

Segurança Informática e nas Organizações

José Mendes 107188

2023/2024



universidade
de aveiro

1 Introdução

1.1 Segurança

Segurança - É o assunto focado na previsão de sistemas, processos, ambientes, ...
Ao longo de todos os aspetos do ciclo de vida de um sistema:

- Planeamento
- Desenvolvimento
- Execução
- Processos
- Pessoas
- Clientes e Supply Chain
- Mecanismos
- Standards e Regulamentos
- Propriedade Intelectual

1.1.1 Planeamento

Design de uma solução está de acordo com alguns requisitos dentro de um contexto normativo.

Sem flaws

- Todos os estados da operação são os previstos;
- Não há estados adicionais que fogem da lógica esperada (mesmo se transições forçadas são usadas);

Dentro do scope de um contexto normativo

- Especifico para cada atividade e setor (Ex: ISO 27001, ISO 27007, ISO 37001);

1.1.2 Desenvolvimento

Implementação de uma solução de acordo com o design, sem outros modos de operação.

Sem bugs a comprometer uma execução correta

- Sem crashes;
- Sem resultados invalidos ou inesperados;
- Com tempos de execução corretos;
- Com consumo de recursos adequado;
- Sem leaks de informação;

Software

- Requer uma implementação cuidadosa;
- Requer testes para obter uma implementação com os comportamentos esperados;

1.1.3 Execução

Código executa tal como foi escrito, com todos os processos previstos.

O ambiente é controlado, não pode ser manipulado ou observado.

Sem a existência de comportamentos anómalos, introduzido por aspetos ambientais
(como velocidade de armazenamento, quantidade de RAM, comunicação confiáveis)



1.1.4 Pessoas e Parceiros

O comportamento do Staff não pode ter um impacto negativo na solução.

- As normas existem para regular que ações são expectáveis;
- O Staff é treinado para distinguir comportamento correto de comportamento incorreto;
- O Staff tem os incentivos corretos para se comportar adequadamente;
- Quando o Staff é comprometido, ou se desvia, as ações têm impacto limitado;

1.1.5 Análise e Auditoria

Qual é o verdadeiro comportamento da solução?

Identificar desvios dos atributos esperados

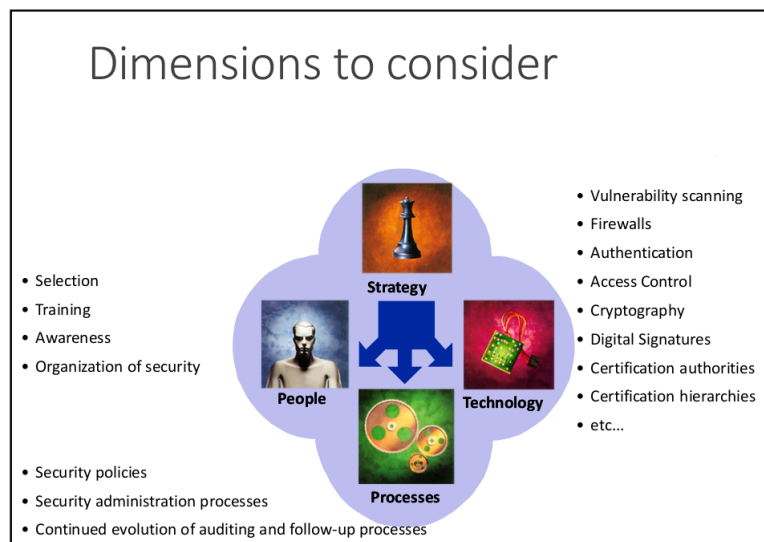
- Faults, erros, comportamentos

Identificar o risco para a solução ser modificada

- Exposição a possíveis atacantes;
- Incentivos que alguém possa ter para modificar a solução;
- Identificar potenciais actors (threats);

Identificar o impacto dos desvios

- Perda total de dados? Denial of Service? Increase Operation Cost?



1.2 Perspetivas

A SegurANÇA tem muitas perspetivas interligadas.

Defensive: Focado em manter previsão,

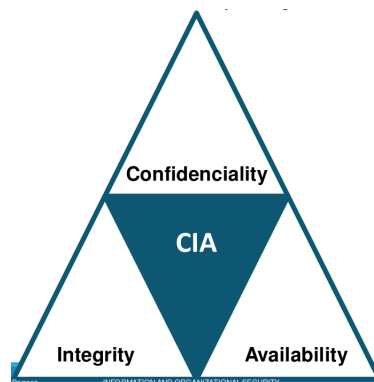
Offensive: Focado em explorar a previsibilidade.

