

Infrastructure Security Hardening: Firewall Audit, IDS Tuning, and SIEM Alert Management

Wed, Feb 11, 9:20 AM EST · <https://scottslab.io/posts/infrastructure-security-hardening-firewall-ids-siem>



TL;DR

Audited 67 pfSense rules and every NAT forward, then killed 21 WAN pass rules and 6 forwards that had no business being open. Rewrote the Zeek C2 whitelists and cut false positives by roughly 90%, then fixed a batch of Wazuh rules that were quietly broken by XML errors and colliding rule IDs. Also chased down a DNS redirect loop on the IoT VLAN and trimmed the agent inventory from 21 to 16.

Published: February 2026 **Category:** Security Operations **Reading Time:** 15 minutes

Executive Summary

- Audited 67 firewall rules and all NAT port forwards; disabled 21 dangerous WAN rules and 6 risky NAT forwards
- Reduced WAN attack surface from 21+ open pass rules to only necessary NAT-associated rules
- Rewrote Zeek C2 detection whitelists, reducing false positive alerts by ~90%
- Fixed broken Wazuh SIEM rules including XML syntax errors, rule ID collisions, and 10 new suppression rules for known infrastructure traffic

- Resolved a DNS redirect loop on IoT VLAN that was generating persistent firewall errors
 - Cleaned up SIEM agent inventory from 21 to 16 active agents
-

Goal

Problem: Over time, my perimeter firewall had accumulated dangerous rules through quick-fix troubleshooting sessions. "Pass any to any" rules, catch-all NAT forwards, and exposed management interfaces were sitting in the ruleset alongside legitimate traffic rules. Meanwhile, my network IDS was flooding alerts from legitimate infrastructure traffic (cloud tunnels, VPN relays, CDN connections), and the SIEM had broken rules that either weren't firing or were generating noise from known-good sources.

Why it mattered: A firewall with 21 overly permissive rules isn't a firewall. It's a suggestion. When your IDS generates hundreds of false positives daily, real threats get lost in the noise. Security tools that cry wolf eventually get ignored, which defeats their entire purpose.

Scope and Constraints

In Scope

- Perimeter firewall rule audit and hardening
- NAT port forward review and cleanup
- Zeek C2 detection script whitelist tuning
- Wazuh SIEM rule syntax fixes and alert suppression
- SIEM agent cleanup (decommissioned hosts)

Out of Scope

- Internal VLAN firewall rules (LAN-to-LAN traffic)
- Endpoint detection and response (EDR) tuning
- Application-layer security (WAF rules)
- Backup and disaster recovery validation

Key Constraints

- **Zero downtime requirement** - Production services must remain accessible throughout changes
- **No service exposure changes** - Cloud tunnel architecture handles most service exposure; I'm hardening, not redesigning

- **Preserve forensic capability** - Suppression rules must not hide actual threats, only known-good infrastructure traffic
-

Tools and References

Tool	Role in This Project
pfSense	Perimeter firewall - rule audit, NAT cleanup, DNS redirect fixes
Zeek	Network IDS - C2 detection via long-lived connections, DNS tunneling detection, certificate anomalies
Wazuh	SIEM - alert aggregation, rule management, agent inventory
Cloudflare Tunnels	Service exposure - outbound-only tunnels eliminate need for inbound NAT
Tailscale	Mesh VPN - secure internal access, DERP relay traffic appears in IDS
Pi-hole	DNS filtering - dedicated appliance replaced non-functional firewall package

References:

- [pfSense Firewall Rule Best Practices](#)
 - [Zeek Intelligence Framework](#)
 - [Wazuh Custom Rules Syntax](#)
 - [MITRE ATT&CK Network Defense](#)
-

Approach

Phase 1: Firewall Rule Audit

What I did: Exported all 67 firewall rules and NAT entries. Reviewed each rule against three criteria: (1) Is there a documented business purpose? (2) Is the scope as narrow as possible? (3) Is the rule still needed?

Why: Firewall rules accumulate like sediment. Every "temporary" troubleshooting rule that never gets removed expands your attack surface. You can't secure what you haven't inventoried.

