Enhanced Security Engineering Interview Prep Guide

Complete 30-Week Curriculum

Comprehensive Security Engineering Interview Preparation

Generated: December 13, 2025

Duration	30 weeks (parallel to AppSec curriculum)
Time Commitment	10-14 hours/week (additional to AppSec study)
Total Investment	360+ hours across 30 weeks
Target Roles	Security Engineer, AppSec Engineer, Detection Engineer
Based On	Grace Nolan's Notes + Team Blind 85 Interviews Post

Table of Contents

Executive Summary

This enhanced 30-week curriculum provides comprehensive Security Engineering interview preparation alongside your focused AppSec curriculum. While your AppSec study emphasizes offensive security skills (vulnerability discovery, exploitation, secure code review), this guide covers broader defensive and infrastructure security topics commonly tested in Security Engineering interviews at companies like Google, Amazon, Microsoft, Trail of Bits, and other major tech firms.

What's New in This Enhanced Version

This version integrates critical missing content identified from Team Blind's 85 Security Engineer Interviews post and Grace Nolan's comprehensive study notes:

- **CIA Triad Fundamentals** - Explicit coverage of Confidentiality, Integrity, Availability (Week 1)
- **Email Security & Phishing Analysis** - Email header analysis, attachment sandboxing, gateway security (Week 9)
- **Vulnerability Management** - Asset discovery, vulnerability scanning, CVSS prioritization (Week 11)
- **VPN Attack Vectors** - Credential attacks, protocol vulnerabilities, session hijacking (Week 3)
- **Network Troubleshooting** - Systematic debugging methodology for common issues (Week 2)
- **5-Step Incident Response** - NIST framework memorization and application (Week 10)
- **Windows EDR & Endpoint Security** - Microsoft Defender, application whitelisting, event log forensics (Week 7)
- **Security-Focused Coding Practice** - Emphasized throughout with log parsers, web scrapers, security tools (Weeks 17-20)

Key Differentiators

Your AppSec curriculum already covers web application security, API security, and OWASP Top 10 in depth. This guide focuses on complementary domains:

- Networking fundamentals (OSI model, TCP/IP, DNS, TLS)
- Operating systems internals (Linux, Windows)
- Cryptography theory (beyond implementation)
- Detection and monitoring (SIEM, IDS, log analysis)
- Incident response and forensics
- Malware analysis and reverse engineering
- Security architecture and threat modeling

Critical Success Factors from Team Blind Analysis

According to the candidate who completed 85 Security Engineer on-sites at top tech companies (Team Blind, June 2022):

1. **Coding ability is the #1 reason candidates fail Security Engineering interviews**
2. Breadth and depth across all OSI model layers is essential

3. Memorization of security fundamentals (CIA triad, IR steps) is non-negotiable
4. Practical scenario questions test real-world problem-solving
5. Cultural fit and communication skills account for ~50% of offer decisions

How to Use This Guide

This 30-week study plan runs in parallel with your 28-week AppSec curriculum, requiring an additional 10-14 hours per week. The topics are organized into weekly modules that align with your existing schedule, with each week covering 2-3 related concepts with specific study materials and practical exercises.

Study Approach (from Grace Nolan's Notes)

- **Spaced Repetition System:** Track concepts in three categories: 'To Learn', 'Revising', 'Done'
- **Active Recall:** Use post-it notes - one term per note, move between categories as you master them
- **Verbal Practice:** Practice explaining concepts out loud, even to yourself
- **Deep Understanding:** Focus on understanding WHY security controls exist, not just memorizing definitions
- **Hands-On Labs:** Apply knowledge through coding challenges and labs weekly
- **Sleep Review:** Use falling asleep time to mentally review terms (Grace guarantees <10 min to sleep!)

Interview Preparation Insights

Common Failure Points (Team Blind Analysis):

- **Insufficient coding practice** - Most critical weakness
- **Lack of fundamental memorization** - CIA triad, IR steps, OSI layers
- **Inability to troubleshoot practical scenarios** - Network issues, email analysis
- **Missing breadth across security domains** - Need all OSI layers, not just web

30-Week Curriculum Overview

Phase	Weeks	Focus Areas
Phase 1	Weeks 1-4	Networking & Systems Foundation + CIA Triad
Phase 2	Weeks 5-7	Cryptography & Binary Exploitation + Windows Security
Phase 3	Weeks 8-10	Detection, Monitoring & Incident Response + Email Security
Phase 4	Weeks 11-14	Malware Analysis, Reverse Engineering & Vulnerability Management
Phase 5	Weeks 15-16	Cloud Security & Container Security
Phase 6	Weeks 17-20	Security Coding Challenges & Tool Development
Phase 7	Weeks 21-25	Mock Interviews & System Design Practice
Phase 8	Weeks 26-30	Company-Specific Prep & Final Review

