Frequencia 2

Aula 8: Linux Kernel Namespaces

Namespaces

Permite isolar e separar recursos do sistema para diferentes processos, de modo que cada conjunto de processos possa operar como se estivesse em um ambiente isolado. Esse conceito é fundamental para a criação de containers, como os usados em tecnologias de virtualização leve, por exemplo, Docker.

Tipos de Namespaces

1. **PID (Process ID) Namespace**: Isola os identificadores de processos. Cada namespace PID pode ter seu próprio conjunto de IDs de processos, o que permite que processos dentro de um container tenham IDs de processos que parecem começar do zero.

2. **NET (Network) Namespace**: Isola a pilha de rede, incluindo interfaces de rede, roteamento, regras de iptables, etc. Cada namespace de rede pode ter suas próprias interfaces de rede e configuração de roteamento, permitindo a criação de ambientes de rede isolados.

3. **MNT (Mount) Namespace**: Isola o sistema de arquivos montado. Permite que diferentes namespaces vejam diferentes conjuntos de sistemas de arquivos montados, permitindo, por exemplo, que um container tenha seu próprio sistema de arquivos independente do sistema de arquivos do host.

4. **UTS (UNIX Time-Sharing) Namespace**: Isola o hostname e o domain name do sistema. Cada namespace UTS pode ter seu próprio nome de host e domínio, o que é útil para renomear containers de maneira independente.

5. **IPC (Interprocess Communication) Namespace**: Isola recursos de comunicação entre processos, como filas de mensagens, semáforos e segmentos de memória compartilhada. Isso permite que containers tenham seus próprios recursos IPC sem interferência mútua.

6. **USER Namespace**: Isola IDs de usuários e grupos. Permite que processos dentro de um namespace vejam um conjunto diferente de IDs de usuários e grupos, o que é útil para rodar processos como root dentro de um container sem ter privilégios de root no host.

7. **CGROUP Namespace**: Isola a visão de control groups (`cgroups`), permitindo que processos dentro de um namespace vejam e manipulem apenas os cgroups atribuídos a eles.

A criação e manipulação de namespaces em Linux pode ser feita através de chamadas de sistema como `clone()`, `unshare()` e `setns()`, bem como utilizando ferramentas como `ip netns` para namespaces de rede ou `docker` para criar e gerir containers que utilizam várias dessas namespaces de forma transparente.

Aula 9: LXC (Linux Containers)

LXC (Linux Containers)

Tecnologia de virtualização a nível de sistema operacional que permite criar e gerir containers no Linux. Diferente das máquinas virtuais, que emulam hardware e executam um sistema operacional completo, os

containers LXC compartilham o mesmo kernel do host, proporcionando uma virtualização mais leve e eficiente.

Tipos de Containers

- **Privilegiados**: Os containers privilegiados (privileged) têm acesso total ao sistema host, permitindo que executem operações que normalmente seriam restritas, como montar sistemas de arquivos, criar dispositivos especiais, etc.
- **Não Privilegiados**: Os containers não privilegiados (non-privileged) executam com privilégios reduzidos, utilizando mapeamentos de UID (User ID) e GID (Group ID) para isolar os processos do container do sistema host.

Vantagens dos Containers LXC

- **Desempenho Superior**: Devido à ausência da sobrecarga de emulação de hardware, os containers LXC são mais rápidos e consomem menos recursos.
- **Facilidade de Uso**: Ferramentas e comandos intuitivos facilitam a criação e gestão de containers.
- **Flexibilidade**: Pode ser usado para diversos fins, como desenvolvimento, testes, ambientes de produção, etc.

Desvantagens dos Containers LXC

- **Segurança**: Embora o isolamento seja forte, ele não é tão completo quanto em máquinas virtuais. Vulnerabilidades no kernel podem afetar todos os containers.
- **Compatibilidade**: Todos os containers devem usar o mesmo kernel do host, limitando a diversidade de sistemas operacionais que podem ser executados simultaneamente.