Phase 2: Dangerous Rule Remediation

What I did: Disabled 21 WAN firewall rules including pass-any rules, catch-all NAT passes, exposed management interfaces (SSH, remote console), and stale EasyRule entries from years-old

troubleshooting sessions. Disabled 6 NAT forwards that were either catch-all rules or exposed sensitive services.

Why: A single "pass any to any" rule negates every other rule in your firewall. Exposed management interfaces are prime targets for credential stuffing and exploitation.

Phase 3: NAT Cleanup and Fixes

What I did: Replaced a dangerous "forward everything" NAT rule with a specific rule for the media server on its correct port. Fixed a DNS redirect NAT on the IoT VLAN where source/destination logic was inverted, creating a DNS resolution loop. Corrected a host alias pointing to the wrong IP.

Why: Catch-all NAT rules are the "sudo rm -rf" of network security: they solve the immediate problem by creating a much bigger one. The DNS loop was generating persistent `pf` errors on every firewall reload.

Phase 4: Zeek C2 Detection Tuning

What I did: Rewrote the C2 detection script with comprehensive whitelists covering: cloud tunnel provider IP ranges, VPN relay server IPs, VPN coordination server addresses, legitimate long-domain services (CDNs, analytics, streaming services), and reverse DNS lookup sources.

Why: Zeek was flagging every Cloudflare tunnel connection as a potential C2 channel (long-lived encrypted connection to external IP). VPN relay traffic triggered the same alerts. Without whitelists, the legitimate traffic volume made the alerts useless.

Phase 5: Wazuh SIEM Rule Fixes

What I did: Fixed XML syntax errors in custom rules, resolved a rule ID collision where a duplicate ID was silently overriding another rule, added 6 suppression rules for pfSense alerts from known infrastructure, and added 4 suppression rules for Zeek alerts from legitimate tunnel/DNS traffic.

Why: Wazuh won't load rules with XML errors, but duplicate rule IDs fail silently: the second definition just overwrites the first. Alert fatigue from known-good traffic trains analysts to ignore the console.

Phase 6: Agent Cleanup

What I did: Identified and removed 5 decommissioned SIEM agents (hosts no longer in service). Reconnected 1 agent that had lost connectivity.

Why: Stale agents clutter the inventory and can mask coverage gaps. If you think you have 21 monitored hosts but only 16 are actually reporting, you have blind spots.

Implementation Notes

Firewall Rule Disable Pattern (Sanitized)

```
# Export current rules for backup
<FIREWALL_CLI> config backup > /backup/fw-rules-$(date +%Y%m%d).xml

# Disable rule by tracker ID (pfSense uses tracker IDs, not rule numbers)
# This is done via GUI or config.xml edit - CLI shown for illustration
<FIREWALL_CLI> rule disable --tracker <RULE_TRACKER_ID>

# Reload firewall
<FIREWALL_CLI> filter reload
```

Assumption: Your firewall supports rule disable (not just delete) to allow easy rollback.

Zeek Whitelist Structure (Sanitized)

```
# c2-whitelist.zeek - Legitimate long-connection sources

module C2Detection;

export {
    # Cloud tunnel provider subnets
    const cloud_tunnel_nets: set[subnet] = {
        <CLOUD_PROVIDER_CIDR_1>,
        <CLOUD_PROVIDER_CIDR_2>,
        <CLOUD_PROVIDER_CIDR_3>,
    } &redef;

    # VPN relay IPs (mesh VPN DERP servers)
    const vpn_relay_ips: set[addr] = {
        <VPN_RELAY_IP_1>,
        <VPN_RELAY_IP_2>,
        <VPN_RELAY_IP_3>,
    } &redef;

    # Legitimate long-domain services (CDNs, analytics)
    const legit_long_domains: pattern = /\.
(ccloudfront|akamai|fastly|analytics|telemetry)\./;
}

# Skip alerting if destination matches whitelist
event connection_state_remove(c: connection) {
    if (c$id$resp_h in cloud_tunnel_nets) return;
    if (c$id$resp_h in vpn_relay_ips) return;
    # ... existing C2 detection logic
}
```

