Magic Tunnel Daemon
mtund

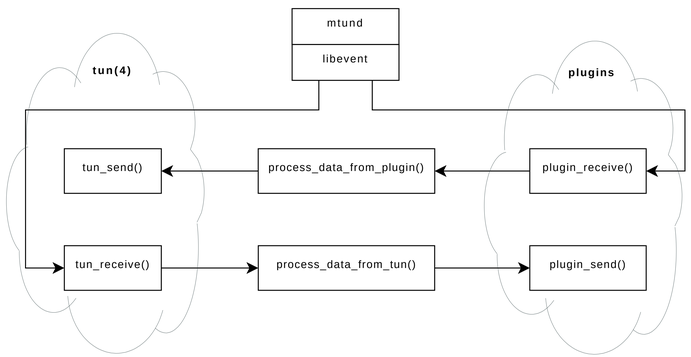
Matúš Harvan
Information Security, ETH Zürich, Switzerland

Introduction

IP can easily be tunneled over a plethora of network protocols at various layers, such as IP, ICMP, UDP, TCP, DNS, HTTP, SSH and many others. While a direct connection may not always be possible due to a firewall, the IP packets could be encapsulated as payload in other protocols, which would get through. However, each such encapsulation requires the setup of a different program and the user has to manually probe different encapsulations to find out which of them works in a given environment.
The Magic Tunnel Daemon (mtund) consists of a daemon and plugins. Each plugin implements a different encapsulation. The daemon automagically selects a working encapsulation in each environment, does the tunneling and can failover to another encapsulation if the environment changes.

The Daemon

- written in C
- using plugins for encapsulation (*dlopen(3)*)
- using *tun(4)* virtual interfaces
- using *libevent* for multiplexing



Features:

- failover between plugins
- probing and keep-alive “pings”
  - detects a broken encapsulation
  - keeps state in firewall
- multi-user support
  - one tun(4) interface per client
  - clients need to associate with the server
- fragmentation and fragment reassembly

Two types of encapsulations:

1. **direct** (TCP, UDP)
   - each side can send data anytime
2. **polling** (ICMP echo request/reply, DNS query/reply)
   - the client can send data anytime but the server can only send data in replies

Plugins

TCP plugin

- send tunneled IP packets as TCP payload in a TCP connection
- *framing* – prepend payload length before the actual payload so that the recipient knows where the tunneled packet ends within the TCP stream
- additional feature: listen on all unused TCP ports
  - sys patch – `TCP_LISTENALL` socket option

UDP plugin

- send tunneled IP packets as UDP payload in a UDP connection
- additional feature: listen on all unused UDP ports
  - sys patch – `net.inet.raw.udp_catchall` sysctl allows receiving unclaimed UDP packets on a raw IP socket, a new UDP socket then has to be bound/connected to the right ports/addresses

ICMP plugin

- using ICMP echo request/reply pairs to pass a stateful NAT gateway
- sys patch – `net.inet.icmp.echo_user` sysctl allows receiving ICMP echo requests on a raw IP socket

DNS plugin

- using DNS queries and replies
- DNS encoding and decoding taken from *iodine*
- if a DNS zone is properly delegated, connection to a working nameserver is sufficient and direct Internet connectivity is not needed (this is the case at many hotspots)

Missing Features

- more plugins
  - HTTP
  - SSH
  - ...
- config file format and parsing
- encryption, client authentication
  - protect tunnel control traffic
  - tunneled traffic can use IPSec on the tun(4) interface
- ICMP plugin probing and non-polling mode
  - instead of ICMP echo request/reply pairs a “direct” mode of operation could be used if the firewall allows it
  - use a different ICMP type so that kernel patching would not be required
  - different strategies for ICMP echo `ID` and `SEQ` fields
- DNS plugin should act as the UDP plugin if non-DNS traffic arrives
- MTU probing (can use probing pings)
- port to other BSDs, linux,... (currently only for FreeBSD)

More information available under
`http://wiki.freebsd.org/mtund`

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich