Pluggable Transports

Marionette as a Testbed for PT User Experience Research

David Oliver
Guardian Project

Ali Hussain
Univ. of Malaya

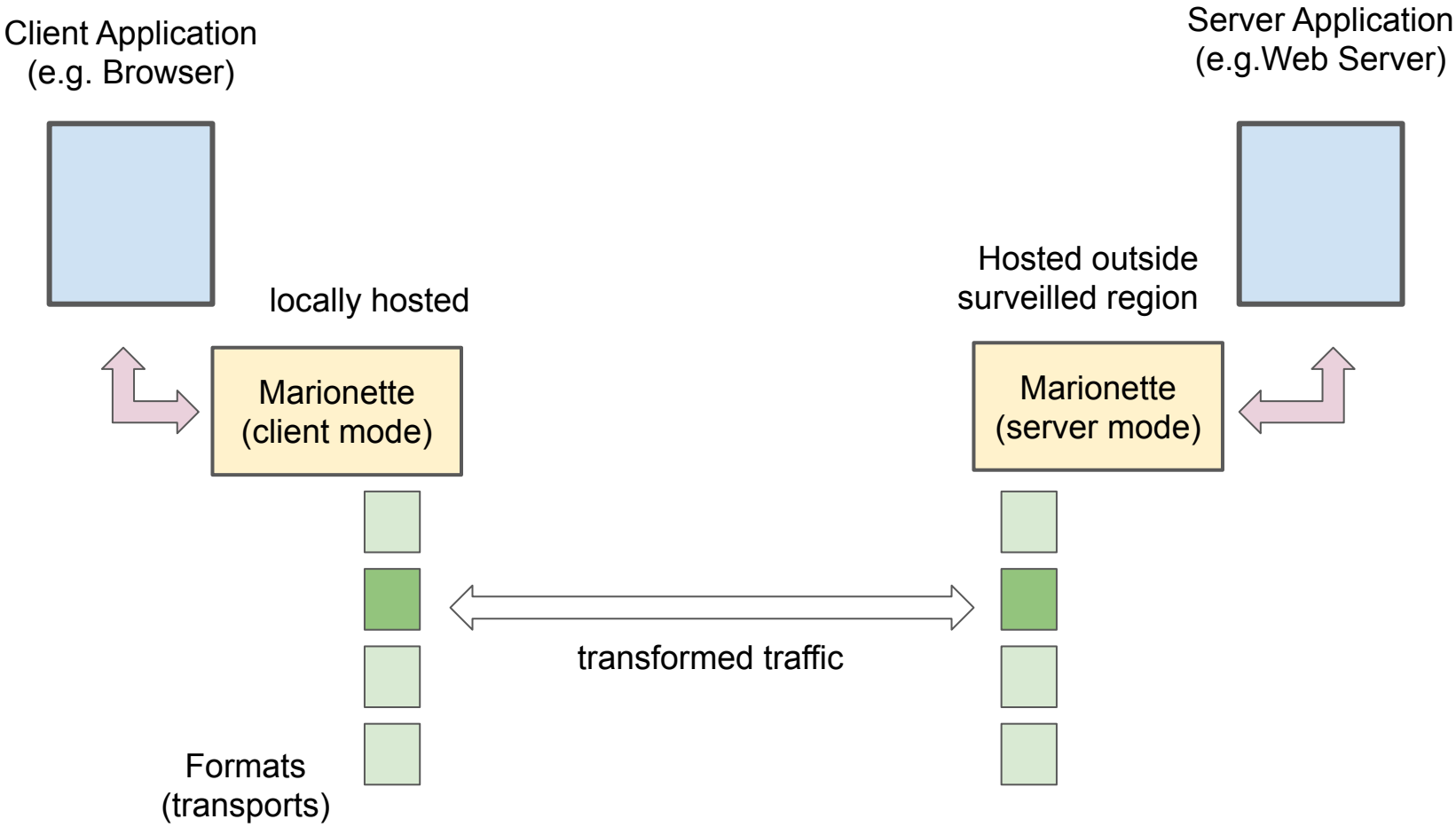
IETF106 Hackathon

TL/DR

- Pluggable Transports (PT) are a defense against Internet censorship, obfuscating network flows to protecting them from DPI surveillance
- Why “pluggable”? Because the half-life of a given obfuscation technique is short and rapid update is necessary
- With many PTs available, which is the right one to use?
- What combination of user-driven and machine-drive intelligence can be applied? How is this implemented technically?
- *Marionette* provides a convenient testbed environment to answer these questions

Marionette

- Golang implementation
- Wrapper around a FSM that abstracts content transforms into ‘formats’ expressed in plain text (and, in general, regex)
  - Format handles single- and multi-message flows
  - A ‘format’ is a Pluggable Transport
- Network proxy model
  - Client <-> local proxy <-> server proxy <-> Application Host
- Limits
  - Proxy pair hard-wired at client startup
  - Format also hard-wired at client startup
  - Predetermined client/server rendezvous


Client Application
(e.g. Browser)
Server Application
(e.g.Web Server)
locally hosted
Hosted outside
surveilled region
Marionette
(client mode)
Marionette
(server mode)
transformed traffic
Formats
(transports)

UX Work Items

- Client
  - Client-side proxy bootstrap
  - Client-side app configuration (non-browser)
    - Browser just standard SOCKS5
  - PT/format selection
  - Dynamic server selection / rendezvous
- Server
  - Serve multiple PTs/formats simultaneously?

Thanks

- Marionette team: https://github.com/redjack/marionette
- Internews - PT project support!
- IETF

More on Pluggable Transports

- Best single source: https://www.pluggabletransports.info/
- PT 2.1 Internet Draft: https://www.ietf.org/id/draft-oliver-pluggable-transports-00.txt