Aula 10: AppArmor

AppArmor

É uma ferramenta de controlo de acesso obrigatório (MAC - Mandatory Access Control) para aplicações específicas. Ele permite que um administrador do sistema imponha limites de acesso detalhados para aplicações individuais usando perfis específicos. Esses perfis permitem a implementação de políticas de controlo de acesso minuciosas aos recursos do sistema em uma base por aplicação.

Aplicação

- **Kernel do Linux**: O AppArmor funciona no kernel do Linux como um Módulo de Segurança do Linux (LSM - Linux Security Module).
- **Framework do LSM**: Desde o kernel 2.6, o LSM fornece "hooks" para a inspeção de chamadas de sistema que estão prestes a fornecer acesso a objetos relevantes do sistema.

Benefícios do AppArmor

- **controlo de Objetos**: Permite controlar os objetos que uma aplicação pode usar, oferecendo um controlo mais detalhado do que o modelo tradicional baseado em identidades de usuários e capacidades.
- **Redução da Superfície de Ataque**: Minimiza a superfície de ataque das aplicações, garantindo que elas operem com o mínimo de privilégios necessários, seguindo o princípio do menor privilégio. Isso é

ideal para prevenir ataques de dia zero e comportamentos indesejados, como cavalos de Troia.
- **controlo de Exposição**: Permite que uma aplicação tenha múltiplas interfaces controladas, onde apenas as interfaces necessárias são exploradas por outras aplicações locais ou remotas.

Políticas de Aplicação

- **Sem Perfil**: Se não houver um perfil para um binário em execução, não há controlo.
- **Com Perfil**: Se houver um perfil para um binário em execução, os controlos de acesso definidos pelo perfil são aplicados.

Modos de Aplicação

- **Kill**: Violações de acesso terminam o processo.
- **Enforce**: Violações de acesso não são permitidas, e erros são retornados.
- **Complain**: Violações de acesso são apenas registadas, não impedindo a execução, permitindo ajuste iterativo dos perfis.

Auditoria e Logging

- **Violações de Acesso**: Podem ser registradas para auditoria posterior.
- **Melhoria Interativa de Perfis**: Violações registadas podem ser usadas para melhorar perfis interativamente com a ferramenta `aa-logprof`.

Perfis e controlo

- **Perfis Textuais**: São arquivos de texto que definem as políticas de segurança para aplicações.
- **Carregamento e Associação**: Perfis são carregados no kernel e associados a processos na execução de syscalls `exec`.
- **Modificações em Tempo de Execução**: Perfis podem ser modificados em tempo de execução, refletindo imediatamente nos processos associados.

Sintaxe dos Perfis

- **Variáveis e Inclusões**: Permitem a definição de variáveis e a inclusão de outros arquivos, proporcionando flexibilidade na definição de perfis.
- **Curingas de Nomes de Arquivos**: Usados para especificar grupos de arquivos ou diretórios de forma concisa.
- **Permissões de Arquivos**: Definem permissões específicas como leitura, escrita, mapeamento de memória, entre outras.

Controlo Abrangente

Os perfis do AppArmor podem controlar:

- **Acesso a Arquivos**: Leitura, escrita, execução, etc.
- **Execução de Binários**: controlo sobre quais binários podem ser executados.
- **Capacidades**: controlo sobre capacidades específicas do sistema.
- **Rede e Sockets**: controlo sobre operações de rede e sockets UNIX.
- **Montagem de Sistemas de Arquivos**: controlo sobre operações de montagem.
- **Sinais**: controlo sobre o envio e recebimento de sinais.

- **Dbus e ptrace**: controlo sobre comunicações Dbus e rastreamento de processos.
- **Rlimits**: controlo sobre limites de recursos.