Phase 1: Networking & Systems Foundation (Weeks 1-4)

Parallel to AppSec Phase 1: Python + SQL Injection Mastery

Total Hours: 48 hours (12 hours/week)

Week 1: TCP/IP Fundamentals, CIA Triad & Network Protocols

Core Topics

- **CIA Triad Fundamentals (CRITICAL - memorize this):**
  - **C**onfidentiality: Encryption, access controls, data classification
  - **I**ntegrity: Hashing, digital signatures, checksums, tamper detection
  - **A**vailability: Redundancy, DDoS mitigation, disaster recovery
  - **AAA Extension:** Authentication, Authorization, Accounting/Auditing
- OSI Model: All 7 layers, encapsulation, data packet names at each layer
- TCP/IP: 3-way handshake, TCP vs UDP, common port numbers (80, 443, 22, 3389, 21, 25)
- IP Addressing: IPv4/IPv6, subnetting, CIDR notation, NAT concepts, private vs public IPs

Reading Materials

- Omnisecu TCP/IP Tutorial (https://www.omnisecu.com/tcpip/) - OSI Model, TCP/IP, IPv4/IPv6 sections
- High Performance Browser Networking by Ilya Grigorik (Chapter 1-3) - Free online
- Beej's Guide to Network Programming - Free online (beej.us/guide/bgnet/)

Labs & Practical Exercises

- Wireshark packet capture: Capture and analyze HTTP, HTTPS, DNS traffic
- Identify all TCP 3-way handshake packets (SYN, SYN-ACK, ACK)
- **Coding Challenge:** Write a simple port scanner in Python using socket library
- **Memorization Exercise:** Create flashcards for CIA triad, OSI layers (mnemonics: All People Seem To Need Data Processing, Please Do Not Throw Sausage Pizza Away)

Time Allocation: 12 hours

- Reading: 5 hours
- Labs: 4 hours
- Coding: 2 hours
- Flashcard creation & review: 1 hour

Week 2: DNS, TLS/SSL & Network Troubleshooting

Core Topics

- DNS: Resolution process, record types (A, AAAA, MX, CNAME, TXT, NS, SOA), DNSSEC
- TLS/SSL: Handshake process, certificate chains, certificate transparency logs
- Common TLS attacks: Heartbleed, POODLE, BEAST, downgrade attacks
- **Network Troubleshooting Methodology (Team Blind emphasis):**
  - Scenario: Website 404 but others can access - systematic debugging

    - Steps: Check browser cache/cookies, DNS resolution (nslookup, dig), firewall/proxy, permissions, browser-specific issues

Reading Materials

- High Performance Browser Networking (Chapter 4: TLS)
- Omnisecu: DNS sections
- SANS Reading Room: DNS Security whitepapers

Labs & Practical Exercises

- Use openssl to inspect certificates, analyze certificate chains, verify certificate validity
- DNS enumeration: Use dig, nslookup, host commands for various record types
- **Coding Challenge:** Build a certificate expiration checker that monitors domains and alerts on expiring certs
- **Troubleshooting Practice:** Create 5 network troubleshooting scenarios and document systematic approach to each

Time Allocation: 12 hours

- Reading: 4 hours
- Labs: 5 hours
- Coding: 2 hours
- Troubleshooting scenarios: 1 hour

Week 3: Firewalls, VPNs & VPN Attack Vectors

Core Topics

- Firewall types: Packet filtering, stateful, application layer, next-gen firewalls
- iptables/nftables basics: Rules, chains, tables
- VPN protocols: IPsec, OpenVPN, WireGuard, site-to-site vs remote access
- **VPN Attack Vectors (Team Blind emphasis):**
    - Credential attacks: MFA bypass, weak passwords, credential stuffing
    - Protocol vulnerabilities: Known CVEs (Heartbleed in OpenVPN, IPsec vulnerabilities)
    - Split tunneling risks: Data leakage outside VPN tunnel
    - VPN session hijacking: Session token theft, session fixation
    - Certificate validation bypasses: MITM attacks, weak certificate validation
- Network segmentation: VLANs, DMZ architecture, Zero Trust principles