- Pode ter uma intenção maliciosa/criminosa;
- Pode ter como objetivo, a validação da solução (Red Teams);

Outras:

- Engenharia Inversa: Recuperar o design de projetos contruídos;
- Forensics: Extrair informação e reconstruir eventos anteriores;
- Recuperação de Desastres: Minimizar o impacto de ataques;
- Auditoria: Avaliar se a solução está de acordo com um conjunto de requisitos;

1.3 Objetivos de Segurança de Informação



Confidencialidade: A informação pode apenas ser acessada por um grupo restrito de entidades;

Medidas:

- Encryptar informação;
- Usar passwords de acesso (fortes);
- Usar sistemas de gestão de identidade e autenticação;
- Doors, Strong Walls;
- Security personnel;
- Treinar (o Staff);

Integridade: A informação permanece inalterada (Pode ser aplicada ao comportamento de dispositivos e serviços);

Medidas:

- Controlo de identidade (hashes);
- Backups;
- Controlo de acesso;
- Dispositivos de armazenamento robustos;
- Processos de verificação de dados;

Disponibilidade: A informação está disponível a target entities (Pode ser aplicada aos serviços e dispositivos);

Medidas:

- Backups;
- Planos de recuperação de desastres;
- Redundância;
- Virtualização;
- Monitorização;

Privacidade: Como a informação pessoal é tratada (isto envolve: Obtida, Processada, Armazenada, Partilhada, Eliminada);

Medidas:

- Controlo de acesso;
- Processos transparentes;
- Ciphers;
- Integridade e controlo de autenticação;
- Logs;

1.4 Objetivos da Segurança

Defesa contra eventos catastróficos:

- Fenómenos naturais;
- Temperaturas extremas, inundações, trevoada, trovões, radiação, ...

Degradação do Hardware do computador:

- Falha no fornecimento de energia;
- Bad sectors em discos;
- Bit errors em células RAM ou SSD;

Defesa contra falhas normais:

- Queda de energia;
- Falhas internas do sistema;
 - Linux Kernel panic, Windows blue screen, OS X panic;
 - Deadlocks;
 - Uso anormal de recursos;
- Falhas de software / Falhas de comunicação;

Defesa contra atividades não autorizadas (adversários):

- Iniciado por alguém "de fora" ou "de dentro";

Tipos de atividades não autorizadas:

- Acesso a informação;
- Alteração de informação;
- Utilização de recursos (CPU, memory, print, network, ...);
- Denial of Service;
- Vandalismo (interferir com o funcionamento normal do sistema, sem obter benefícios);

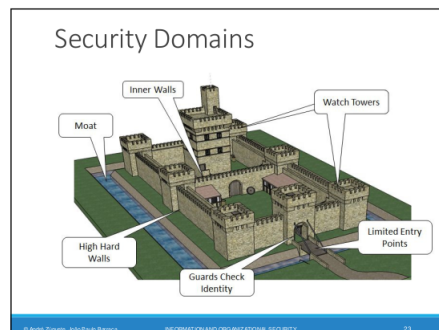
1.5 Conceitos Base

1. Domínios;
2. Políticas;
3. Mecanismos;
4. Controlos;

1.5.1 Domínios

Um conjunto de entidades que partilham atributos de segurança semelhantes.

- Permite gerir segurança de uma forma agregada;
 - A gestão define os atributos do domínio;
 - As entidades adicionadas ao domínio herdam os atributos do "grupo";
- Comportamento e interações são homogéneas dentro do domínio;
- Domínios podem ser organizados em hierarquias;
- As interações entre domínios são, normalmente, controladas;



1.5.2 Políticas

Conjunto de guidelines relacionados com a segurança, que mandam sobre o domínio.

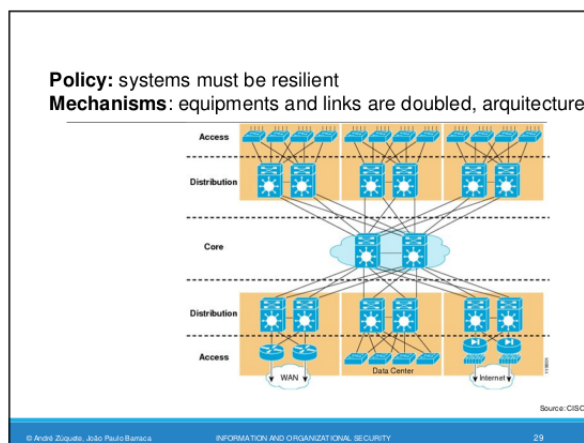
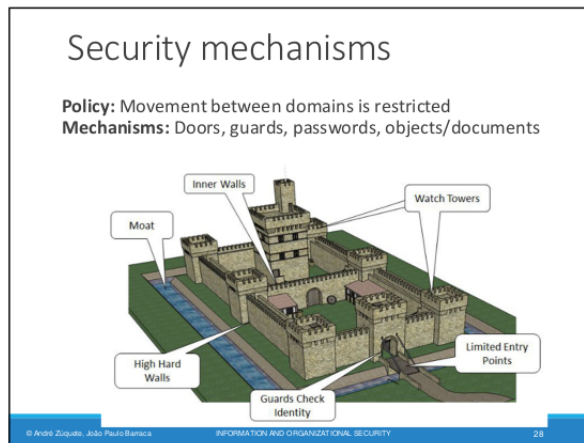
- Organizações têm múltiplas políticas;
 - Aplicáveis a cada domínio específico;
 - Podem dar overlap e terem scopes diferentes/níveis abstratos;
- As múltiplas políticas têm de ser coerentes;
- Exemplos:
 - Users apenas podem acessar serviços web;
 - Os assuntos devem ser autenticados para entrar no domínio;
 - Walls devem ser construídas de betão;
 - Comunicações devem ser encriptadas;
- Define o poder para cada assunto;
 - Least privilege principle: cada assunto apenas deve ter os privilégios necessários para executar as suas tarefas;
- Define procedimentos de segurança (quem faz o quê em que situação);
- Define os requisitos de segurança mínimos para um domínio;
 - Security levels, Security Groups
 - Autorização é necessária (and the related minimum authentication requirements (Strong/weak, single/multifactor, remote/face-to-face))
 - Define estratégias de defesa e táticas de contra-ataque;
 - * Arquitetura defensiva;
 - * Monitorização de atividades críticas ou sinais de ataque;
 - * Reação contra ataques ou outros cenários anormais;
 - Define que atividades são legais e ilegais;
 - * Forbid list model: Some activities are denied, the rest are allowed;
 - * Permit list model: Some activities are allowed, the rest is forbidden;