Wazuh Suppression Rule Pattern (Sanitized)

```
<!-- Suppress pfSense alerts from known cloud provider subnet -->
<rule id="100201" level="0">
  <if_sid>87701</if_sid>
  <srcip><CLOUD_PROVIDER_CIDR></srcip>
  <description>Suppress: Known cloud tunnel provider traffic</description>
</rule>

<!-- WRONG: Comma-separated CIDRs not supported -->
<!-- <srcip><CIDR_1>,<CIDR_2></srcip> -->

<!-- CORRECT: One rule per subnet -->
<rule id="100202" level="0">
  <if_sid>87701</if_sid>
  <srcip><CLOUD_PROVIDER_CIDR_2></srcip>
  <description>Suppress: Known cloud tunnel provider traffic (range 2)
</description>
</rule>

<!-- Field matching requires explicit pcre2 type for regex -->
<rule id="100210" level="0">
  <if_sid>87900</if_sid>
  <field name="dns.query" type="pcre2">.*\.(cloudprovider|cdnprovider)\.com$</field>
  <description>Suppress: Known CDN DNS queries</description>
</rule>
```

DNS Redirect Fix (Sanitized)

```
# BEFORE (inverted logic - caused DNS loop)
NAT Rule: Redirect DNS
  Source: <IOT_VLAN_SUBNET>
  Destination: <DNS_SERVER_IP> # WRONG - was catching replies
  Redirect to: <DNS_SINKHOLE_IP>

# AFTER (correct logic)
NAT Rule: Redirect DNS
  Source: <IOT_VLAN_SUBNET>
  Destination: !<DNS_SINKHOLE_IP> # NOT the sinkhole
  Destination Port: 53
  Redirect to: <DNS_SINKHOLE_IP>
```

Validation and Evidence

Signals That Proved It Worked

Check	Before	After
WAN pass rules	21 overly permissive	3 NAT-associated only
NAT port forwards	9 (including catch-all)	4 specific rules
Zeek C2 alerts/day	~150 (mostly false positives)	~15 (legitimate investigation targets)
Wazuh rule validation	3 XML errors, 1 ID collision	Clean load, no errors
Active SIEM agents	21 (5 stale)	16 (all reporting)
pfSense filter reload	DNS redirect errors	Clean reload

Validation Commands (Sanitized)

```
# Verify Wazuh rules load cleanly
<WAZUH_BIN>/wazuh-analysisd -t
# Expected: No errors

# Check Zeek script syntax
<ZEEK_BIN>/zeek -a <POLICY_DIR>/c2-whitelist.zeek
# Expected: No errors

# Verify firewall rule count
<FIREWALL_CLI> rule count --interface WAN --action pass
# Expected: Reduced count

# Test service accessibility through cloud tunnel
curl -I https://<SERVICE_DOMAIN>
# Expected: 200 OK (services still accessible)
```

Results

Metric	Outcome
WAN Attack Surface	Reduced from 21+ open pass rules to 3 NAT-associated rules
Dangerous NAT Forwards	Eliminated 6 (catch-all, SSH, SMTP, remote console)
Zeek False Positives	Reduced ~90% (cloud tunnel, VPN, CDN traffic no longer triggers)
Wazuh Alert Noise	Infrastructure IP alerts suppressed (10 new rules)
DNS Loop	Fixed - IoT VLAN DNS redirect now functions correctly
SIEM Coverage	Cleaned from 21 to 16 verified active agents
Service Availability	100% - all services remained accessible throughout

What I Learned

1. **Cloud tunnels obsolete most inbound NAT.** Services connect outbound through encrypted tunnels. You don't need to punch holes in your firewall when your services are dialing out to the tunnel provider.
 2. **Catch-all NAT rules are technical debt with interest.** Every "forward everything to this host" rule added during troubleshooting becomes a permanent attack surface expansion. They're easy to add and easy to forget.
 3. **Quarterly firewall audits are non-negotiable.** Rules from decommissioned services persist forever. If you're not actively pruning, you're passively expanding your attack surface.
 4. **Wazuh's `<srcip>` tag is single-value only.** No comma-separated CIDR lists, no pipe alternation. One rule per subnet. The documentation doesn't make this obvious.
 5. **Wazuh's `<field>` tag defaults to OS_Match, not regex.** If you want regex matching, you MUST specify `type="pcre2"`. Your patterns will silently fail otherwise.
 6. **Duplicate Wazuh rule IDs fail silently.** The second rule just overwrites the first. No error, no warning. You can have detection gaps and never know it.
 7. **Zeek whitelists need both IP and domain exclusions.** VPN relays trigger long-connection alerts (IP-based). CDN/cloud services trigger DNS tunneling alerts (domain-based). You need both.
 8. **VPN mesh relay IPs are dynamic and distributed.** Services like Tailscale host DERP relays across various cloud providers. You can't whitelist them with a single broad subnet. You need to track individual relay IPs or resolve them dynamically.
 9. **FreeBSD uses `sed -i ''` not `sed -i`.** Small difference, critical when scripting changes on pfSense. The GNU vs BSD sed distinction bites everyone eventually.
 10. **Orphaned flags create invalid firewall syntax.** Removing a NAT source address without also removing the "not" flag creates `from !` with no address. The firewall won't load the ruleset.
-

