Ethical Hacking and Network Defense Lab Exam - Complete Guide

Based on your lab manual, you've got **11 experiments** covering four major tools: **Wireshark** (Experiments 1-3), **Hping** (4-5), **Nmap** (6-8), **iptables** (9-10), and **Autopsy** (11). Let me break down each experiment with procedures, commands, and critical exam tips from an ethical hacker instructor's perspective.

Experiment 1: Wireshark - Protocol Analysis (HTTP, TCP, UDP, DNS, ARP)

Procedure:

1. **Start Wireshark** in promiscuous mode on your primary network interface (eth0 or wlan0)
2. **Generate traffic** for each protocol:
   - HTTP: Browse to `http://example.com` (non-HTTPS sites)
   - TCP: Any web connection or SSH
   - UDP: DNS queries, streaming, `ping` command
   - DNS: Open any website
   - ARP: `ping` a device on local network
3. **Apply Display Filters:** [1] [2]
   - HTTP: `http` or `http.request`
   - TCP: `tcp` or `tcp.port == 80`
   - UDP: `udp` or `udp.port == 53`
   - DNS: `dns`
   - ARP: `arp`
4. **Apply Capture Filters (before capturing):** [1]
   - HTTP: `tcp port 80`
   - TCP: `tcp`
   - UDP: `udp`
   - DNS: `port 53`
   - ARP: `arp`

Pro Tips:

- **Syntax**: Display filters use `==` (e.g., `ip.addr == 192.168.1.1`), capture filters use single `=` or just keywords[2]
- **Green = valid filter, Red = syntax error**[3]
- Show **protocol hierarchy** (Statistics → Protocol Hierarchy) to demonstrate understanding
- Right-click packets → **Follow TCP Stream** to show complete conversations
- Explain **3-way handshake** (SYN, SYN-ACK, ACK) when analyzing TCP

Experiment 2: Wireshark - Statistics & Traffic Analysis

Procedure:

1. **Capture traffic** with promiscuous mode enabled
2. **Apply specific filters** as instructed (instructor will specify packet types)
3. **Navigate to Statistics tab:**
   - **I/O Graphs**: Statistics → I/O Graph[4] [5]
     - Shows packets per second over time
     - Modify X-axis (time intervals), Y-axis (packets/bytes/bits)
     - Add custom filters to individual graphs
   - **Conversations**: Statistics → Conversations (shows traffic between endpoints)
   - **Endpoints**: Statistics → Endpoints (traffic to/from specific addresses)
4. **ACL (Access Control List) identification:**
   - Filter malicious traffic patterns
   - Example: `ip.src == 192.168.1.100 && tcp.flags.syn == 1` for SYN attacks

Pro Tips:

- **I/O Graphs are powerful** - use them to visualize traffic spikes indicating attacks[4] [5]
- Change graph color schemes for multiple protocols simultaneously
- Export graphs as CSV for reports
- Use `tcp.stream eq X` to filter specific TCP conversations[2]
- Practice identifying normal vs. abnormal traffic patterns visually

Experiment 3: Wireshark - Data Compromise Detection

Procedure:

1. Visit **HTTP sites** (not HTTPS) - try `http://testphp.vulnweb.com` or local test servers
2. **Submit login credentials** through HTTP forms
3. **Filter for POST requests**: `http.request.method == POST`
4. **Locate credentials** in packet details → HTML Form URL Encoded
5. **Demonstrate other functionalities:**
    - Export objects (File → Export Objects → HTTP)
    - Time sequence graphs (Statistics → TCP Stream Graphs)
    - Expert Info (Analyze → Expert Information)

Pro Tips:

- **HTTPS traffic cannot be decrypted** without SSL keys - emphasize HTTP vulnerability
- Show how **cookies and session tokens** are exposed in HTTP
- Demonstrate **image extraction** from HTTP traffic
- Practice explaining why **encryption matters** (real-world context)
- Use `http contains "password"` to quickly find credential transmissions

Experiment 4: Hping - Spoofed, UDP & ICMP Flood Attacks

Procedure:

1. Spoofed Scan: [6]

```
sudo hping3 -S -p 80 <target-IP> -a <spoofed-IP>
```

2. UDP Flood Attack: [7]

```
sudo hping3 --udp -p 80 --flood <target-IP>
# Or with data size
sudo hping3 -2 -d 65495 --flood <target-IP>
```

3. ICMP Flood Attack: [7]

```
sudo hping3 --icmp --flood <target-IP>
# Or
sudo hping3 -1 -d 65495 --flood <target-IP>
```

Display in Wireshark:

- Start Wireshark capture before running hping3
- Filter: `icmp` or `udp` or `tcp.flags.syn == 1`

Pro Tips:

- Always use `sudo` for hping3 (requires root)
- **--flood** sends packets at maximum speed[6]
- `-d` flag sets payload size (larger = more bandwidth consumption)
- **-a** flag spoofs source IP (makes tracking difficult)[6]
- Watch Wireshark I/O graphs spike dramatically during floods
- Explain **DoS vs DDoS** concepts during demonstration

Experiment 5: Hping - Random Source & SYN Flood Attacks

Procedure:

1. Random Source Attack:[7]

```
sudo hping3 -S -p 80 <target-IP> --flood --rand-source
```

2. SYN Flood Attack (DDoS simulation):[6] [8]

```
sudo hping3 -S -p 80 <target-IP> --flood
# With random source for DDoS effect
sudo hping3 -S -p 80 <target-IP> --flood --rand-source
```

3. Set Packet Flags:[8]

```
# SYN flag
sudo hping3 -S <target-IP>
# FIN flag
sudo hping3 -F <target-IP>
# PUSH flag
sudo hping3 -P <target-IP>
# RESET flag
sudo hping3 -R <target-IP>
# ACK flag
sudo hping3 -A <target-IP>
# URG flag
sudo hping3 -U <target-IP>
# Multiple flags (XMAS scan)
sudo hping3 -FXYAP -p 80 <target-IP>
```

Pro Tips:

- **--rand-source** randomizes source IPs (simulates botnet)[6] [7]
- SYN floods create **half-open connections**, exhausting server resources[8]
- In Wireshark, filter `tcp.flags.syn == 1 && tcp.flags.ack == 0` to see SYN packets
- **Xmas scan** sets FIN, PSH, URG flags - named after lit-up Christmas tree
- Always run against **your own lab environment** or with permission
- Count SYN packets in Wireshark Statistics to demonstrate impact

Experiment 6: Nmap - Host Discovery & Ping Scans

Procedure:

1. Host Discovery:[9] [10]

```
# Single target
nmap www.srinivasuniversity.edu.in
nmap www.gmail.com

# Basic scan
nmap <target-IP>
```

2. Ping Scan (Host Alive Check):[10]

```
nmap -sn <target-IP>
# Scan subnet
nmap -sn 192.168.1.0/24
```

3. Host Scan:[9]

```
# Aggressive scan (OS detection, version, scripts, traceroute)
nmap -A <target-IP>
```

Display in Wireshark:

- Filter: `icmp` for ping sweeps
- Filter: `tcp.flags.syn == 1` for SYN scans

Pro Tips:

- **-sn** = ping scan only, no port scan[10]
- Nmap sends **ICMP echo, TCP SYN, and ARP** requests for discovery
- Use `-Pn` to **skip ping** and assume host is up[11]
- `-A` is noisy but comprehensive - good for lab demos[9]

- Show **latency and TTL** values in output to demonstrate network analysis
- Always check `/etc/hosts` if domain names resolve incorrectly

Experiment 7: Nmap - Port Scanning Techniques

Procedure:

1. Port Scanning: [9] [10]

```
# Scan specific ports
nmap -p 80,443,22 <target-IP>

# Scan port range
nmap -p 1-1000 <target-IP>

# Scan all ports
nmap -p- <target-IP>
```

2. SYN Scan (Stealth Scan): [10] [11]

```
sudo nmap -sS <target-IP>
```

3. TCP Connect Scan: [10] [11]

```
nmap -sT <target-IP>
```

Pro Tips:

- **-sS** requires sudo, faster and stealthier (doesn't complete handshake) [10] [11]
- **-sT** is default for non-root users, completes full TCP handshake [11]
- In Wireshark: SYN scan shows `SYN → SYN-ACK → RST`, Connect scan shows `SYN → SYN-ACK → ACK`
- Use `-T4` for faster scans (timing template) [10]
- `-F` fast scan (top 100 ports) [10]
- Explain **open vs closed vs filtered** port states
- Show how **firewalls block** certain scan types

Experiment 8: Nmap - Advanced Scans

Procedure:

1. UDP Scan: [10] [11]

```
sudo nmap -sU -p 53,161,162 <target-IP>
```

2. TCP INIT Scan (should be SCTP INIT): [12]

```
sudo nmap -sY <target-IP>
```

3. TCP NULL Scan: [10]

```
sudo nmap -sN <target-IP>
```

Pro Tips:

- **UDP scans are slow** - target specific ports (DNS-53, SNMP-161) [11]
- **NULL scan** sends packets with no flags set (firewall evasion) [10]
- Also practice: **FIN scan** (`-sF`), **XMAS scan** (`-sX`) [10]
- These scans exploit **TCP RFC loopholes** for stealth
- In Wireshark, NULL scan packets have **all flags = 0**
- UDP responses are slow/unreliable - watch for ICMP port unreachable messages
- Combine with `-Pn` if host seems down but isn't

Experiment 9: iptables - Basic Firewall Rules

Procedure:

1. View current rules: [13]

```
sudo iptables -L -v -n
```

2. Block specific IP address: [13] [14]

```
sudo iptables -A INPUT -s 203.0.113.51 -j DROP
```

3. Flush all rules: [15] [16]

```
sudo iptables -F
sudo iptables -X  # Delete custom chains
sudo iptables -t nat -F  # Flush NAT table
```

Display in Wireshark:

- Before blocking: ping and see responses
- After blocking: ping shows no responses
- Filter: `ip.addr == 203.0.113.51`

Pro Tips:

- **-A** = append rule, **-I** = insert at top (higher priority)
- **-s** = source, **-d** = destination
- **DROP vs REJECT**: DROP silently discards, REJECT sends error message[14]
- **-L** lists rules, **-v** verbose, **-n** numeric (no DNS resolution)
- Always **save rules** after changes: `sudo iptables-save > /etc/iptables/rules.v4`
- Test connectivity with `ping` before and after blocking
- Watch packet counters with `watch -n 1 sudo iptables -L -v -n`

Experiment 10: iptables - Advanced Blocking

Procedure:

1. Block IP on specific interface: [14]

```
sudo iptables -A INPUT -i eth0 -s 203.0.113.51 -j DROP
```

2. Block outgoing TCP traffic on specific port: [15]

```
sudo iptables -A OUTPUT -p tcp -d 203.0.113.51 --dport 1111 -j DROP
```

3. Delete rule by chain and number: [17]

```
# List rules with line numbers
sudo iptables -L INPUT --line-numbers

# Delete rule number 2 from INPUT chain
sudo iptables -D INPUT 2
```

Pro Tips:

- **-i** = input interface, **-o** = output interface
- **--dport** = destination port, **--sport** = source port[15]
- **Use line numbers** for easier rule management: `iptables -L --line-numbers`
- Test with **nc (netcat)** or **telnet** to verify port blocking

- Chains: **INPUT** (incoming), **OUTPUT** (outgoing), **FORWARD** (routing)
- Default policy matters: `sudo iptables -P INPUT DROP` makes everything denied by default
- Document rule numbers before deleting in exam

Experiment 11: Autopsy - Digital Forensics

Procedure:

1. **Launch Autopsy:**

```
autopsy
# Or use GUI application
```

2. **Create new case:** [18] [19]
   - Case name, base directory, investigator name
   - Case type: Single-user or Multi-user
3. **Add data source:** [19]
   - Disk Image/VM File
   - Local Disk
   - Logical Files
   - Select provided file (USB image, disk image, etc.)
4. **Configure ingest modules:** [19]
   - Recent Activity (browser history, downloads)
   - Hash Lookup (known files)
   - File Type Identification
   - Keyword Search
   - Email Parser
   - Extension Mismatch Detector
5. **Analyze results:** [18] [19]
   - **File Views**: Browse file system structure
   - **Data Artifacts**: Web history, cookies, bookmarks
   - **Keyword Search**: Search for specific terms
   - **Timeline**: Event timeline analysis
   - **Tags**: Mark evidence items
6. **Generate report:** [19]
   - Tools → Generate Report
   - HTML or Excel format