Reading Materials

- NIST SP 800-41r1: Guidelines on Firewalls and Firewall Policy
- NIST SP 800-77: Guide to IPsec VPNs
- SANS Reading Room: VPN security whitepapers

Labs & Practical Exercises

- Configure iptables rules on Linux VM: Allow SSH, block all other inbound
- Test firewall behavior with nmap scans before/after rules
- Set up OpenVPN server/client, test connectivity

- **Practical Exercise:** Design network architecture for fictional company with proper segmentation (public DMZ, internal network, management network)
- **Attack Practice:** Document how you would test each VPN attack vector in a penetration test

Time Allocation: 12 hours

- Reading: 4 hours
- Labs: 6 hours
- Architecture design: 2 hours

Week 4: Linux OS Internals & System Calls

Core Topics

- Linux file system: Hierarchy (FHS), file permissions (chmod, chown, ACLs)
- Linux processes: Process creation (fork, exec), signals, process states
- System calls: open, read, write, socket, bind, listen, accept
- Linux security mechanisms: SELinux, AppArmor, capabilities, namespaces, cgroups
- User/group management: /etc/passwd, /etc/shadow, sudo configuration

Reading Materials

- The Linux Programming Interface by Michael Kerrisk (Chapters 1-5, 24-27)
- Linux man pages: Read man pages for chmod, chown, fork, exec, signals

Labs & Practical Exercises

- Practice file permissions: Create files with different permissions, test access as different users
- Write C program using fork() and exec() to spawn child processes
- Use strace to trace system calls of running programs (e.g., ls, cat, netstat)
- **Security Exercise:** Configure SELinux policies to restrict application access

Time Allocation: 12 hours

- Reading: 5 hours
- Labs: 5 hours
- Coding: 2 hours

Phase 2: Cryptography & Binary Exploitation (Weeks 5-7)

Parallel to AppSec Phase 2: XSS, CSRF, Access Control

Total Hours: 36 hours (12 hours/week)

Week 5: Cryptography Fundamentals Beyond Implementation

Core Topics

- Symmetric encryption: AES modes (CBC, GCM, CTR, ECB), authenticated encryption (AEAD)
- Block cipher modes: Differences, security properties, when to use each
- Asymmetric encryption: RSA, ECC, key sizes, performance trade-offs
- Key exchange: Diffie-Hellman, ECDH, forward secrecy
- Hash functions: SHA-256, SHA-3, collision resistance, preimage resistance
- Message authentication: HMAC, MAC vs digital signatures
- Password hashing: bcrypt, Argon2, PBKDF2, salting, work factors
- Common crypto vulnerabilities: Padding oracle, timing attacks, IV reuse, ECB penguin

Reading Materials

- Serious Cryptography by Jean-Philippe Aumasson (Chapters 1-8)
- CryptoHack.org documentation and challenges

Labs & Practical Exercises

- Complete 10-15 CryptoHack challenges (cryptohack.org) - beginner level
- Demonstrate padding oracle attack on CBC mode encryption
- Explain difference between GCM and CBC modes - when to use each
- **Interview Prep:** Practice explaining crypto concepts out loud without notes

Time Allocation: 12 hours

- Reading: 6 hours
- CryptoHack challenges: 5 hours
- Practice explanations: 1 hour

Week 6: Memory Corruption & Binary Exploitation Basics

Core Topics

- Stack layout: Stack frames, return addresses, saved base pointers
- Buffer overflows: Stack-based, heap-based, return-to-libc
- Modern mitigations: Stack canaries, NX/DEP, ASLR, PIE, CFI
- Heap exploitation: Use-after-free, double-free, heap overflow
- Format string vulnerabilities: Reading/writing arbitrary memory

Reading Materials

- Hacking: The Art of Exploitation by Jon Erickson (Chapters 3-5)
- SANS Reading Room: Memory corruption whitepapers

Labs & Practical Exercises

- Complete 10 PicoCTF binary exploitation challenges

- Complete 5-10 pwnable.kr challenges (beginner level)
- Write vulnerable C program and exploit it in controlled environment
- Practice with gdb: Set breakpoints, examine stack, memory

