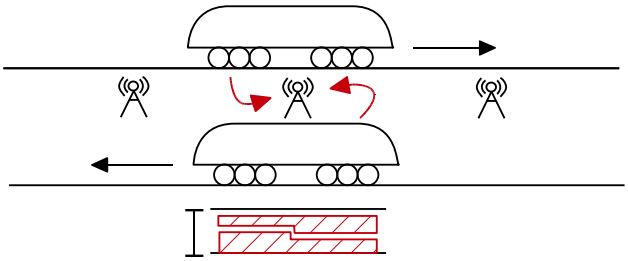
Using C-THRU, these Applications may either scale the codec as the occupied bandwidth changes, or temporarily select an alternate codec; the Application supplied callback is called once when the Result's value become out of bounds, and again when its value returns within range.

Mobile Networking for Vehicles

Automated vehicles are often controlled from a central location. These may communicate over intermittent connections made unreliable from physically moving between different wireless base stations, or even between access points of different network types (often supplied for fail-over should one network malfunction).



These wireless networks may be subject to channel saturation (malicious or otherwise) from third parties, or simply from two vehicles passing through the same area.



Probe feedback for data prioritisation allows these vehicles to prioritise critical traffic (such as control data and logs) whilst reducing non-essential traffic such as CCTV video feeds, or, in extreme cases, cutting the video feed entirely when reduced to low bandwidth auxiliary connections.

A comprehensive list of use-cases is available on request, giving each product feature in context.