1.5.3 Mecanismos

- Implementam as políticas;
 - Definem, num nível mais elevado, o que precisa de ser feito ou evitado;
 - São usados para implementar políticas;

- Mecanismos de segurança genéricos:

- Confinamento (sandboxing);
- Autenticação;
- Controlo de acesso;
- Execução privilegiada;
- Filtragem;
- Logging;
- Auditoria;
- Algoritmos criptográficos;
- Protocolos criptográficos;



1.5.4 Controlos

Controlos são qualquer aspeto que permita minimizar o risco (proteger as propriedades **CIA**)

- Controlos incluem políticas e mecanismos, mas também:
 - Standards e regulamentos;
 - Processos;
 - Técnicas;
- Controlos são explicitamente definidos e podem ser auditáveis;
 - E.g.: ISO 27001 defines 114 controls in 14 groups (... asset management, physical security, incidente management...)

	Prevention	Detection	Correction
Physical	<ul style="list-style-type: none">- Fences- Gates- Locks	<ul style="list-style-type: none">- CCTV	<ul style="list-style-type: none">- Repair Locks- Repair Windows- Redeploy access cards
Technical	<ul style="list-style-type: none">- Firewall- Authentication- Antivirus	<ul style="list-style-type: none">- Intrusion Detection Systems- Alarms- Honeypots	<ul style="list-style-type: none">- Vulnerability patching- Reboot Systems- Redeploy VMs- Remove Virus
Administrative	<ul style="list-style-type: none">- Contractual clauses- Separation of Duties- Information Classification	<ul style="list-style-type: none">- Review Access Matrixes- Audits	<ul style="list-style-type: none">- Implement a business continuity plan- Implement an incident response plan

Horizontal: Relação ao evento
Vertical: Relação à sua natureza

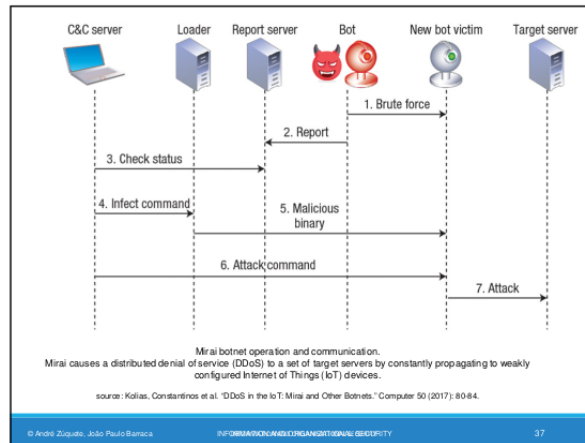
1.6 Segurança na Prática

Prevenção realista.

- Segurança perfeita é impossível;
- Focar nos eventos mais prováveis (pode depender de localização, legal framework, ...)
- Considerar o custo e o profit;
 - Um grande número de controlos tem um low cost;
 - No entanto, não limite superior para o custo de uma estratégia de segurança;
- Considerar todos os domínios e entidades;
 - Um simples breach pode escalar para um problema maior;
- Considerar impacto (Under the light of CIA and other potential impact areas (e.g., brand))
- Considerar o custo e o tempo de recuperação;
- Caracterizar atacantes (definir controlos específicos para esses, vão sempre existir atacantes com mais recursos);
- Considerar que o sistema será comprometido (Ter planos de recuperação);

1.7 Segurança em Sistemas Computacionais

- Computadores podem fazer grandes danos em pouco tempo;
 - Gerem grandes quantidades de informação;
 - Processam e comunicam com grande velocidade;
- O número de **weaknesses está sempre a aumentar**;
 - Devido a complexidade acrescida;
- As redes permitem mecanismos de ataque mais sofisticados;
 - Ataques anónimos de qualquer parte do mundo;
 - Espalha-se rapidamente através de barreira geográficas;
 - Exploitation of insecure hosts and applications
- Os atacantes constroem ataques em cadeia complexos;
 - First exploration
 - Lateral movement
 - Exfiltration



- A maior parte das vezes os users não sabem dos riscos
 - Não sabem os problemas, impacto, boas práticas nem as soluções;
- A maior parte das vezes os users são descuidados
 - Porque tomam riscos;
 - Não querem saber (não têm/identificam alguma responsabilidade);
 - Não estimam o risco corretamente;

1.8 Maiores fontes de vulnerabilidades

Aplicações hostis ou com bugs

- Rootkits: Insert elements in the operating system
- Worms: Software programs controlled by an attacker
- Virus: Pieces of code that infect other files (e.g., macros)

Users

- Ignorantes, descuidados, não querem saber
- Usam alternativas não seguras
- Confiam que as aplicações de segurança resolvem os problemas
- Download de software de fontes não confiáveis
- hostis

