```
$ ./incident_response --start
```

Incident Response

for Linux Administrators

An Open Source Approach to Security Incidents

Linux | Open Source | Security

KWLUG // No servers were harmed in the making of this presentation

What is Incident Response?

Your game plan for when things go sideways, detecting breaches, stopping the bleeding, and getting back to normal.

277

days avg breach detection (IBM 2024)

Why It Matters

- Minimize damage and reduce recovery time
- Preserve evidence for investigation
- Prevent future incidents through lessons learned
- Stay on the right side of compliance audits

```
# The goal:
detect in
minutes,
not months
```

The 6 Phases of Incident Response

Based on NIST SP 800-61

1
Preparation
Build your toolkit, document procedures, train your team

2
Identification
Detect and determine if an incident has occurred

3
Containment
Stop the spread and limit damage while preserving evidence

4
Eradication
Remove malware, close vulnerabilities, eliminate root cause

5
Recovery
Restore systems to production, verify functionality

6
Lessons Learned
Document findings, improve processes, share knowledge

continuous cycle ▶

1 Preparation

Before the incident happens

Essential Preparation Steps

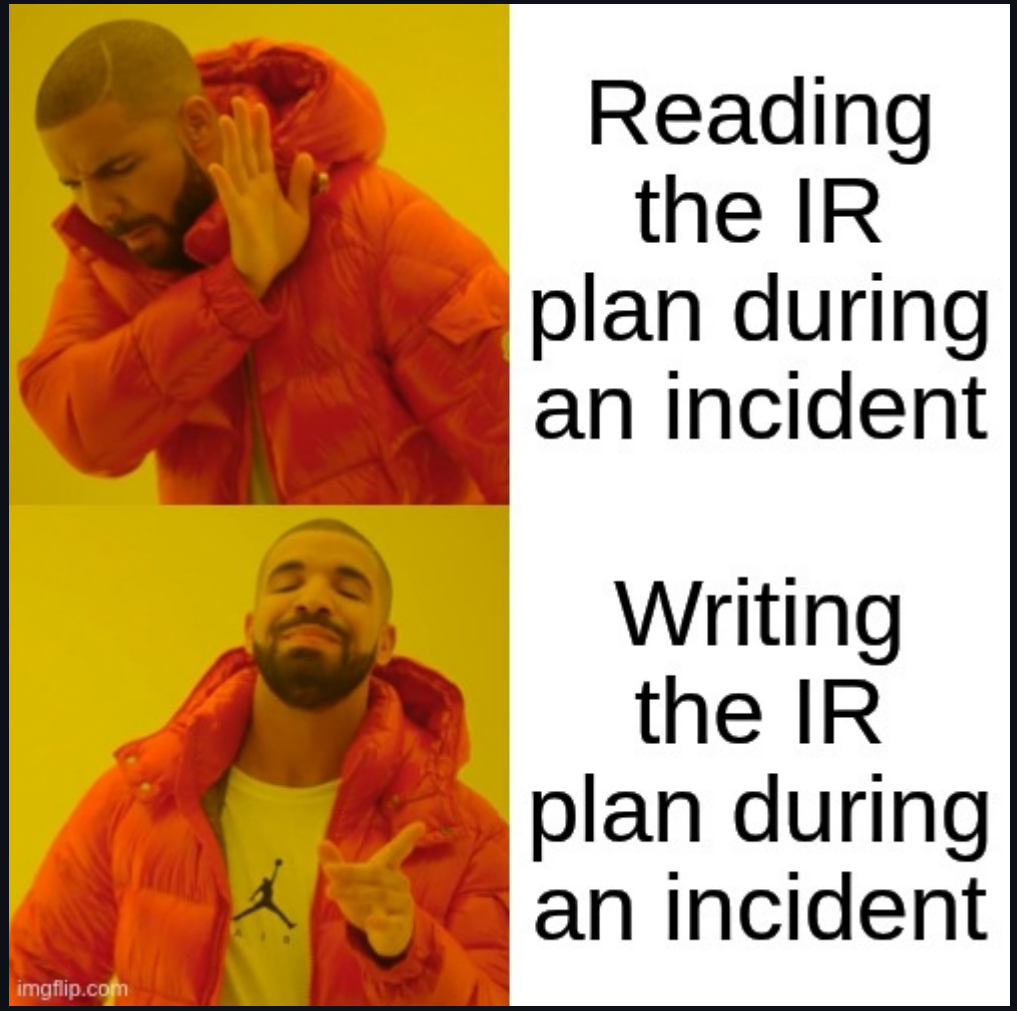
- **Document your network**
  Asset inventory, network diagrams, baseline configs
- **Build response toolkit**
  Bootable USB, forensic tools, live response scripts
- **Create runbooks**
  Step-by-step procedures for common incidents
- **Establish communication plan**
  Contact lists, escalation paths, out-of-band channels

IR USB Toolkit Basics

Essential tools to include:

- `volatility3`
- `sleuthkit / autopsy`
- `chkrootkit / rkhunter`
- `tcpdump / wireshark`
- `lsof, netstat, ss`
- `dd / dcfldd`
- `strings, grep, find`
- `yara`

Pro Tip: Keep static binaries, compromised systems may have trojaned utilities.