Aula 11: TPM (Trusted Platform Module)

TPM

O Trusted Platform Module (TPM) é um microchip projetado para fornecer funções relacionadas à segurança, principalmente envolvendo a criptografia. É um componente essencial para a implementação de medidas de segurança avançadas em sistemas informáticos, neste caso, garantir a integridade e a segurança de um sistema.

Funcionalidades do TPM

- **Geração de Chaves Criptográficas**: O TPM pode gerar chaves criptográficas seguras, que são armazenadas no chip e não podem ser exportadas.
- **Armazenamento Seguro de Chaves**: As chaves geradas pelo TPM são armazenadas dentro do chip, protegidas contra acesso não autorizado.
- **Criptografia**: O TPM pode realizar operações criptográficas, como cifragem e decifragem de dados, assinaturas digitais, etc.
- **Autenticação de Plataforma**: O TPM pode ser usado para verificar se um dispositivo está autorizado a aceder a uma rede ou recurso. Isso é feito através de um processo chamado de *attestation* (certificação), onde o TPM fornece provas de que o hardware e o software não foram alterados.
- **Verificação de Integridade**: O TPM pode armazenar hashes de software e firmware críticos, permitindo a verificação da integridade desses componentes durante o arranque do sistema.
- **Timestamping**: O TPM pode associar data e hora a um conjunto de dados, fornecendo uma prova de que esses dados existiam em um determinado momento.

Evolução do TPM 1.2 para 2.0

- **Problemas com SHA-1**: O TPM 1.2 confiava fortemente no algoritmo SHA-1, que foi atacado com sucesso em 2005.
- **TPM 2.0**: Projetado para permitir algoritmos de digest alternativos e alternativas para todos os algoritmos criptográficos, além de introduzir criptografia simétrica para implementar cifras híbridas.

Conceitos Criptográficos

- **Hash Extends**: Usado para implementar PCRs (Platform Configuration Registers), criar logs de auditoria e políticas de autenticação.
- **Tickets**: Estruturas de dados contendo um HMAC calculado sobre alguns dados, permitindo que o TPM reconheça a informação posteriormente sem ter que armazená-la.
- **Cifras Simétricas**: Utilizadas para garantir a confidencialidade dos dados privados do TPM e das comunicações.

Modos de Cifra Simétrica

- **Modos de Bloco**: ECB, CBC (necessitam de preenchimento quando os dados não são múltiplos do tamanho do bloco).

- **Modos de Fluxo**: CFB, OFB, CTR (usados quando os dados não estão alinhados ao bloco).
- **controlo de Integridade Integrado**: HMAC-based Encrypt-then-MAC, com HMACs calculados com nonces para prevenção de replay.

Chaves de Endosso (EKs)

- **Chaves de Endosso**: Pares de chaves que identificam dispositivos TPM, certificadas pelo fabricante do TPM e utilizadas para certificar outras chaves TPM.

Casos de Uso do TPM

- **Identificação do Host**: Autorizar a participação em ambientes protegidos.
- **Criptografia de Dados do Host**: Proteção de arquivos e sistemas de arquivos.
- **Geração de Números Aleatórios**: Essencial para gerar suas próprias chaves e nonces.
- **NVRAM**: Armazenamento de dados críticos.
- **PCRs (Platform Configuration Registers)**: Usados para relatar sequências de medições e como sinais de autenticação.

Pilha de Software TPM (TSS)

- **TSS**: Padrão de software do TCG, onde as aplicações escritas para o TSS devem funcionar em qualquer sistema que implemente um TSS compatível.
- **Camadas do TSS**: Incluem Feature API (FAPI), Enhanced System API (ESAPI), System API (SAPI), TPM Command Transmission Interface (TCTI), TPM Access Broker (TAB) e Resource Manager (RM).

Entidades TPM

- **Permanentes**: Incluem índices não voláteis, objetos, e sessões de autorização por senha.
- **Não Persistentes**: Desaparecem no power-on e podem ser salvas fora do TPM, mas não recarregadas após um ciclo de energia.
- **Persistentes**: Objetos que o proprietário de uma hierarquia define para persistir entre ciclos de energia.