Time Allocation: 12 hours

- Reading: 4 hours
- CTF challenges: 6 hours
- Coding/debugging: 2 hours

Week 7: Windows Security Architecture & EDR

Core Topics

- Windows authentication: Kerberos, NTLM, pass-the-hash, golden tickets
- Active Directory: Domains, forests, trusts, group policy, LDAP
- Windows access control: ACLs, security descriptors, tokens, privileges
- **Windows EDR & Endpoint Security (Grace Nolan emphasis):**
  - Microsoft Defender for Endpoint (formerly Defender ATP)
  - AMSI (Antimalware Scan Interface) for PowerShell/script scanning
  - Credential Guard: Virtualization-based security for credentials
  - Application whitelisting: AppLocker, Windows Defender Application Control (WDAC)
  - PowerShell logging: Script block logging, transcription, module logging
  - Windows Event Logs: Event IDs 4624 (logon), 4625 (failed logon), 4688 (process creation)
- Common Windows attacks: Mimikatz, pass-the-hash, Kerberoasting, DCSync

Reading Materials

- Windows Internals by Russinovich (Chapters 6-7) OR Microsoft security documentation
- Microsoft Defender for Endpoint documentation
- SANS Reading Room: Active Directory security whitepapers

Labs & Practical Exercises

- Set up Windows VM with Active Directory domain controller
- Practice basic AD enumeration: Users, groups, computers, GPOs
- Configure AppLocker policies to whitelist specific applications
- Enable PowerShell logging and analyze logs for suspicious activity
- Analyze Windows Event Logs for logon events (4624, 4625)
- **Interview Prep:** Explain how Kerberos authentication works in Active Directory

Time Allocation: 12 hours

- Reading: 5 hours
- Labs: 6 hours
- Practice explanations: 1 hour

Phase 3: Detection, Monitoring & Incident Response (Weeks 8-10)

Parallel to AppSec Phase 3: Authentication & Authorization

Total Hours: 42 hours (14 hours/week)

Week 8: Log Analysis & SIEM Fundamentals

Core Topics

- Common log sources: syslog, Windows Event Logs, web server logs, firewall logs, proxy logs
- Log aggregation: Centralized logging, log forwarders (Fluentd, Logstash)
- Log parsing & normalization: Structured vs unstructured logs, field extraction
- SIEM basics: Splunk, ELK stack (Elasticsearch, Logstash, Kibana), detection rules, alerts
- Key IOCs in logs: Suspicious IPs, domains, file hashes, user agents, unusual access patterns

Reading Materials

- Applied Security Visualization by Raffael Marty (Chapters 1-4)
- Splunk/ELK documentation: Getting started guides

Labs & Practical Exercises

- Install ELK stack locally (Docker recommended)
- Ingest sample logs (Apache, auth.log, firewall logs)
- Create detection rules for: Brute force attacks, port scanning, SQL injection attempts
- **Coding Challenge:** Write log parser to extract IOCs (IPs, domains, file hashes) from mixed logs

Time Allocation: 14 hours

- Reading: 4 hours
- Labs: 7 hours
- Coding: 3 hours

Week 9: Network IDS & Email Security

Core Topics

- IDS vs IPS: Signature-based vs anomaly-based detection
- Snort/Suricata: Rule syntax, common attack patterns in network traffic
- Network forensics: PCAP analysis techniques, packet carving
- Common network attacks: Port scans, DDoS, C2 beaconing, lateral movement
- **Email Security & Phishing Analysis (Team Blind emphasis):**
  - Email header analysis: SPF, DKIM, DMARC validation, authentication results
  - Suspicious attachment analysis: Sandboxing, detonation, hash reputation

    - Phishing indicators: Urgency tactics, spoofed sender domains, suspicious links
    - Email gateway security: Proofpoint, Mimecast, Microsoft 365 ATP
    - Safe handling: Never open attachments directly, use VMs/sandboxes

Reading Materials

- The Practice of Network Security Monitoring by Richard Bejtlich
- SANS SEC487: Open-Source Intelligence (free intro materials on email analysis)

Labs & Practical Exercises

- Analyze 5-10 malicious PCAP files from malware-traffic-analysis.net
- Write Suricata rules to detect: SQL injection in HTTP, SSH brute force, TLS certificate anomalies
- Analyze phishing email samples: Extract headers, check SPF/DKIM/DMARC, identify red flags
- Practice detonating suspicious attachments in sandbox (Any.run, Hybrid Analysis)
- **Interview Scenario:** 'How do you analyze a suspicious email attachment?' - practice answering this

Time Allocation: 14 hours

- Reading: 4 hours
- PCAP analysis labs: 5 hours
- Email security labs: 4 hours
- Interview practice: 1 hour