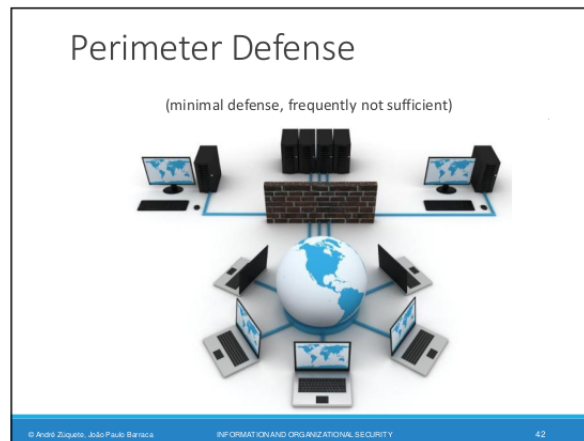
Administração defeituosa

- A configuração default é a mais segura
- Security restriction vs flexible operation
- Excessões a indivíduos

Comunicação através de redes desconhecidas/não controladas

- Public hotspots, campus networks, hostile governments

1.9 Perimeter Defense



Proteção contra atacantes externos

- Internet, Foreign users, outras organizações

Assume que os users internos são confiáveis e partilham as mesmas políticas

- Amigos, família, colaboradores

Usados em cenários domésticos ou em pequenas empresas

Limitações:

- Muito simples;
- Não protege contra ataques internos (users previamente confiáveis, atacantes que adquiriram acesso interno);

1.10 Defesa em Profundidade

Proteção contra atacantes externos e internos

- Da internet, de outras organizações, de users internos;

Assume domínios bem definidos pela organização

- Walls, doors, authentication, security personell, ciphers, secure networks

Limitações

- Precisa de coordenação entre os diferentes controlos (podemos acabar com controlos overlapping, mas também com "buracos" nos perímetros de segurança);

1.11 Zero Trust

Modelos de defesa sem perímetros específicos

- Não há confiança por herança nas entidades só por serem internas (na verdade, pode não haver noção de "interno" e "externo");

Modelo recomendado para novos sistemas

- Sistemas tradicionais deviam migrar para este modelo;
- Implies the design of systems/services specific for this model
- Legacy systems vão precisar de camadas de proteção adicionais (Firewalls, filtros, adaptadores, plugins)

1.11.1 Princípios (NCSC)

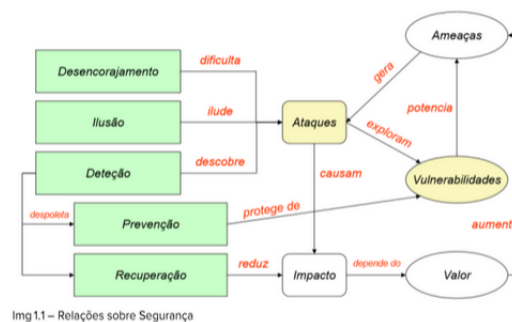
1. Saber a arquitetura (users, devices, services e data)
2. Saber as identidades (users, devices, services e data)
3. Avaliar o comportamento do user, service e saúde do device
4. Usar políticas para autorizar requests
5. Autenticar e autorizar em todo o lado (No open APIs, or IP address-based access)
6. Focar a Monitorização nos users, devices e services
7. Não confiar em nenhuma rede, incluindo a nossa (Os atacantes internos não devem ter mais privilégios que os externos)
8. Escolher services feitos para **zero trust** (evitar legacy services, mas podem ser integrados)

2 Vulnerabilidades

Uma empresa é tão mais suscetível de ataques quanto maior a sua dimensão, uma vez que ataques bem sucedidos serão mais rentáveis

De forma a prevenir **ataques**, que exploram **vulnerabilidades**, as organizações devem investir na **defesa** dos seus sistemas, de forma a garantir a segurança da informação que armazenam.

2.1 Segurança de Informação



Img 1.1 – Relações sobre Segurança

2.1.1 Medidas (e algumas ferramentas)

No entanto, **defesa** é um conceito abstrato, que na realidade ganha forma em cinco medidas.

Desencorajamento: através da punição dos infratores (restrições legais e forensic evidences) e utilização de barreiras de segurança (firewalls, Autenticação, Sandboxing, ...)

Detecção: sistema de deteção de intrusões (e.g Seek, Bro, Suricata), ou através de auditorias e análises forenses;

Ilusão: dos atacantes com honeypots ou honeynets (como que pishing para atacantes) e follow-up com análise forense;

Prevenção: através de políticas de segurança (e.g least priviledge principle), deteção (e.g OpensVas, metasploit) e correção de vulnerabilidades (e.g updates regulares);

Recuperação: com backups, sistemas redundantes, recuperação forense;

2.2 Vulnerabilidade

É um erro no software que pode ser diretamente usado por um atacante para ganhar acesso a um sistema ou rede.

Um erro é uma vulnerabilidade se permitir a um atacante usá-lo para violar uma política de segurança para esse sistema.

Isto exclui políticas de segurança completamente "abertas" em que todos os users são confiáveis, ou onde não há consideração do risco do sistema.

