Project Description: Trust-Aware Tor Path Selection Prototype

Kevin Gallagher
Department of Computer Science
NOVA School of Science and Technology

Project Overview

In 2017, Aaron Johnson, Rob Jansen, Aaron D. Jaggard, Joan Feignenbaum, and Paul Syverson set out to create a new path selection algorithm that was based on trust in certain political realities, such as alliances and data sharing between different nation. The result of that is the Trust-Aware Path Selection (TAPS) algorithm, which chooses paths based on information about alliances between each country, etc. This project implements a simplified version of the Trust-Aware Path Selection (TAPS) algorithm from Johnson et al.'s NDSS 2017 paper [1]. To this end, you will create a standalone path selection tool that prioritizes anonymity through trust modeling, using pre-processed Tor relay data. We will be focusing only on the Countries policy for the TrustAll algorithm. We will simplify the path selection by assuming that countries connect directly to each other, and not worrying about the underlying realities of the Internet.

This project is to be done in pairs of two.

Learning Objectives

- Understand Tor's path selection challenges
- Implement trust-based relay scoring
- Explore security/performance tradeoffs

Technical Scope

- **Input**:
  - Pre-processed relay descriptors (JSON) containing:
    - Fingerprint
    - Nickname
    - IP
    - Bandwidth
    - Family
    - AS number
  - A configuration JSON that states:
    - Alliances: List of:
      - sets of which countries are collaborating (Specified as ISO 3166-1 alpha-2 country code), and

        - the user's trust in each of the countries.
    - Client: The client IP address.
    - Destination: The destination IP address.

- **Core Implementation**:
  - Trust scoring functions:

```
        def guard_security(client_loc, guards):
            # Calculate security score for guard set
            # based on client location and adversary model

        def exit_security(client_loc, dest_loc, guard, exit):
            # Score exit relay based on guard/destination
```

  - TrustAll parameters structure:

```
        def select_path(relays, alpha_params):
            #SUGGESTED_GUARD_PARAMS = {
            #    'safe_upper': 0.95,
            #    'safe_lower': 2.0,
            #    'accept_upper': 0.5,
            #    'accept_lower': 5.0,
            #    'bandwidth_frac': 0.2
            #}

            #SUGGESTED_EXIT_PARAMS = {
            #    'safe_upper': 0.95,
            #    'safe_lower': 2.0,
            #    'accept_upper': 0.1,
            #    'accept_lower': 10.0,
            #    'bandwidth_frac': 0.2
            #}
            # Sort relays by descending trust score
            # Separate into safe/acceptable categories (recommended
                values above)
            # Select until Bandwidth threshold reached.
            # Return bandwidth-weighted choice.
```

- **Output**: Guard-middle-exit path in JSON format

Evaluation Metrics

- Security: % of paths avoiding adversarial entities (8 values)
- Performance: Bandwidth utilization distribution (5 values)
- Robustness: Handling of edge cases (e.g., no valid paths) (5 values)
- Justification: Report describing why decisions were made (2 values)
- Creativity: Any improvements that can be made to the algorithm to improve it. (0.5 values)

Deliverables

- Implementation in language of your choice (I recommend using unit tests)
- Technical report (2 pages maximum) justifying decisions and analyzing tradeoffs

These deliverables should be sent to k.gallagher@fct.unl.pt on or before June 13th at 11h59.

Recommended Timeline (3 Weeks)

- **Week 1**: Read the article + trust scoring implementation
- **Week 2**: Path selection algorithm + basic testing
- **Week 3**: Edge case handling + final documentation

Prerequisites

- Programming skills (I honestly recommend a simple language like Python for this)
- Basic understanding of Tor architecture
- No prior experience with Shadow/Tor codebase required

References

[1] Johnson et al. "Avoiding The Man on the Wire: Improving Tor's Security with Trust-Aware Path Selection." NDSS 2017.

[2] Tor Directory Protocol Specification.
`https://gitweb.torproject.org/torspec.git/`