Week 10: Incident Response & Digital Forensics

Core Topics

- **5-Step NIST Incident Response Framework (CRITICAL - memorize this):**
    - **1. Preparation:** IR playbooks, team training, tooling, communication plans
    - **2. Identification:** Detection, triage, scoping, severity classification
    - **3. Containment:** Short-term: Isolate affected systems; Long-term: Persistent isolation
    - **4. Eradication:** Remove threat, patch vulnerabilities, rebuild systems
    - **5. Recovery:** Restore services, enhanced monitoring, verification
    - **6. Lessons Learned:** Post-mortem, process improvements (SANS adds this as 6th step)
- Incident classification: SEV1/SEV2/SEV3, impact assessment
- Evidence collection: Chain of custody, forensic imaging, memory dumps
- Disk forensics: File systems (NTFS, ext4), deleted file recovery, timeline analysis
- Memory forensics: Volatility framework, process analysis, network connections

Reading Materials

- NIST SP 800-61r2: Computer Security Incident Handling Guide
- SANS Incident Handler's Handbook (free PDF)

Labs & Practical Exercises

- Analyze compromised system image using Autopsy or similar tool
- Practice Volatility: Analyze memory dump for malicious processes, network connections
- **Create IR Playbook:** Document incident response process for web application compromise
- **Memorization Exercise:** Create flashcards for 5-step IR process, practice reciting from memory
- **Interview Practice:** Explain IR process for ransomware attack scenario

Time Allocation: 14 hours

- Reading: 5 hours
- Labs: 6 hours
- Playbook creation: 2 hours
- Memorization & practice: 1 hour

Phase 4: Malware Analysis & Vulnerability Management (Weeks 11-14)

Parallel to AppSec Phase 4: SAST/DAST & Security Automation

Total Hours: 52 hours (13 hours/week)

Week 11: Malware Analysis & Vulnerability Management

Core Topics

- Malware types: Viruses, worms, trojans, ransomware, rootkits, APTs
- Static analysis: Strings, PE structure, imports/exports, entropy analysis
- Dynamic analysis: Sandbox, behavior monitoring, network traffic
- **Vulnerability Management (Team Blind emphasis):**
  - Vulnerability scanners: Nessus, Qualys, OpenVAS, configuration
  - Asset discovery: Network scanning, Shodan, passive reconnaissance
  - CVSS scoring: Severity calculation, prioritization, risk assessment
  - Patch management: Vulnerability lifecycle, SLA-based remediation
  - Continuous monitoring: Automated scanning, vulnerability tracking

Reading & Labs

- Practical Malware Analysis by Sikorski & Honig (Chapters 1-7)
- NIST SP 800-40r4: Enterprise Patch Management Planning
- Labs: Analyze 5-10 malware samples (PEStudio, Ghidra, Wireshark)
- Labs: Install Nessus Essentials, scan home lab network
- **Coding:** Write malware signature scanner using YARA rules
- **Interview Question:** 'How do you find vulnerable software on workstations?' - practice answer

Week 12: Reverse Engineering Basics

Core Topics

- x86/x64 assembly: Common instructions (mov, push, pop, jmp, call, ret)
- Function calling conventions: cdecl, stdcall, stack frames
- Disassemblers: IDA Pro, Ghidra, Binary Ninja
- Debuggers: gdb, x64dbg, WinDbg
- Code obfuscation: Packing, unpacking, anti-debugging techniques

Labs & Exercises

- Complete 10 crackme challenges from crackmes.one (beginner level)
- Reverse engineer simple keygenme to understand algorithm
- Reading: Reverse Engineering for Beginners by Dennis Yurichev (free online)

Week 13: Threat Modeling & Security Architecture

Core Topics

- **STRIDE framework:** Spoofing, Tampering, Repudiation, Information Disclosure, DoS, Elevation
- Attack trees: Systematic threat decomposition

- Data flow diagrams: Trust boundaries, data flows, processes
- Secure design principles: Defense in depth, least privilege, fail securely, minimize attack surface
- **YOUR ADVANTAGE:** Leverage Intel threat modeling experience (553+ threats) as major differentiator!