What I Would Improve Next

P0 (Do This Week)

- **Automated firewall rule audit** - Monthly script that flags unused rules (no hits in 30 days) and dangerous patterns (any-any, wide open ports)

- **Media server investigation** - Port not listening on expected port; service may need restart or troubleshooting (pre-existing issue, not caused by these changes)

P1 (Do This Month)

- **Wazuh active response for port scans** - Auto-block IPs at firewall when Zeek detects repeated scanning behavior
- **Firewall block trends dashboard** - Wazuh visualization of top blocked sources, ports, frequency over time
- **Dynamic Tailscale DERP whitelist** - Cron job to resolve `derp*.tailscale.com` and update Zeek whitelist automatically

P2 (Do This Quarter)

- **Split DNS implementation** - Separate internal/external resolution to avoid hairpin NAT issues
 - **GeoIP firewall rules** - Block inbound from countries with no business purpose (reduces noise and attack surface)
 - **Alert escalation playbook** - Document which Zeek/Wazuh alerts require immediate action vs. weekly review
 - **Firewall rule documentation** - Spreadsheet with owner, purpose, and last-reviewed date for every rule
-

Common Failure Modes

1. **"I disabled a rule and now X is broken"** - Cloud tunnel services should handle most access. If something breaks, check if it was relying on a NAT forward that should have been migrated to tunnel exposure.
2. **"Zeek alerts came back after whitelist update"** - The whitelist file may not have been deployed to all Zeek workers, or the script has a syntax error preventing load. Check `zeekctl status` and script validation.
3. **"Wazuh rules aren't suppressing alerts"** - Verify rule ID is unique, XML syntax is valid, and `<srcip>` contains a single value (not a list). Run `wazuh-analysisd -t` to validate.
4. **"pfSense shows filter reload errors"** - Look for orphaned flags in NAT rules (negation without address), invalid alias references, or circular DNS redirect rules. Check `/tmp/rules.debug`.

5. **"SIEM agent shows disconnected but host is running"** - Agent may have lost connectivity after IP change or firewall rule change. Restart the agent service and verify it can reach the Wazuh manager.
-

Security Considerations

Attack Surface Reduction

- Eliminated direct WAN exposure for SSH, SMTP, and remote console services
- Removed catch-all NAT rules that effectively bypassed firewall policy
- Reduced permissive pass rules from 21 to 3 (NAT-associated only)

Detection Integrity

- Suppression rules are scoped to specific source IPs/subnets, not blanket disables
- Rule ID collision fix restored detection coverage that was silently missing
- Agent cleanup ensures SIEM coverage matches actual infrastructure

Defense in Depth

- Cloud tunnels provide application-layer authentication even if network layer is compromised
- VPN mesh ensures internal traffic doesn't traverse untrusted networks
- DNS filtering at dedicated appliance provides redundant protection if firewall DNS package fails

Least Privilege

- NAT forwards now specify exact ports, not port ranges or "all"
 - Firewall rules use specific host aliases, not subnet-wide permissions where possible
-

Runbook

If Services Become Inaccessible

1. **Check cloud tunnel status** - Most services use outbound tunnels; verify tunnel daemon is running
2. **Review recent firewall changes** - Check if a required NAT rule was disabled; re-enable if needed
3. **Verify DNS resolution** - IoT VLAN DNS redirect issue could recur if misconfigured

- 4. **Check firewall filter log** - Look for blocks on the affected traffic in pfSense logs
- 5. **Test from multiple networks** - Issue may be client-side or ISP-related, not firewall