- Include tagged items

Pro Tips:

- **Patience**: Ingest takes time - don't skip it[18]
- **Hash analysis** identifies known good/bad files (NSRL database)[19]
- Look for **deleted files** in $OrphanFiles directory
- **Registry analysis** (Windows) shows user activity, USB history[18]
- **Browser artifacts** are goldmine: history, downloads, form data[18]
- **Timeline analysis** correlates events across system[19]
- **File carving** recovers deleted data from unallocated space[19]
- **EXIF data** from images reveals camera info, GPS, timestamps[19]
- Tag evidence systematically for cleaner reports
- Screenshot key findings for presentation

General Exam Strategy & Tricks

Before the Exam:

1. **Set up your environment:**
   - Kali Linux or Ubuntu with all tools installed
   - Two VMs: attacker and victim (or use vulnerable VMs like Metasploitable)
   - Network configured in bridged or NAT mode
   - Wireshark permissions set: `sudo usermod -aG wireshark $USER`
2. **Practice commands:**
   - Create cheat sheet with command syntax
   - Practice typing commands quickly without typos
   - Know your target IPs and interfaces
3. **Verify tools work:**

```
# Check installations
wireshark --version
hping3 --version
nmap --version
iptables --version
autopsy
```

During the Exam:

Time Management:

- Wireshark experiments: 15-20 min each
- Hping experiments: 10-15 min each
- Nmap experiments: 10-15 min each
- iptables experiments: 10-15 min each
- Autopsy: 20-30 min
- **Total: ~3 hours**

Critical Steps:

1. **Always start Wireshark BEFORE running attacks** - can't capture retrospectively
2. **Use sudo** for hping3, nmap scans, iptables
3. **Apply filters immediately** after capture - saves time
4. **Screenshot everything** - packet captures, terminal outputs, graphs
5. **Label your work** clearly (Experiment number, date, your name)
6. **Verify each step** before moving to next

Common Mistakes to Avoid:

- Forgetting sudo privileges
- Wrong network interface selected
- Mixing up capture vs display filters[1] [2]
- Not waiting for Autopsy ingest to complete
- Forgetting to save iptables rules
- Using HTTPS sites for Wireshark credential capture
- Running attacks on wrong IP address
- Not documenting findings

Pro Hacker Mindset:

- **Think like an attacker** for hping/nmap exercises - what would cause maximum impact?
- **Think like a defender** for Wireshark/iptables - how do I detect and block?
- **Think like an investigator** for Autopsy - what story does the evidence tell?
- **Always document** - in real-world scenarios, reporting is 50% of the job
- **Chain attacks** - show how nmap reconnaissance leads to targeted hping attacks
- **Explain the "why"** - don't just run commands, understand the underlying protocol behavior

Quick Command Reference Card:

```
# Wireshark
wireshark -i eth0 -k  # Start capturing on eth0

# Hping
sudo hping3 -S -p 80 --flood <IP>  # SYN flood
sudo hping3 --icmp --flood <IP>    # ICMP flood
sudo hping3 -2 --flood <IP>        # UDP flood

# Nmap
sudo nmap -sS <IP>                 # SYN scan
sudo nmap -A <IP>                  # Aggressive scan
sudo nmap -p- <IP>                 # All ports

# iptables
sudo iptables -A INPUT -s <IP> -j DROP     # Block IP
sudo iptables -L -v -n                     # List rules
sudo iptables -F                           # Flush rules
```

Final Tips for Tomorrow:

1. **Sleep well** - clarity beats cramming
2. **Arrive early** - troubleshoot tech issues before exam starts
3. **Read instructions carefully** - instructors may specify specific IPs/protocols
4. **Stay calm** - if something fails, move to next experiment and return later
5. **Ask for clarification** - if instructor requirements unclear
6. **Double-check syntax** - one typo can waste precious minutes
7. **Use tab completion** in terminal - faster and fewer errors
8. **Keep notes handy** - this guide + your own practice notes

You've got this! Your cybersecurity background and hands-on experience will shine through. Remember: examiners value **understanding over memorization** - explain what's happening at the protocol level, demonstrate critical thinking about security implications, and show systematic methodology.