Labs & Exercises

- Create STRIDE threat model for fictional e-commerce system
- Reading: Threat Modeling: Designing for Security by Adam Shostack

Week 14: Advanced Threat Hunting

Core Topics

- MITRE ATT&CK framework: Tactics, techniques, procedures (TTPs)
- Hypothesis-driven hunting: Develop and test threat hypotheses
- Threat intelligence integration: IOCs, TTPs, threat actor profiling
- Behavioral analytics: Baseline normal behavior, detect anomalies

Labs & Exercises

- Map real-world attack to MITRE ATT&CK framework
- Develop threat hunting hypothesis and hunt in sample dataset

Phase 5: Cloud Security & Container Security (Weeks 15-16)

Parallel to AppSec Phase 5: Research & Discovery

Total Hours: 26 hours (13 hours/week)

Week 15: AWS Security & Cloud IAM

Core Topics

- AWS IAM: Users, groups, roles, policies, least privilege
- S3 security: Bucket policies, ACLs, encryption, versioning
- Lambda security: Execution roles, environment variables, VPC configuration
- API Gateway security: Authentication, authorization, rate limiting, WAF integration
- CloudTrail: Audit logging, monitoring API calls

Labs & Exercises

- Complete flAWS challenges (6 levels) - cloud security CTF
- Complete CloudGoat scenarios - vulnerable AWS infrastructure
- **Coding Challenge:** Build AWS security scanner checking S3 buckets, IAM policies

Week 16: Kubernetes & Container Security

Core Topics

- Kubernetes RBAC: Roles, ClusterRoles, bindings, service accounts
- Network policies: Pod-to-pod communication control
- Pod security: Security contexts, admission controllers, PodSecurityPolicy
- Docker security: Image scanning, secrets management, rootless containers
- Container runtime security: Seccomp, AppArmor, SELinux

Labs & Exercises

- Complete Kubernetes Goat scenarios - vulnerable K8s cluster
- **Coding Challenge:** Build Kubernetes RBAC auditor tool
- Reading: Container Security by Liz Rice

Phase 6: Security Coding Challenges (Weeks 17-20)

CRITICAL: This phase addresses #1 failure point in Security Engineer interviews

Parallel to AppSec Phase 6: Enterprise Production Tools

Total Hours: 52 hours (13 hours/week)

Week 17: Text Parsing & String Manipulation

Security Coding Challenges (from Grace Nolan)

- Cypher implementation: Caesar cipher, ROT13, Vigenère cipher
- String encoding/decoding: Base64, URL encoding, hex encoding
- Text obfuscation detection: Identify encoding schemes
- Pattern matching: Regex for security analysis

Deliverables

- Complete 10-12 string manipulation challenges
- LeetCode: 5-8 Easy/Medium string problems

Week 18: Log Parsing & IOC Extraction

Security Coding Challenges

- Log parser: Extract domains from mixed log formats
- IP address extraction: IPv4 and IPv6 from text
- Timestamp normalization: Convert various formats to standard
- Executable name extraction from process logs

Deliverables

- Build comprehensive log parser handling 5+ log formats
- IOC extraction tool: IPs, domains, file hashes, URLs
- LeetCode: 5-8 parsing/regex problems

Week 19: Web Scraping & Network Tools

Security Coding Challenges

- Web scraper: Extract security-relevant data from websites
- Port scanner: TCP/UDP with service detection
- Credential generator: Complex password generation with entropy calculation
- Login tracker: Track failed login attempts, detect brute force

Deliverables

- Build 3-4 security tools demonstrating network/web skills
- LeetCode: 5-8 problems involving data structures

Week 20: File Analysis & Binary Tools

Security Coding Challenges

- PDF metadata forensics tool: Extract creation date, author, software

- Malware signature detection: YARA-like pattern matching in binaries
- File hash calculator: MD5, SHA-1, SHA-256 for integrity verification
- PE file analyzer: Extract imports, exports, sections from Windows executables

Deliverables

- Build 3-4 file analysis tools
- LeetCode: 5-8 problems involving binary data, bit manipulation
- **PHASE GOAL:** 40-50 security-focused coding challenges + 20-30 LeetCode problems

Phase 7: Mock Interviews & System Design (Weeks 21-25)

Parallel to AppSec Weeks 21-25: DevSecOps, Tool Polish, Interview Prep

Total Hours: 65 hours (13 hours/week)

Week 21: Security System Design - Authentication & Authorization

System Design Topics

- Design secure authentication system: MFA, session management, password policies
- Design OAuth 2.0 / OpenID Connect implementation
- Design SSO system for enterprise

Practice & Deliverables

- Whiteboard 2-3 authentication system designs
- Document trade-offs, threat considerations, scalability

Week 22: Security System Design - Detection & Monitoring

System Design Topics

- Design SIEM architecture for enterprise (10K+ employees)
- Design anomaly detection system for network traffic
- Design incident response workflow and tooling

Practice & Deliverables

- Whiteboard 2-3 detection system designs
- Practice explaining log aggregation pipeline architecture