Uma vulnerabilidade **CVE** é um estado num sistema computacionais (ou conjunto de sistemas) que podem:

- Permitir ao atacante executar comandos como outro user;
- Permitir ao atacante aceder a dados que é contrário às restrições de acesso especificadas para esses dados;
- Permitir a um atacante fingir ser outra entidade;
- Permitir ao atacante realizar denial of service;

2.3 Exposição

Problema de configuração que permite ao atacante aceder a informação ou capacidades que o podem auxiliar, sem conseguir no entanto comprometer diretamente o sistema.

Um problema de configuração ou um erro é uma exposição se não permitir diretamente comprometer a segurança do sistema, mas pode ser um componente importante para a realização de um ataque bem sucedido, e é uma violação de uma política de segurança.

Uma exposição descreve um estado no sistema computacional (ou conjunto de sistemas) que não é uma vulnerabilidade mas pode:

- Permitir a um atacante conduzir atividades para obter informação;
- Permitir a um atacante esconder atividades;
- Inclui uma capacidade que se comporta como esperado, mas pode ser facilmente abusada;
- É o ponto primário de entrada em que um atacante pode tentar usar para ganhar acesso ao sistema ou aos dados;
- É considerado um problema por algumas políticas de segurança;

2.4 CVE - Common Vulnerabilities and Exposures

É um repositório público de vulnerabilidades, que lista e descreve vulnerabilidades e exposições de segurança.

Dicionário de vulnerabilidades e exposições sobre segurança de informação

- Para gestão de vulnerabilidades;
- Para gestão de resolução de problemas;
- Para alertar sobre novas vulnerabilidades;
- Para deteção de intrusões;

Usa identificadores comuns para os mesmos CVEs

- Permite a partilha de dados entre produtos de segurança;
- Oferece um baseline index point para avaliar coverage of tools and services;

Detalhes sobre uma vulnerabilidade podem ser mantidos privados

- Parte da divulgação responsável: até que o proprietário forneça uma solução;

(Ver imagem no slide 4)

2.4.1 Identificadores CVE

Aka CVE names, CVE numbers, CVE-IDs, CVEs

Identificador único e comum para vulnerabilidades de segurança de informação publicamente conhecidas

- Têm status "candidate" ou "entry";
- Candidato: Em review para inclusão na lista;
- Entry: Aceite na lista CVE;

Formato

- Numero identificador CVE (CVE-Year-Order);
- Status (candidate, entry);
- Descrição curta da vulnerabilidade ou exposição;
- Referências a fontes de informação;

2.4.2 Benefícios do CVE

Fornece uma linguagem comum para os problemas referenciados

- Facilita a partilha de dados entre ferramentas e serviços;
- Sistemas de deteção de intrusões;
- Ferramentas de acesso;
- Bases de dados de vulnerabilidades;
- Researchers;
- Equipes de resposta a incidentes;

Vai liderar para melhorar as ferramentas de segurança (mais compreensivo, melhores comparações, interoperabilidade)

Vai originar mais inovação (Ponto focal para discutir questões críticas de conteúdo de banco de dados)

2.4.3 CVE e ataques

Ataques são tornados possíveis através de múltiplas vulnerabilidades (um CVE para cada vulnerabilidade)

2.5 Detecção de Vulnerabilidades

Ferramentas específicas podem ser usadas para detetar vulnerabilidades

Estas exploram vulnerabilidades conhecidas, testando padrões (e.g buffer overflow, SQL injection, XSS, ...)

Ferramentas específicas podem replicar ataques conhecidos

Usar exploits conhecidos para vulnerabilidades conhecidas. Podem ser usadas para implementar medidas de defesa.

Vital para certificar a robustez de um sistema de produção e aplicações

Serviço muitas vezes oferecido por empresas externas.

Pode ser aplicado a:

- Source code;
- Aplicações em execução (análise dinâmica);
- Externamente como um cliente remoto;

Não dever ser aplicado cegamente a sistemas de produção

Potencial perda de dados/corrupção, DoS, atividade ilegal, ...

2.6 CWE - Common Weakness Enumeration

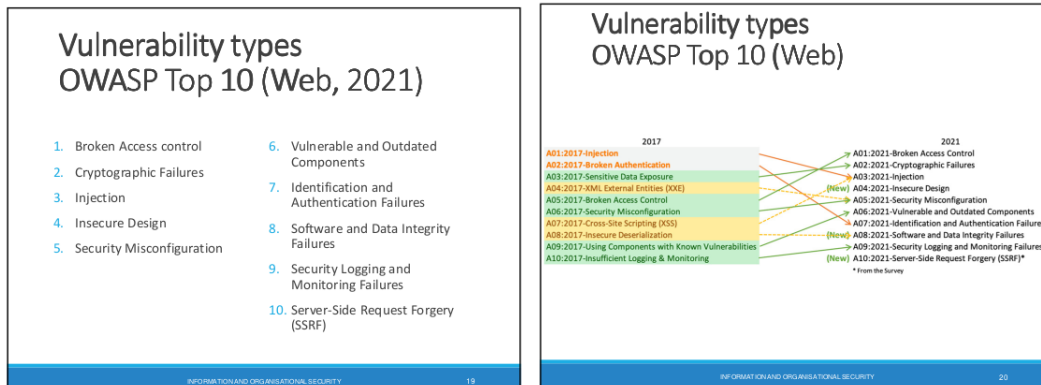
De forma complementar temos outro repositório, mas focado na exploração das causas das vulnerabilidades, ou seja, identifica as vulnerabilidades provocadas pelos developers devido a uma utilização incorreta do software.

