TCP Analysis Quick Reference Guide

For Ethical Hackers

TCP 3-Way Handshake Flow

```
Client              Server
  |                   |
  |-------- SYN ----------->|  Step 1: Client requests connection
  |                   |
  |<----- SYN-ACK ----------|  Step 2: Server acknowledges & responds
  |                   |
  |-------- ACK ----------->|  Step 3: Client confirms connection
  |                   |
  |===== DATA TRANSFER =====|  Connection established
```

Packet Details

- **SYN**: Contains client's initial sequence number (ISN)
- **SYN-ACK**: Server's ISN + acknowledgment of client's SYN
- **ACK**: Client confirms server's sequence number

Wireshark Analysis Commands

Essential Filters

```
tcp.flags.syn == 1          # Show only SYN packets
tcp.flags.syn == 1 and tcp.flags.ack == 0   # SYN only (no ACK)
tcp.flags.syn == 1 and tcp.flags.ack == 1   # SYN-ACK packets
tcp.flags.ack == 1 and tcp.flags.syn == 0   # ACK only packets
tcp.flags.reset == 1        # RST packets (connection resets)
tcp.port == 80              # Filter by specific port
ip.addr == 192.168.1.100    # Filter by IP address
```

Analyzing Handshake Issues

```
tcp.analysis.retransmission    # Find retransmitted packets
tcp.analysis.duplicate_ack    # Duplicate ACKs (possible packet loss)
tcp.analysis.lost_segment    # Missing segments
tcp.seq == 0                # Initial SYN packets
```

Command Line Tools

tcpdump Commands

bash

```bash
# Capture all TCP traffic
sudo tcpdump tcp

# Capture SYN packets only
sudo tcpdump 'tcp[tcpflags] & tcp-syn != 0'

# Capture handshake to specific host
sudo tcpdump -i eth0 host 192.168.1.100 and tcp

# Save capture to file
sudo tcpdump -w capture.pcap tcp port 80

# Monitor specific port
sudo tcpdump port 443
```

nmap Port Scanning

bash

```
# Basic TCP port scan
nmap -p 80 192.168.1.100

# SYN scan (stealth scan)
nmap -sS 192.168.1.100

# Full TCP connect scan
nmap -sT 192.168.1.100

# Scan multiple ports
nmap -p 22,80,443 192.168.1.100

# Scan port range
nmap -p 1-1000 192.168.1.100
```

netcat Testing

```bash
# Test TCP connection
nc -v 192.168.1.100 80

# Listen on port (server mode)
nc -l 8080

# Test with timeout
nc -w 3 192.168.1.100 22

# Banner grabbing
echo "" | nc 192.168.1.100 80
```

Common Attack Patterns

SYN Flood Attack

Signature:

- Many SYN packets from various sources
- No corresponding ACK packets

- Target appears unresponsive

Wireshark Filter:

```
tcp.flags.syn == 1 and tcp.flags.ack == 0
```

TCP Reset Attack

Signature:

- Unexpected RST packets
- Connections terminated abruptly
- May indicate session hijacking

Wireshark Filter:

```
tcp.flags.reset == 1
```

Man-in-the-Middle Detection

Look for:

- Duplicate IP addresses with different MAC addresses
- Unexpected sequence number gaps
- Connection anomalies

Wireshark Filter:

```
tcp.analysis.duplicate_ack or tcp.analysis.retransmission
```

Normal vs Suspicious Behavior

Normal Handshake Indicators

- ✅ Complete SYN → SYN-ACK → ACK sequence
- ✅ Reasonable timing between packets (< 1 second)
- ✅ Proper sequence number progression

- ✅ Successful data transfer after handshake

Suspicious Patterns

- ❌ SYN packets without ACK responses
- ❌ Excessive retransmissions
- ❌ Unexpected connection resets
- ❌ Malformed sequence numbers
- ❌ Rapid-fire connections from single source

Security Testing Checklist

Connection Analysis

- [ ] Monitor handshake completion rates
- [ ] Check for excessive SYN retransmissions
- [ ] Verify proper sequence number handling
- [ ] Test connection timeout behaviors

Attack Detection

- [ ] Set up SYN flood detection rules
- [ ] Monitor for RST packet anomalies
- [ ] Check for duplicate MAC addresses
- [ ] Analyze connection timing patterns

Defense Testing

- [ ] Test firewall SYN flood protection
- [ ] Verify IDS handshake monitoring
- [ ] Check connection rate limiting
- [ ] Test proper connection cleanup

Quick Troubleshooting

Connection Fails

1. Check if target port is open: `nmap -p [port] [target]`

2. Verify firewall rules aren't blocking
3. Confirm service is running on target port
4. Check network connectivity: `ping [target]`

Slow Connections

1. Look for retransmissions in Wireshark
2. Check network latency: `traceroute [target]`
3. Monitor for packet loss
4. Verify MTU size issues

Connection Drops

1. Filter for RST packets in Wireshark
2. Check for keepalive failures
3. Monitor for network instability
4. Verify timeout configurations

Key TCP Flags Reference

Flag	Binary	Meaning
SYN	000010	Synchronize sequence numbers
ACK	010000	Acknowledgment field significant
RST	000100	Reset connection
FIN	000001	Finish - no more data
PSH	001000	Push - deliver immediately
URG	100000	Urgent pointer field significant