Week 23: Security System Design - Network & Cloud

System Design Topics

- Design network segmentation for cloud environment
- Design Zero Trust architecture
- Design secure multi-region cloud deployment

Practice & Deliverables

- Whiteboard 2-3 network architecture designs
- Document trust boundaries, data flows, security controls

Week 24: Mock Technical Interviews

Interview Practice

- Conduct 2-3 full mock interviews with peers or mentors
- Practice coding challenges under time pressure (45 mins)
- Record yourself explaining security concepts, review for clarity
- Practice whiteboarding threat models and architecture diagrams

Focus Areas

- Technical depth: Can you go 3-4 levels deep on any topic?
- Communication: Explaining complex concepts clearly and concisely
- Problem-solving: Thinking out loud, asking clarifying questions

Week 25: Behavioral Interviews & STAR Stories

STAR Method Preparation

- Prepare 5-7 STAR stories highlighting Intel threat modeling experience
- **Categories:** Conflict resolution, leadership, teamwork, ownership, adaptability, mistakes/learning
- Practice delivering stories naturally (not memorized word-for-word)

Company Research

- Research target companies: Mission, values, recent news, tech stack
- Prepare thoughtful questions for interviewers
- Review company-specific patterns (see Phase 8 for details)

Phase 8: Company-Specific Prep & Final Review (Weeks 26-30)

Parallel to AppSec Weeks 26-28: Active Job Applications & Networking

Total Hours: 70 hours (14 hours/week)

Week 26: Google Security Engineering Prep

Company-Specific Focus

- **Coding emphasis:** Expect LeetCode Medium level problems - practice 10-15 problems
- **Deep technical dives:** Networking, cryptography, system internals
- **Security system design:** Design SIEM, secure authentication, botnet detection
- **Googleyness:** Behavioral questions using Google's framework

Preparation Activities

- Review Grace Nolan's Google-specific interview notes thoroughly
- Practice explaining 'What happens when you visit google.com in browser?'

Week 27: Amazon & Microsoft Security Engineering Prep

Amazon Security Engineering

- **Leadership Principles:** Prepare stories for all 16 principles
- **Threat modeling:** Emphasis on systematic approach (your Intel advantage!)
- **AWS security:** Strong advantage but not always required
- **Coding:** Security-focused (log parsers, web crawlers)

Microsoft Security Engineering

- **Windows internals:** Deep knowledge highly valued (your Week 7 prep)
- **Cryptography:** TLS/authentication protocols depth
- **Enterprise architecture:** Large-scale system security
- **Threat modeling:** For Microsoft products/services

Week 28: Startup/Consulting Firms Prep (Trail of Bits, NCC Group)

Company-Specific Focus

- **Practical exploitation skills:** Emphasis on hands-on offensive security
- **Tool development:** Your AppSec curriculum aligns perfectly
- **Code review:** Vulnerability research methodology
- **Client communication:** Consulting mindset, explaining technical findings to non-technical audiences
- **Research mindset:** Novel vulnerability discovery, CVE publication

Preparation Activities

- Review your AppSec portfolio: Highlight tool development, blog posts
- Prepare to discuss P2P open source project vision
- Practice explaining complex vulnerabilities clearly

Week 29: Comprehensive Review - Networking & Cryptography

Spaced Repetition Review

- **Review ALL flashcards:** CIA triad, OSI layers, TCP/IP, DNS, TLS
- Practice verbal explanations: Explain each concept out loud without notes
- Cryptography refresh: AES modes, RSA, hash functions, common attacks
- Network troubleshooting: Practice systematic debugging scenarios

Weak Area Reinforcement

- Identify your 3-5 weakest topics from monthly reviews
- Deep dive into weak areas: Re-read sections, redo labs

Week 30: Final Review & Interview Readiness

Comprehensive Final Review

- **Critical memorization check:**
  - CIA Triad: Confidentiality, Integrity, Availability (+ AAA)
  - 5-Step Incident Response: Preparation, Identification, Containment, Eradication, Recovery (+ Lessons Learned)
  - OSI Model: All 7 layers with mnemonics
  - STRIDE: Spoofing, Tampering, Repudiation, Information Disclosure, DoS, Elevation
- **Final mock interviews:** 2-3 complete practice interviews
- **Coding practice:** 5-10 final security coding challenges
- **Mental preparation:** Review success factors, competitive advantages

You're Ready!

- 30 weeks of intensive preparation complete
- 360+ hours invested in Security Engineering mastery
- Unique combination: Intel threat modeling + AppSec skills + defensive security
- Trust your preparation - you've done the work!

