

# ZERO TRUST A NIVEL DE SO

INTEGRANTES:

ESCOBAR DIAZ VICTOR MANUEL  
HERNANDEZ RUBIO JOSUE



# PUNTOS A TRATAR

01

## El problema

¿Por qué falla la seguridad tradicional?

02

## Zero Trust

Cambiando el paradigma de seguridad

03

## Implementación en SO

- Procesos y control de acceso
- Gestión de memoria
- IPC y comunicación
- Kernel y system calls

04

## Casos reales y ejemplos prácticos

05

## Beneficios y desafíos

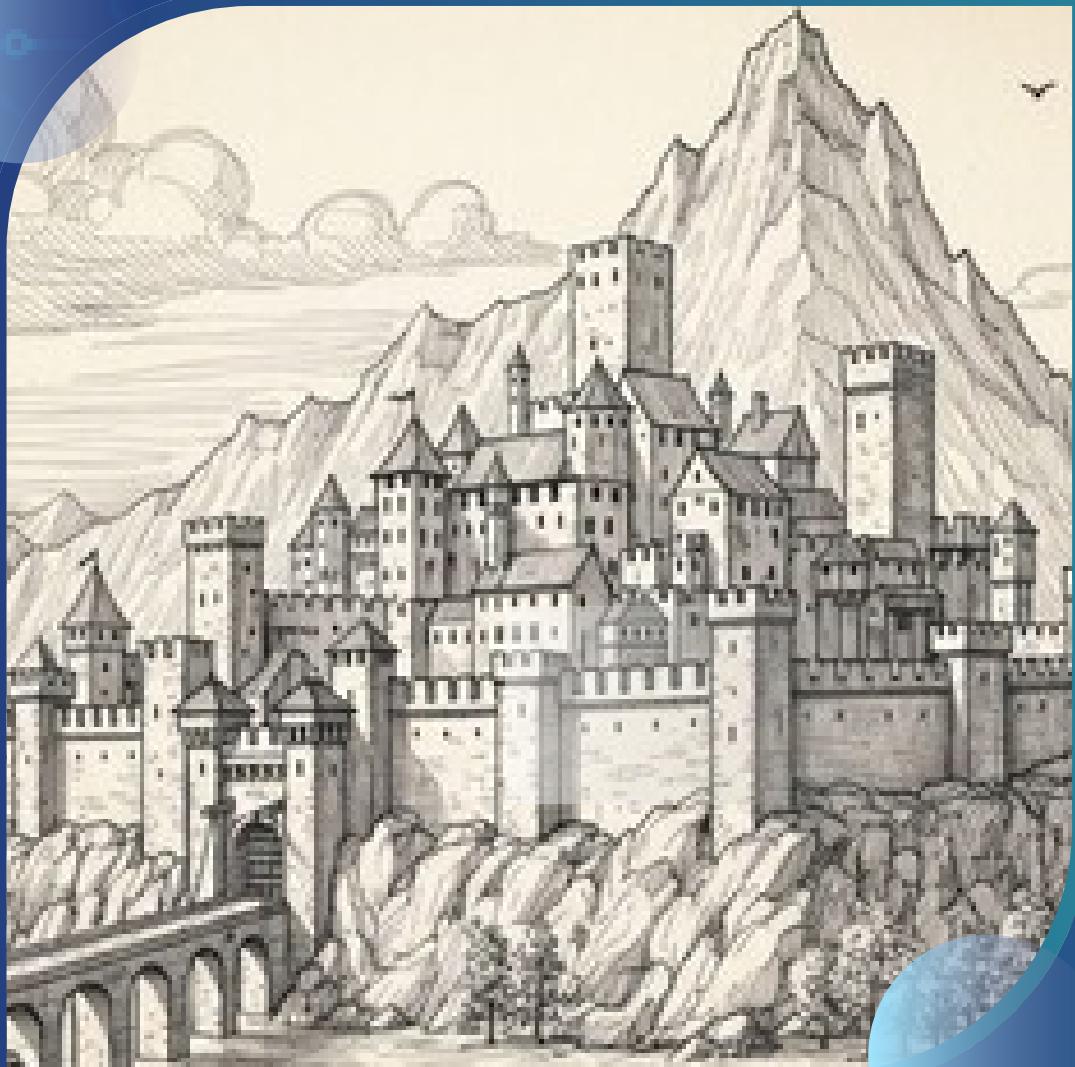
06

## Conclusiones



# EL MODELO DEL CASTILLO

## CONFIANZA IMPLÍCITA



"Una vez dentro del perímetro, confianza total"

- Firewall como muralla exterior
- Autenticación inicial = acceso completo
- Procesos legítimos pueden volverse maliciosos
- No hay verificación continua

PROBLEMAS:

- Ataques internos no detectados
- Escalamiento de privilegios
- Movimiento lateral de amenazas



# ZEROTRUST

"NUNCA CONFÍES, SIEMPRE VERIFICA"



## Tres principios fundamentales:

1. VERIFICACIÓN CONTEXTUAL CONTINUA
  - Cada acción se valida en tiempo real
  - No solo al inicio de sesión
2. PRIVILEGIO MÍNIMO ESTRICTO
  - Solo los permisos necesarios
  - Revocados inmediatamente
3. PRESUNCIÓN DE COMPROMISO
  - Asumir que cualquier componente puede estar comprometido