Don’t be this person

2 Identification

Detecting that something is wrong

Common Indicators of Compromise

- Unusual outbound network traffic
- Unexpected privilege escalation attempts
- Modified system files or binaries
- Unknown processes or services
- Failed login attempts from unusual sources
- Unexpected cron jobs or scheduled tasks

Quick Check Commands

```
# Active connections
ss -tulpn | grep LISTEN

# Running processes
ps auxf --sort=-%cpu | head -20

# Recent logins
last -a | head -20

# Modified files (last 24h)
find /etc -mtime -1 -ls
```

Key Question: Is this normal behavior, or is something anomalous?

Open Source Detection Tools

The tools that do the job and they won't cost you a dime

Wazuh

SIEM + XDR Platform

- Log analysis and correlation
- File integrity monitoring
- Rootkit detection
- Active response
- Compliance reporting

wazuh.com

Suricata

Network IDS/IPS

- Deep packet inspection
- Protocol identification
- Multi-threaded engine
- Compatible with Snort rules
- JSON logging

suricata.io

OSSEC

Host-based IDS

- Log-based intrusion detection
- File integrity checking
- Rootkit detection
- Real-time alerting
- Cross-platform agents

ossec.net


This is you at 2 AM when
Wazuh starts pinging your phone.
THIS IS FINE.
imgflip.com

3 + 4 Containment + Eradication

Stop the bleeding, remove the threat

Containment Actions

- Isolate affected systems from the network
- Block malicious IPs/domains at firewall
- Disable compromised user accounts
- Take forensic image BEFORE cleanup

Eradication Actions

- Remove malware and backdoors
- Patch exploited vulnerabilities
- Reset all potentially compromised credentials
- Rebuild from known-good sources if needed

```
# Quick network isolation

iptables -I INPUT -j DROP
iptables -I OUTPUT -j DROP
```

Critical Reminder

If you can't trust the kernel, you can't trust anything. Consider full rebuild for rootkit infections.

Scenario: The 2 AM Cryptominer

Walking through a real-world incident

02:00

Alert fires

Wazuh alerts on CPU spike, host at 100% for 15 min

```
$ top -b -n1 | head -15
```

02:10

Identify the process

Unknown binary /tmp/.x/kworker running as www-data

```
$ ls -la /proc/$(pgrep kworker)/exe
```

02:20

Trace the entry point

Apache logs show exploit of unpatched plugin

```
$ grep 'POST.*upload'
/var/log/apache2/access.log
```

02:35

Contain + eradicate

Isolate host, image disk, kill process, patch vuln

```
$ iptables -I OUTPUT -j DROP && dd
if=/dev/sda ...
```

What NOT to Do During an Incident

Learn from other people's mistakes

Don't reboot the system	You'll lose volatile memory, running processes, network connections, loaded modules. Memory forensics becomes impossible.
Don't wipe before imaging	Always dd the disk first. Once you rm -rf the malware, the evidence is gone. Courts and insurance companies want the original.
Don't post about it publicly	No tweets, no Slack #general, no Reddit posts. You don't know what the attacker is monitoring. Always use out-of-band comms.
Don't change passwords on a compromised box	If there's a keylogger, you just gave them your new credentials. Reset from a clean machine.
Don't just 'patch and pray'	Patching the exploited vuln doesn't remove backdoors already installed. Assume persistence until proven otherwise.

```
$ sudo reboot --please-dont-be-hacked    # this is not a real flag, and not a real plan
```


SYSADMIN
PROPER
FORENSIC IMAGING
JUST REBOOT IT
imgflip.com

Open Source Forensics Toolkit

Memory dumps and Malware analysis

Volatility 3

Memory forensics framework: The Gold Standard

1. Capture

```
sudo insmod lime.ko 'path=/tmp/mem.raw format=raw'
```

2. Processes

```
vol -f mem.raw linux.pslist
```

3. Network

```
vol -f mem.raw linux.sockstat
```

4. Rootkits

```
vol -f mem.raw linux.check_syscall
```

Autopsy

Digital forensics GUI: timeline analysis, keyword search, file carving, hash filtering. Best for disk image analysis when you prefer visual over CLI.