Essential Resources

Free Resources (Start Here)

- Grace Nolan's Security Engineering Notes: github.com/gracenolan/Notes
- SANS Reading Room: Free security whitepapers on all topics
- NIST Cybersecurity Publications: SP 800 series
- High Performance Browser Networking: hpbn.co (free online)
- Beej's Guide to Network Programming: beej.us/guide/bgnet/
- CryptoHack: cryptohack.org - Interactive crypto challenges
- PicoCTF: picoctf.org - Beginner CTF challenges
- Pwnable.kr: Binary exploitation practice
- Malware Traffic Analysis: malware-traffic-analysis.net

Essential Books (~$500 Investment)

- Computer Networking: A Top-Down Approach by Kurose & Ross (~$80)
- The Linux Programming Interface by Michael Kerrisk (~$70)
- Serious Cryptography by Jean-Philippe Aumasson (~$40)
- Hacking: The Art of Exploitation by Jon Erickson (~$40)
- Practical Malware Analysis by Sikorski & Honig (~$50)
- The Practice of Network Security Monitoring by Richard Bejtlich (~$45)
- Threat Modeling: Designing for Security by Adam Shostack (~$50)
- Container Security by Liz Rice (~$40)

Hands-On Labs & Platforms

- TryHackMe: $10/month for guided learning paths
- HackTheBox: Free tier available, $14/month VIP
- LetsDefend: Blue team simulation platform
- Crackmes.one: Free reverse engineering challenges

Progress Tracking System

Weekly Review Template

6. **Concepts Mastered:** List 5-10 new terms/concepts you can explain clearly
7. **Labs Completed:** Document which hands-on exercises you finished
8. **Coding Challenges:** Track security coding problems solved
9. **Weak Areas:** Note topics requiring additional study
10. **Interview Readiness:** Rate 1-10 for each topic area
11. **Time Invested:** Log actual hours spent (goal: 10-14 hours/week)

Monthly Milestone Checkpoints

- **Month 1 (Weeks 1-4):** Networking fundamentals, TCP/IP, DNS, TLS, CIA triad mastery
- **Month 2 (Weeks 5-8):** Cryptography depth, binary exploitation, Windows security, log analysis
- **Month 3 (Weeks 9-12):** Email security, incident response, malware analysis, vulnerability management
- **Month 4 (Weeks 13-16):** Threat modeling, cloud security, container security
- **Months 5-6 (Weeks 17-24):** Security coding challenges, system design, mock interviews
- **Month 7-8 (Weeks 25-30):** Company-specific prep, final review, active interviewing

Final Advice & Success Factors

Critical Success Factors

12. **Consistency Over Intensity:** 12 hours every week beats 50 hours one week then nothing
13. **Active Learning:** Labs and coding beat passive reading every time
14. **Spaced Repetition:** Review concepts regularly, don't cram
15. **Teach to Learn:** Explain concepts to others or yourself to identify gaps
16. **Build in Public:** Document learning on GitHub and dev.to
17. **Community Engagement:** OWASP LA meetings and Null Space Labs provide accountability

Your Competitive Advantages

- **Enterprise Threat Modeling:** 553+ threats documented is rare and valuable
- **Systems Programming:** C/C++/Golang background differentiates from web-focused candidates
- **Cryptography Implementation:** XMSS, ChaCha20, Argon2 shows depth beyond typical candidates
- **Dual Curriculum:** Both offensive AND defensive skills - most candidates have only one
- **Systematic Approach:** This structured study demonstrates exceptional commitment

When You Feel Overwhelmed

- You don't need to master everything to get interviews - breadth over depth initially
- Your Intel experience already differentiates you significantly
- Scale back to 8-10 hours/week if needed - AppSec curriculum takes priority
- Take breaks when needed - burnout helps no one
- This is a marathon, not a sprint - trust the process

Key Reminders from Team Blind Analysis

The candidate who completed 85 interviews emphasized:

- **Success is 50% prep, 50% luck/personality fit**
- Younger interviewers may be stricter on trivia - know fundamentals cold
- Cultural fit matters - hiring managers liking you can overcome technical gaps
- Don't give up - 85 interviews shows persistence pays off

Good luck! You've got this.

Document Version: 2.0 Enhanced | *Generated: December 13, 2025*
For: Tanveer Salim (Fosres)
Sources: Grace Nolan's Notes | *Team Blind 85 Interviews* | *Complete 28-Week AppSec Curriculum*