Good luck! Break a leg (ethically, of course) 🍀

⁂

1. https://wiki.wireshark.org/CaptureFilters
2. https://petermoorey.github.io/wireshark-capture-verus-display-filters-copy/
3. https://www.geeksforgeeks.org/ethical-hacking/steps-of-filtering-while-capturing-in-wireshark/
4. https://www.wireshark.org/docs/wsug_html_chunked/ChStatIOGraphs.html
5. https://www.geeksforgeeks.org/ethical-hacking/i-o-graphs-window-in-wireshark/
6. https://gist.github.com/davidlares/0c2109b448302b8adecb837923cb1cc7

7. https://nccs.gov.in/public/events/DDoS_Presentation_17092024.pdf
8. https://clouddocs.f5.com/training/community/firewall/html/archive/archive2/module2/lab2.html
9. https://www.geeksforgeeks.org/ethical-hacking/nmap-cheat-sheet/
10. https://github.com/jasonniebauer/Nmap-Cheatsheet
11. https://hackertarget.com/nmap-cheatsheet-a-quick-reference-guide/
12. https://highon.coffee/blog/nmap-cheat-sheet/
13. https://www.servermania.com/kb/articles/ip-blocking-and-iptables-in-linux
14. https://www.digitalocean.com/community/tutorials/iptables-essentials-common-firewall-rules-and-commands
15. https://www.cherryservers.com/blog/how-to-manage-linux-system-routing-rules-with-iptables
16. https://www.geeksforgeeks.org/linux-unix/how-to-flush-the-iptables-and-clear-the-firewall-rules/
17. https://stackoverflow.com/questions/10197405/how-can-i-remove-specific-rules-from-iptables
18. https://www.ccslearningacademy.com/what-is-autopsy-in-cybersecurity/
19. http://www.autopsy.com/wp-content/uploads/sites/8/2016/02/Autopsy-4.0-EN-optimized.pdf
20. EHND-Lab-Manual-1.pdf
21. https://unit42.paloaltonetworks.com/using-wireshark-display-filter-expressions/
22. https://asec.ahnlab.com/en/85165/
23. https://www.chappell-university.com/post/wireshark-capture-filters-subnet-masks
24. https://www.linkedin.com/learning/ethical-hacking-denial-of-service-2/tcp-syn-flooding-using-hping3
25. https://www.tutorialspoint.com/nmap-cheat-sheet
26. https://www.varonis.com/blog/how-to-use-wireshark
27. https://onlinelibrary.wiley.com/doi/10.1155/2024/6651584
28. https://www.stationx.net/nmap-cheat-sheet/
29. https://www.youtube.com/watch?v=-HDpYR_QSFw
30. https://www.imperva.com/learn/ddos/ping-icmp-flood/
31. https://www.rosehosting.com/blog/blocking-abusive-ip-addresses-using-iptables-firewall-in-debianubuntu/
32. https://www.slideshare.net/slideshow/autopsy-digital-forensics-tool/145970562
33. https://www.youtube.com/watch?v=Vf2S-XeZoEQ
34. https://www.youtube.com/watch?v=q334lq-28e8
35. https://www.packtpub.com/en-in/learning/how-to-tutorials/statistical-tools-in-wireshark-for-packet-analysis
36. https://www.eurovps.com/faq/how-to-block-an-ip-address-on-a-linux-server/
37. https://www.geeksforgeeks.org/techtips/analysis-of-data-source-using-autopsy/
38. https://www.cellstream.com/2025/05/14/zero-to-hero-with-wireshark-tcp-graphs-a-tutorial/
39. https://www.autopsy.com
40. https://www.wireshark.org/docs/wsug_html_chunked/ChUseStatisticsMenuSection.html
41. https://www.sleuthkit.org/autopsy/