BENCHMARKING CURVECP

Thorben Krüger
benthor@os3.nl

July 4, 2011
OUTLINE

Introduction

Research Questions

Methods and Results

Problems

Conclusion
CurveCP?

Encrypted application-layer protocol for internet communication.
CurveCP in a Nutshell

- Brain child of Dan Bernstein (djb)
- To be used instead of TCP
- Packet-based encryption on top of UDP
- Treat crypto as instantaneous
- Advanced packet scheduler
Claimed CurveCP Security Features

- mandatory server authentication
- optional client authentication
- no man-in-the-middle attacks possible
- active and passive forward secrecy
Claimed CurveCP Availability Features

- no RST-type attacks possible
- protection against traffic prediction
- cannot be used for amplification attacks
- no SYN-flooding-type attacks possible
- worst-case CPU loads kept small
Claims about CurveCP Efficiency

- bigger overhead than plain TCP
- for short connections, less traffic than HTTPS
- for short connections, much less traffic than SSH
Claimed CurveCP Decongestion Features

- minimizes packet-loss
- minimizes significant latency increases
- therefore mitigates buffer bloat
Claimed CurveCP Addressing Features

- multiple CurveCP servers can share single IPv4 address and port
- CurveCP servers inherently anti-aliased from addresses
- rapid failover to redundant server if original is down
- session/connection not invalidated if IP address changes
**Major Topics for Investigation**

- Create CurveCP-enabled SSH-like remote shell
- Create CurveCP-enabled SCP-like remote file copy tool
- Benchmark CurveCP vs SSH/SCP/HTTPS
  - CPU usage
  - available Bandwidth
    - ideal
    - competitive
  - message latencies
- Verify CurveCP robustness claims
Remote Shell/Copy via CurveCP

Results:
▶ CurveCP-enabled remote ptty fully functional
▶ http://github.com/benthor/remotty

Problems:
▶ Only Python implementation so far
▶ Useless for meaningful benchmarks
▶ No real session handling yet
▶ Doesn’t support file transfer yet
**CurveCP-enabled file transfer?**

For now based on `cat/dd`

- No support for arbitrary files yet
**Bonus: CurveCP enabled VPN**

Results:
- Can create tunnel devices connected via CurveCP
- ICMP can be successfully tunneled

Problems:
- TCP only works for small packets so far
- Only Python implementation
Benchmarking: Methods

Tools:
- plain `dd`
- custom stream copy tool with statistics (`ddstat`)
- `sysstat` suite
- `ethtool`
Saturating 10Mbit Link

Results:

- CPU usage of SSH/SCP/HTTPS very similar

Problems:

- CurveCP scheduler is too nice
Best-case Payload Bandwidth on 10MBit Link

Results:
- SSH/SCP/HTTPS: nearly the full 10MBit/s
- CurveCP: between 600KB/s and 800KB/s, average 650KB/s

Problems:
- CurveCP scheduler is too nice
**CurveCP and TCP: Competitive Scheduling (10MBit Link)**

Problems:
- CurveCP bandwidth drops to 0
- no matter who starts first
- no matter if LAN or Internet connection
Trying to saturate 100Mbit Link

Results:
- SSH/SCP/HTTPS can saturate
- CurveCP probably CPU bound

Problems:
- Detailed measurements skew CPU usage
Best-case Payload Bandwidth on 100MBit Link

Results:
- SSH/SCP/HTTPS: nearly the full 12.5MB/s
- CurveCP: tops out at 4MB/s
CurveCP and TCP: Competitive Scheduling
(100MBit Link)

Results:
- CurveCP bandwidth 1/100th of that of TCP: 120KB/s
- More than one connection: share up to 240KB/s
**CurveCP vs SSH: Latencies**

Results:
- (Extremely) similar latencies to within fractions of ms

Problems:
- Does not take SSH handshake into account
**Bonus: CurveCP and Buffer Bloat**

Queue delay on top of minimum path RTT - bufferbloat-curvecp1

![Graph showing queue delay variations over time for different links with CurveCP and Buffer Bloat](image-url)
Questions not yet answered

Will be addressed in paper:
  ▶ Compare SSH handshake to CurveCP
  ▶ CurveCP overhead in general
  ▶ CurveCP addressing and failure modes
Encountered Problems
Problem: (Reverse) HeisenBugs

Packet scheduler gets confused:
  ▶ when UDP statistics are collected
  ▶ when in the presence of irregular TCP traffic
Result:
  ▶ endless tracing/profiling/code-reviewing
**Problem:** FreeBSD

- CurveCP pipes close prematurely
- file descriptor issues
Problem: Confusing CPU statistics

- disagreement between tools
- reported percentages add up to over 100
CONCLUSION

- Remote shell: works
- Remote copy: possible
- Performance: comparable to (but worse than) SSH/HTTPS
- Decongestion: works somewhat
- Verdict: CurveCP worthy of attention
More Info

http://curvecp.org
Questions?