El Sistema Operativo es ideal porque controla:

- TODAS las llamadas al sistema
- CADA acceso a memoria
- TODA comunicación entre procesos
- TODOS los recursos del hardware

El kernel puede verificar identidades a nivel de:

- Procesos
- Memoria
- Dispositivos
- Archivos

# EL SO EL GUARDIAN PERFECTO



# PROCESOS VERIFICACIÓN CONTINUA

## Tradicional

- Proceso autenticado = acceso completo según UID/GID



## Trust zero

- Cada acceso a recurso se revalida
- Context-aware permission checking
- Análisis de comportamiento en tiempo real



Técnicas Zero Trust en memoria:



# MEMORIA

## DE AISLAMIENTO BÁSICO A PROTECCIÓN ACTIVA

### **W^X (Write XOR Execute) Estricto**

- Previene inyección de código malicioso

### **ASLR Mejorado**

- Randomización durante ejecución

### **Memory Tagging**

- Hardware para detectar accesos ilegítimos

### **Análisis de Heap/Stack**

- Detecta patrones sospechosos



# IPC DE COMUNICACIÓN ABIERTA A DIÁLOGO VERIFICADO

## Tradicional

- Procesos se comunican libremente
- Sin verificación de identidad
- Sin integridad de mensajes



## Trust zero

- Autenticación en memoria compartida
- Firmado digital de mensajes
- Cuotas estrictas de recursos
- Auditoría de comunicaciones sospechosas



# KERNEL

## SISTEMA INMUNOLÓGICO ACTIVO

### Técnicas de hardening:

- System Call Filtering (seccomp-bpf)
  - Restringe syscalls por proceso
- Kernel Module Signing
  - Solo módulos firmados cripticamente
- Privilege Escalation Monitoring
  - Auditoría continua de setuid/capabilities
- Control Flow Integrity
  - Protección contra code reuse attacks



# CASOS REALES

## DONDE ZERO TRUST PREVIENE ATAQUES



### RANSOMWARE

- Detecta cifrado masivo de archivos
- Bloquea procesos anómalos



### EXPLOITACIÓN DE VULNERABILIDADES

- Previene escalamiento de privilegios
- Bloquea code injection



### ATAQUES INTERNOS

- Detecta acceso a datos fuera de ámbito
- Auditoría de comportamientos sospechosos



### MALWARE AVANZADO

- Detecta patrones de comunicación anómalos
- Previene movimiento lateral



# IMPLEMENTACIÓN EN SISTEMAS MODERNOS

## LINUX

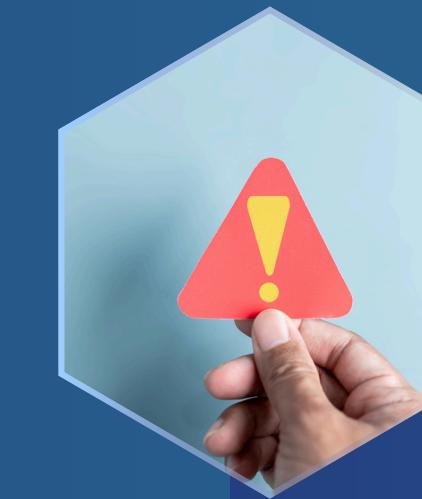
- \* SELinux (Security-Enhanced Linux)
- \* AppArmor
- \* Seccomp-BPF
- \* Integrity Measurement Architecture (IMA)

## WINDOWS

- \* Virtualization-Based Security (VBS)
- \* User Account Control (UAC)
- \* Windows Defender Application Guard
- \* Credential Guard



# DESAFÍOS Y CONSIDERACIONES



## DESAFÍOS TÉCNICOS:

- \* Complejidad de configuración
- \* Overhead de rendimiento
- \* Compatibilidad con aplicaciones legacy
- \* Mantenimiento de políticas

## BALANCE:

- \* Seguridad vs. Usabilidad
- \* Control vs. Flexibilidad
- \* Protección vs. Rendimiento

## RECOMENDACIÓN:

- \* Implementación gradual
- \* Políticas basadas en riesgos
- \* Monitoreo continuo



# CONCLUSIONES CLAVE

1. Zero Trust no es un producto, es una filosofía
2. El SO es la plataforma ideal para implementación
3. La verificación continua previene amenazas internas
4. El balance seguridad/rendimiento es manejable
5. El futuro está en la desconfianza inteligente



# GRACIAS POR LA ATENCION

## REFERENCIAS

Kindervag, J. (2010). Build Security Into Your Network's DNA: The Zero Trust Network Architecture. Forrester Research.

National Institute of Standards and Technology (NIST). (2020). SP 800-207: Zero Trust Architecture. U.S. Department of Commerce.

Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). Zero Trust Architecture. NIST Special Publication 800-207.

Saltzer, J. H., & Schroeder, M. D. (1975). The Protection of Information in Computer Systems. Proceedings of the IEEE, 63(9), 1278-1308.

Loscocco, P. A., & Smalley, S. D. (2001). Integrating Flexible Support for Security Policies into the Linux Operating System. NSA Technical Report.

McKusick, M. K., Neville-Neil, G. V., & Watson, R. N. M. (2014). The Design and Implementation of the FreeBSD Operating System. Addison-Wesley Professional.