São encontradas no código, design, arquitetura do sistema. Cada CWE representa um único tipo de vulnerabilidade. É mantido pelo MITRE e esta lista fornece detalhes para cada CWE.

Um CWE podem organizar-se de forma hierárquica, havendo um pai que fornece uma descrição genérica e vários filhos, cada um focado numa parte concreta do problema.

Níveis mais profundos de CWEs, oferecem mais granularidade, normalmente com menos filhos, ou sem filhos.

CWE \neq CVE



2.7 Rastreamento de Vulnerabilidades por parte dos vendedores

Durante o ciclo de desenvolvimento, as vulnerabilidades são tratadas como bugs, pode existir uma equipa de segurança ou não. Quando o software está disponível, as vulnerabilidades também são rastreadas globalmente, para cada sistema e software disponível ao público.

O rastreamento público ajuda a:

- Focar a discussão à volta do problema;
- Aos defensores a facilmente testar o sistema, aumentando a segurança;
- Aos atacantes a facilmente saberem quais as vulnerabilidades a explorar;

As vulnerabilidades são rastreadas de forma privada (consitui um arsenal para ataques futuros contra alvos)

O conhecimento sobre vulnerabilidades é publicamente disponível e pode ser trocado por dinheiro. Mas também pode ser trocado de forma privada por ainda mais dinheiro.

2.8 Rastreamento de Vulnerabilidades

Não é algo fácil de fazer, uma vez que os exploits não são sempre conhecidos, o impacto e o custo podem ser difíceis de estimar (underestimated).

Feeds anteriores podem criar um falso sentido de segurança.

Possuir uma **comunicação dinâmica** é bom:

- Para os defensores, pois eles podem testar e implementar defesas;
- Para atacantes, pois estes podem incorporar os exploits;

2.9 Ataques de dia zero

Aka Zero Day (or Zero Hour) Attacks/Threat.

Este tipo de ataque caracteriza-se por explorar uma vulnerabilidade desconhecida. Este ocorre no dia zero do conhecimento da vulnerabilidade, para a qual não existe um security fix.

Se for explorada de forma discreta, pode durar meses ou até anos, conhecido por atacantes e não pelos outros, frequentemente parte do arsenal de ataque, sendo inclusive comercializadas em certos mercados (negro).

2.10 Sobrevivência

Como sobreviver a um ataque Dia Zero? Como podemos reagir a um destes ataques?

Apesar de ser o oposto do que geralmente é esperado dos sistemas (estandardização, protocolos bem definidos e regulares), a **diversidade** é a chave para a sobrevivência.

Isto porque dada a sua exclusividade, operações e protocolos distintos são mais difíceis de contornar, uma vez que requerem um estudo dedicado do sistema em particular e não podem ser aplicados de forma generalizada a outros.

Dada a sua diversidade, o SO Android terá menos probabilidade de ser atacado que o iOS.

2.11 CERT - Computer Emergency Readiness Team

Esta é uma equipa responsável por resistir a ataques em sistemas distribuídos (em rede), limitando o dano e garantindo a continuidade dos serviços críticos.

CERT/CC (Coordination Center) @ CMU

Um componente de um maior programa CERT, é um centro importante para problemas de segurança na internet.

2.12 CSIRT - Computer Security Incident Response Team

Dentro das equipas CERT, há uma componente de sigla CSIRT, cuja responsabilidade é receber, analisar e responder a relatórios de incidente e atividade.

2.13 Alertas de Segurança e activity trends

Vital para a disseminação rápida de conhecimento sobre novas vulnerabilidades (e.g US-CERT Cyber Security Alerts, SANS Internet Storm Center, Cisco Security Center, ...)

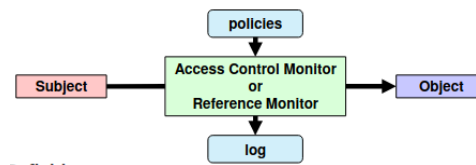
3 Modelos de Controlo de Acesso

3.1 Tipos de Acesso

Acesso Físico: Contacto físico entre o sujeito e o objeto de interesse (e.g. acesso a um edifício, internet, computador, aparelho, token ...)

Acesso Informático ou Eletrónico: Contacto orientado à informação entre o sujeito e o objeto de interesse, isto é, contacto através de diálogos request-response. Este contacto é mediado por computadores, redes, sistemas operativos, aplicações, middleware, ...

3.2 Controlo de Acesso



Políticas e mecanismos que mediam o acesso do sujeito a um objeto.

Requisitos Normais (AAA):

- Autenticação (com algum Level of Assurance (LoA))
- Políticas de Autorização
- Accountability (Auditoria) → logging

Sujeitos e objetos são os dois entidades digitais.

Sujeitos: Algo exibindo atividade (e.g. processos, computadores, redes)

Objetos: O alvo da ação (e.g. dados armazenados, tempo de CPU, memória, processos, computadores, redes)

Nota: Uma entidade pode ser um sujeito e um objeto ao mesmo tempo.

3.3 Princípio do Menor Privilégio aka Least Privilege Principle

Todos os programas e todos os utilizadores do sistema devem operar usando o menor conjunto de privilégios necessários para completar o trabalho.