Hierarquias TPM

- **Hierarquia de Plataforma**: Controlada pelo fabricante da plataforma.
- **Hierarquia de Armazenamento**: Controlada pelo proprietário da plataforma.
- **Hierarquia de Endosso**: Controlada pelo utilizador da plataforma.

Gestão de Chaves

- **Geração e Exportação de Chaves**: Chaves podem ser geradas internamente, exportadas e importadas.
- **Árvores de Chaves (Hierarquias de Chaves)**: Sequência de chaves começando por uma chave primária.

Sessões TPM

Mantêm o estado entre sequências de comandos, permitindo a continuação de operações criptográficas e de controlo de acesso.

- **Variações de Sessões**:
  - **Bound/Unbound**: Sessões encontra-se ligada a um valor de autorização e computa a chave de sessão com base nesse valor.
  - **Salted/Unsalted**: Usam um valor aleatório para computar a chave de sessão (adiciona entropia).
- **Modificadores de Uso de Sessão**: Instruções por comando como continuar, cifrar, auditar.

Tipos de Sessões

- **Sessão de Senha**: Sessão de comando único dependente de uma senha fornecida em texto claro.
- **Sessão HMAC**: Sessão com HMAC computado com um valor authValue compartilhado.
- **Sessão de Política (Extended Authorization)**: Construída sobre a sessão HMAC e usa políticas para calcular um segredo compartilhado.

Vantagens do TPM

- **Segurança Avançada**: Protege contra roubo e manipulação de chaves criptográficas.
- **Confiança e Integridade**: Assegura que o hardware e o software do sistema não foram alterados.
- **Gestão de Chaves**: Simplifica a gestão e armazenamento de chaves criptográficas de forma segura.

Limitações do TPM

- **Dependência de Hardware**: Requer um chip TPM dedicado no hardware, o que pode limitar a sua disponibilidade.
- **Compatibilidade**: Nem todos os sistemas suportam ou são compativeis com o TPM, o que pode dificultar a sua implementação em alguns ambientes.

Aula 13: Bootstrap

AEGIS

- **Objetivo**: Garantir a integridade do bootstrap do sistema, assegurando que o kernel do sistema operativo é lançado por um processo de confiança.
- **Abordagem**: Construção de uma cadeia de verificações de integridade desde o momento de ligar o sistema até à transferência final de controlo para o sistema operativo. Verificações são feitas através da correspondência de hashes criptográficos calculados com assinaturas digitais armazenadas.

Cadeias de Confiança

- **Princípio**: A confiança é construída sobre medições. Se um código é medido e validado antes da execução, uma cadeia de confiança pode ser estabelecida. Cada componente mede e verifica o próximo antes de transferir o controlo.

Medições de Raiz de Confiança

- **SRTM (Static Root of Trust for Measurement)**: Parte do BIOS/UEFI e inicia a cadeia de confiança no momento da ligação do sistema.
- **DRTM (Dynamic Root of Trust for Measurement)**: Utiliza um Módulo de Código Autenticado (ACM) e requer um modo seguro especial da CPU (Intel TXT ou AMD SVM).

Trusted Computing Platform Alliance (TCPA)

- **Definição de Confiança**: Um componente ou processo é considerado confiável se o seu comportamento for previsível sob quase todas as condições operacionais e altamente resistente à subversão.
- **Atestado Remoto**: Permite que uma plataforma comprove a sua configuração atual a outra plataforma de uma maneira confiável.

TPM (Trusted Platform Module)

- **Funções Criptográficas**: Geração de números aleatórios, hashing, cifragem simétrica e assimétrica, geração e armazenamento seguro de chaves.
- **Tipos de TPM**:
  - **Discrete**: Implementado como um chip separado, altamente seguro e resistente a manipulações físicas.
  - **Integrated**: Integrado em um chip que fornece outras funções, ainda seguro, mas menos resistente a manipulações físicas.
  - **Firmware**: Implementado em software protegido, geralmente dentro de um TEE (Trusted Execution Environment).
  - **Software**: Implementado puramente em software, útil para protótipos, mas não recomendado para produção devido à falta de segurança.
  - **Virtual**: Fornecido por um hypervisor em ambientes de nuvem.

