BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

FILE TRANSFER OVER A COVERT ICMP CHANNEL

PŘENOS SOUBORŮ PŘES SKRYTÝ ICMP KANÁL

AUTHOR **MICHAL REPČÍK**
AUTOR PRÁCE

BRNO 2025

REPČÍK, Michal. *File Transfer over a Covert ICMP Channel*. Brno, 2025. . Brno University of Technology, Faculty of Information Technology.

Contents

1 Overview **2**

2 Design **3**

Chapter 1

Overview

This documentation describes a client/server application that transfers files covertly using ICMP/ICMPv6 Echo-Request/Response packets. The app encrypts files with AES (using the user's login as the key) to keep data secure and splits large files into chunks to fit within standard network packet sizes (1500 bytes). The client sends the encrypted file to a specified IP or hostname, while the server listens for ICMP packets, decrypts them, and saves the file locally. A custom protocol ensures reliable transfer, including sending the filename and verifying the file is complete. The app is built in C++ using standard libraries, OpenSSL for encryption, and libpcap for packet handling, tested on GNU/Linux (merlin.fit.vutbr.cz).

Chapter 2

Design

The application is designed to address the need for secure, covert file transfer via ICMP/ICMPv6. Existing tools like `pingtunnel` lack encryption or robust file transfer protocols, so this app adds AES encryption and a custom protocol for reliability.

Key design features:

- **Encryption**: AES-256-CBC encrypts files using the user's login as the key.
- **Fragmentation**: Files larger than 1400 bytes (to fit ICMP payloads) are split into chunks.
- **Protocol**: Includes a header with filename (up to 255 bytes), file size, sequence number, and CRC32 checksum for integrity.
- **Operation Modes**: Client mode (`-r <file> -s <ip|hostname>`) sends files; server mode (`-l`) listens and saves files.

The app uses raw sockets for sending ICMP packets and libpcap for capturing them, supporting both IPv4 and IPv6. It's built to be portable, using only allowed libraries (e.g., `netinet/*`, OpenSSL, libpcap).