Privilégios: Autorização para executar um dado trabalho, parecido com access control clearance.

Cada sujeito deve ter, em todo o momento, o número de privilégios exato necessário para completar os trabalhos dados. Menos privilégios criam barreiras intrespassáveis, enquanto que mais privilégios criam Vulnerabilidades (dano causado por acidentes ou erros, má utilização de privilégios, ...)

3.4 Modelos de Controlo de Acesso

Access control models

	O1	O2	...	Om-1	Om
S1		Access rights			
S2					
...					
Sn-1					
Sn					

Access control matrix

- Matrix with all access rights for subjects relatively to objects
- Conceptual organization

Mecanismos ACL-based: ACL: Access Control List, coluna da matrix

Lista de direitos de acesso para sujeitos específicos: Os direitos de acesso podem ser positivos e negativos, sujeitos base podem ser usados normalmente.

Normalmente, ACLs estão guardados com os objetos

Mecanismos Capability-based: Capability: token de autorização impossível de falsificar, linha da matrix, contém referências a objetos e direitos de acesso.

Conceder acesso: Transmissão de capabilities entre sujeitos (mediado/não mediado)

Normalmente, capabilities estão guardados com os sujeitos

3.4.1 Tipos de Controlo de Acesso: MAC e DAC

Mandatory Access Control (MAC):

- Política de controlo de acesso é fixa e implementada pelo monitor de controlo de acesso;
- Os direitos de controlo de acesso não podem ser adaptados pelos sujeitos ou pelos donos dos objetos;

Discretionary Access Control (DAC):

- Alguns sujeitos podem alterar os direitos dados ou negados a outros sujeitos para um dado objeto;
- Normalmente, isto é dado aos donos dos objetos e aos administradores do sistema;

3.4.2 Tipos de Controlo de Acesso: Role-Based Access Control (RBAC)

Não é nem MAC nem DAC

- Os roles são dinamicamente atribuídos aos sujeitos;
- Para controlo de acesso, o role importa usado pelo sujeito e não a identidade do sujeito (a identidade é mais relevante para acesso a roles e logging);

Controlo de Acesso vincula funções a operações (significativas)

- Operações são transações de sistema complexas e significativas;
- Operações podem envolver múltiplos objetos individuais lower-level;

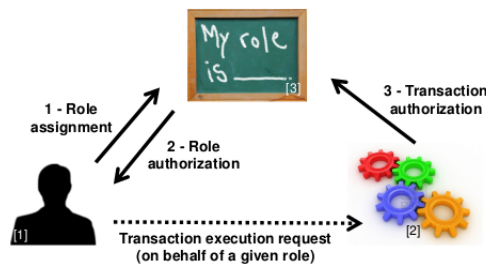
3.4.3 Regras RBAC

Atribuição de roles:

- Toda a atividade do sujeito num sistema é conduzida através de transações. As transações são permitidas para roles específicos, logo, todos os sujeitos têm de ter algum role ativo;
- Um sujeito pode executar uma transação **apenas se** tiver selecionado/tiver sido atribuído o role que permite a execução dessa transação;

Autorização de role: O role ativo de um sujeito tem de ser autorizado para esse sujeito;

Autorização de transação: Um sujeito pode executar uma transação **apenas se** essa transação for autorizada através dos role membership's do sujeito e se não houver nenhuma limitação que possa ser aplicada sobre sujeitos, roles e permissões.



3.4.4 RBAC: Roles e Groups

Roles: São uma coleção de permissões que são atribuídas aos sujeitos, que num determinado instante têm esse role. Um sujeito pode (deve) apenas ter um role ativo de cada vez.

Groups: São conjuntos de users, e as permissões podem ser atribuídas a ambos, users e groups. Um sujeito pode pertencer a vários grupos ao mesmo tempo.

O conceito de sessão: A atribuição de um role é tipo a ativação de uma sessão. O group membership é um atributo estático ordinário.

RBAC variants

RBAC 0

- No role hierarchies
- No role constraints

RBAC 1

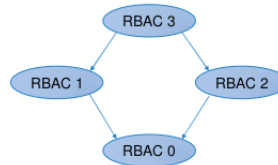
- RBAC 0 w/ role hierarchies (privilege inheritance)

RBAC 2

- RBAC 0 w/ role constraints (separation of duties)

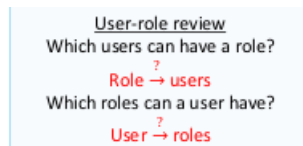
RBAC 3

- RBAC 1 + RBAC 2



3.4.5 Modelo NIST RBAC

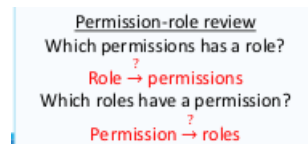
Flat RBAC: Simples modelo RBAC, com **user-role review**



Hierarchical RBAC: Flat RBAC com **role hierarchies** (DAG ou árvore). Hierarquias gerais e restritas.

Constrained RBAC: RBAC com **role constraints** para separar deveres.

Symmetric RBAC: RBAC com **permission-role review**



3.4.6 Tipos de Controlo de Acesso: Context-Based Access Control (CBAC)

Os direitos de acesso têm um contexto histórico, não podem ser determinados sem considerar operações de acesso anteriores.