Registos de Configuração da Plataforma (PCR)

- **PCRs**: Registos que mantêm uma cadeia de hashes para verificar a integridade do sistema ao longo do tempo.
- **Tipos de PCR**: Divididos por função, como SRTM e DRTM, e usados para garantir que somente software autorizado seja executado.

Modos de Arranque da TCPA

- **Secure Boot**: O arranque é interrompido se um valor de PCR não corresponder a um valor esperado.
- **Authenticated Boot (Trusted Boot)**: Valores são armazenados nos registos PCR durante o arranque, permitindo verificações posteriores.

UEFI (Unified Extensible Firmware Interface)

- **Secure Boot**: Processo de validação do firmware, que define como o firmware da plataforma gerencia certificados de segurança e valida o firmware e o bootloader do sistema operativo.
- **Chaves do Secure Boot**:
  - **Platform Key (PK)**: Controlada pelo proprietário da plataforma.
  - **Key Exchange Key (KEK)**: Conjunto de chaves assimétricas controladas pelo fabricante e fornecedores de sistemas operativos.
- **Bases de Dados de Assinaturas**:
  - **db (whitelist)**: Contém chaves ou certificados que validam firmware autorizado.
  - **dbx (blacklist)**: Contém hashes de firmware não autorizado.

Recomendações de Segurança de Arranque da NSA

- **Modos de Arranque**:

    - **Legacy Boot**: Compatível com hardware antigo, menos restritivo em termos de segurança.
    - **UEFI Native Boot**: Oferece mais flexibilidade e modularidade, mas requer suporte de hardware e software modernos.
    - **Secure Boot**: Valida todas as partes do software durante o arranque, impedindo a execução de software não autorizado.
    - **TPM Auditing**: Regista hashes de componentes do firmware no TPM para verificações de integridade, mas não impede a execução de malware.

Aula 14: Smartcards

- **Definição e Componentes**:
    - **Smartcard**: Cartão com capacidades de processamento computacional, incluindo CPU, ROM, EEPROM e RAM. Pode ter interfaces com contato ou sem contato.
    - **Componentes Principais**:
        - **CPU**: Processador de 8/16 bits, com opção de co-processador criptográfico.
        - **ROM**: Armazena o sistema operativo, comunicação e algoritmos criptográficos.
        - **EEPROM**: Armazena o sistema de arquivos para programas/aplicações, chaves e senhas.
        - **RAM**: Armazena dados transitórios que são apagados quando o cartão é desligado.
        - **Segurança Física**: Inclui uma caixa à prova de violação e resistência a ataques de canal lateral.
- **Aplicações e Protocolos**:
    - **APDU (Application Protocol Data Unit)**: Usado para comunicação entre aplicações fora e dentro do cartão.
        - **T=0**: Cada byte é transmitido separadamente, sendo mais lento.
        - **T=1**: Transmissão em blocos de bytes, sendo mais rápido.
        - **ATR (Answer to Reset)**: Resposta do cartão a uma operação de reset, reportando o protocolo esperado pelo cartão.
- **Java Cards**: Smartcards que executam applets Java utilizando o Java Card Runtime Environment (JCRE), que roda sobre um sistema operacional nativo.
- **Sistemas de Arquivos**:
    - **Estrutura**: Composta por arquivos mestres (MF), arquivos dedicados (DF) e arquivos elementares (EF).
    - **Tipos de Arquivos**: Transparentes, de registos fixos, de registos variáveis e cíclicos.
    - **controlo de Acesso**: Podem ser configurados para exigir autenticação externa e proteção com MACs para acessar dados.