TheHive + Cortex + MISP

Case management platform. Cortex auto-enriches IOCs (VirusTotal, AbuseIPDB). MISP shares threat intel with the community.

Also Worth Knowing

YARA: Malware pattern matching

ClamAV: Open source antivirus

Velociraptor: Endpoint visibility at scale

GRR: Google's remote forensics

5 + 6 Recovery + Lessons Learned

Get back online and figure out what went wrong

Recovery Checklist

- [ ] Restore from verified clean backups
- [ ] Validate system integrity before reconnecting
- [ ] Monitor for signs of re-infection
- [ ] Phased return to production
- [ ] Verify all credentials have been rotated

Post-Incident Review

Document and discuss:

- Timeline of events
- How was the incident detected?
- What worked well?
- What could be improved?
- What tools/processes were missing?
- Root cause analysis

Blameless Culture: Focus on process improvement, not individual blame. People report problems faster when they're not afraid of consequences.


WAIT...
YOU GUYS ARE DOING POST-MORTEMS?
imgflip.com

Common Linux Attack Patterns

Know what to look for

Privilege Escalation

- SUID binary exploitation
- Kernel exploits (DirtyPipe, Copy Fail)
- Sudo misconfigurations
- Writable PATH directories

Detection:

```
find / -perm /4000 -ls
```

Persistence Mechanisms

- Crontab entries / systemd timers
- SSH authorized_keys injection
- Modified .bashrc / .profile
- Rogue systemd services

Detection:

```
systemctl list-unit-files --state=enabled
```

Lateral Movement

- SSH key harvesting
- Credential reuse from configs
- Internal network scanning
- Exploiting NFS / SMB shares

Detection:

```
ss -antp | grep ESTABLISHED
```

Essential Linux Commands for IR

Quick reference cheat sheet

Process & Users

```
# Running processes
ps auxf

# Process tree
pstree -p

# Open files by PID
lsof -p [PID]

# Logged in users
w; who; last

# User cron jobs
crontab -l -u [user]
```

Network

```
# Listening ports
ss -tulpn

# Active connections
ss -antp

# Network by process
lsof -i

# Capture traffic
tcpdump -i eth0 -w out.pcap

# ARP cache
ip neigh show
```

Files & Logs

```
# Recently modified
find / -mtime -1 -ls

# SUID/SGID files
find / -perm /6000 -ls

# Auth logs
journalctl -u sshd

# File hashes
sha256sum [file]

# Strings in binary
strings [file] | less
```

Tip: Use static binaries from your IR toolkit, system utilities may be compromised

Live Demo: IR in Action

Let's actually run some of these tools

Demo 1: Baseline vs Compromised

Run the cheat sheet commands on a clean VM, then introduce a simulated threat and run them again.

Demo 2: Wazuh Dashboard Tour

Walk through the Wazuh web UI, show real alerts, file integrity events, and how to investigate from alert to root cause.

Demo 3: Memory Forensics

Analyze a pre-captured memory dump, find the hidden process, trace network connections, identify the malware.

Try It Yourself!

Open a terminal and run:

```
ss -tulpn
find / -perm /4000 -ls 2>/dev/null
```

Building Your IR Toolkit

Clone the repo, start tonight

DETECTION

Wazuh · Suricata · Zeek

ANALYSIS

Volatility · Autopsy · YARA

MANAGEMENT

TheHive · MISP · Cortex

github.com/Thomas-Busch-Waterloo/linux-ir-toolkit

Setup script · Static binaries · Cheat sheets · Wazuh docker-compose · Sample configs

Start Tonight

- git clone the IR toolkit repo
- Run the setup script on a spare VM
- Build your forensic USB

This Month

- Deploy Wazuh on your network
- Add Suricata for network monitoring
- Write your first runbook

Questions?

Let's discuss incident response in your environment

Resources & Further Reading

NIST SP 800-61
Computer Security Incident Handling Guide

SANS Reading Room
sans.org/reading-room

Linux Security Wiki
wiki.archlinux.org/title/Security

MITRE ATT&CK for Linux
attack.mitre.org/matrices/enterprise/linux

Key Takeaways

- Prepare before incidents happen
- Open source tools are production-ready
- Document everything
- Practice your response
- Learn from every incident

```
$ echo "Thanks for attending!"
```