If Alert Volume Spikes

- 1. **Check for new infrastructure** - New service deployments may not be whitelisted yet
- 2. **Verify whitelist deployment** - Zeek whitelists must be on all workers; Wazuh rules must pass validation
- 3. **Look for actual attack** - Not all spikes are false positives; correlate with firewall blocks
- 4. **Check for rule ID collision** - New custom rules may have duplicated an existing ID
- 5. **Review suppression rule scope** - Ensure rules match on correct fields (srcip, dstip, field name)

Rollback Procedure

```
# Restore firewall config from backup
<FIREWALL_CLI> config restore /backup/fw-rules-<DATE>.xml
<FIREWALL_CLI> filter reload

# Restore previous Zeek whitelist
cp /backup/c2-whitelist-<DATE>.zeek <POLICY_DIR>/c2-whitelist.zeek
<ZEEK_CTL> deploy

# Restore previous Wazuh rules
cp /backup/local_rules-<DATE>.xml <WAZUH_DIR>/etc/rules/local_rules.xml
<WAZUH_CTL> -R
```

Appendix

Glossary

Term	Definition
NAT (Network Address Translation)	Translates private IPs to public IPs; port forwarding is a type of NAT
IDS (Intrusion Detection System)	Monitors network traffic for suspicious patterns
SIEM (Security Information and Event Management)	Aggregates and correlates security logs from multiple sources

Term	Definition
C2 (Command and Control)	Communication channel between malware and attacker infrastructure
DERP (Designated Encrypted Relay for Packets)	Tailscale's relay servers for NAT traversal
DGA (Domain Generation Algorithm)	Malware technique that generates pseudo-random domains to evade blocking
Hairpin NAT	Traffic that exits and re-enters the same interface; often causes routing issues

MITRE ATT&CK Mapping

Technique ID	Name	How I Addressed It
T1190	Exploit Public-Facing Application	Disabled catch-all NAT rules that exposed internal services to WAN
T1133	External Remote Services	Removed SSH and remote console exposure on WAN interface
T1071	Application Layer Protocol	Zeek C2 detection via long-lived connection analysis (with whitelist tuning)
T1071.004	DNS	DNS tunneling and exfiltration detection via Zeek (with CDN/cloud whitelist)
T1046	Network Service Discovery	Port scan detection via Zeek connection analysis
T1568.002	Domain Generation Algorithms	DGA detection in Zeek with false positive tuning for legitimate long domains
T1573.002	Encrypted Channel: Asymmetric Cryptography	Self-signed certificate detection for potential C2 identification
T1021	Remote Services	Lateral movement detection via honeypot monitoring in Wazuh

Detection Logic Summary

Detection	Tool	Logic	Whitelist Applied
Long-lived connections	Zeek	Connections >30min to external IPs	Cloud tunnel subnets, VPN relay IPs
DNS tunneling	Zeek	High query volume, long subdomains, TXT record abuse	CDN domains, cloud API domains
DGA domains	Zeek	High entropy domain names, rapid NXDOMAIN responses	Analytics/telemetry services
Self-signed certs	Zeek	Certificates not in known CA list	Internal PKI, development domains
Firewall blocks	Wazuh	pfSense block events aggregated	Known scanner IPs, cloud provider ranges
Authentication failures	Wazuh	Repeated failed logins across services	None (all auth failures logged)
Port scans	Zeek + Wazuh	Connection attempts to multiple ports from single source	Internal vulnerability scanners

Artifacts Produced

- **Config: Firewall Backup** - Pre-change backup of all firewall rules and NAT entries
- **Script: Zeek C2 Whitelist** - Comprehensive whitelist for cloud tunnels, VPN relays, and CDN traffic
- **Rules: Wazuh pfSense Suppressions** - 6 rules suppressing known infrastructure firewall alerts
- **Rules: Wazuh Zeek Suppressions** - 4 rules suppressing known-good Zeek alerts
- **Documentation: Disabled Rule Inventory** - List of all disabled rules with original purpose and disable rationale
- **Documentation: NAT Cleanup Log** - Before/after state of all NAT port forwards

Questions or war stories about your own firewall archaeology expeditions? Find me in the usual places.