Chinese Wall Policy: Grupos conflitantes, políticas de controlo de acesso devem considerar acessos passados a objetos em diferentes membros de grupos conflitantes.

3.4.7 Tipos de Controlo de Acesso: Attribute-Based Access Control (ABAC)

Decisões de controlo de acesso são baseadas em atributos associados a entidades relevantes

Arquitetura OASIS XACML:

- Policy Administration Point (PAP), onde as políticas são geridas
- Policy Decision Point (PDP), onde as decisões de autorização são avaliadas e tomadas
- Policy Enforcement Point (PEP), onde os pedidos de acesso são interceptados e confrontados com decisões PDP
- Policy Information Point (PIP), onde o PDP obtém informação externa

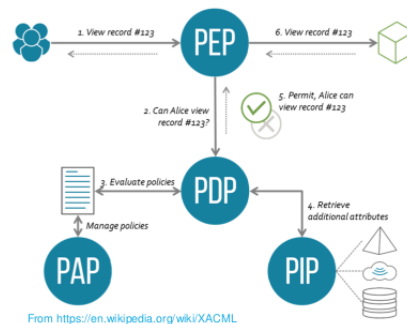
3.4.8 XACML: Controlo de Acesso com PEP e PDP

Um sujeito realiza um pedido, interceptado pelo PEP, que envia o pedido de autorização para o PDP.

O PDP avalia o pedido contra as suas políticas e chega a uma decisão, que é retornada pelo PEP.

- As políticas são devolvidas por um Policy Retrieval Point (PRP)
- Atributos úteis são obtidos por um Policy Information Point (PIP)
- As políticas são geridas pelo Policy Administration Point (PAP)

XACML big picture



3.5 Modelos de Controlo de Acesso: Break-the-Glass

Em alguns cenários, pode ser necessário ultrapassar os limites de acesso estabelecidos (e.g questão de vida ou morte).

Nestes casos, o sujeito pode usar uma decisão break-the-glass sobre a recusa de acesso. Ultrapassa a recusa com a própria responsabilidade, logging é fundamental para prevenir abusos.

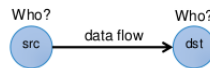
3.6 Separação de Deveres

Requisito fundamental de segurança para prevenção de fraude e erro. Disseminação de tarefas e privilégios associados, para um negócio específico, por múltiplos sujeitos. Muitas vezes implementado com RBAC.

Controlo de Danos. Segregação de deveres ajuda a reduzir os potenciais danos das ações de uma pessoa. Alguns deveres não devem ser combinados numa única posição.

3.7 Modelos do flow de informação

Autorização é aplicada ao flow dos dados



Objetivo: evitar flows de informação que não queremos/perigosos

Atributos de segurança Src e Dst, o flow apenas deve acontecer entre duas entidades com os atributos de segurança apropriados. A autorização é baseada nos atributos de segurança (SL).

3.8 Segurança Multinível

Sujeitos (ou roles) atuam em diferentes níveis de segurança. Este níveis não se interseção a si próprios, e possuem uma ordem parcial (hierarquia, lattice).



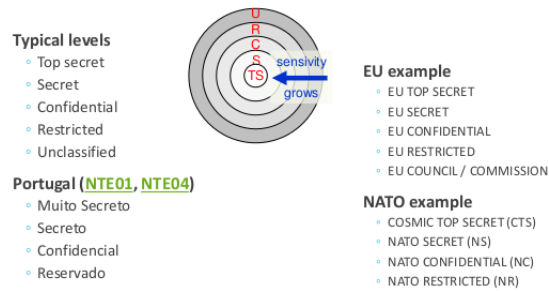
Os níveis são usados como atributos dos sujeito e dos objetos.

- **Sujeitos:** clearance a nível de segurança;
- **Objetos:** classification a nível de segurança;

Flows de informação e níveis de segurança

- Mesmo nível de segurança: atutorizado;
- Nível de segurança diferente: controlado;

Multilevel security levels: Military / Intelligence organizations



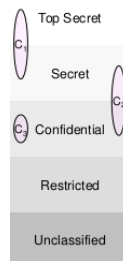
3.9 Categorias de segurança (ou compartimentos)

Ambientes de self-contained information, podem abranger vários níveis de segurança.

Ambientes militar, ramos militares, unidades militares

Ambientes civís, departamentos, unidades de organização

Um objeto pode pertencer a diferentes compartimentos e ter diferentes classificações de segurança para cada um ((top-secret, crypto), (secret, weapon))



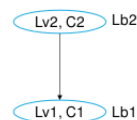
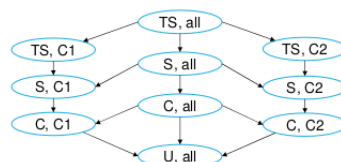
3.10 Labels de segurança

Label = Category + Level

Relative order between labels

$$Lb1 \leq Lb2 \Rightarrow C1 \leq C2 \wedge Lv1 \leq Lv2$$

Labels form a lattice



3.11 Modelos MLS Bell-La Padula

Política de controlo de acesso para controlar flows de informação. Aborda a confidencialidade dos dados e o acesso a informação classificada. Aborda a divulgação de informação classificada (controlo de acesso dos objetos não é suficiente).

Usa um modelo de transação de estados. Em cada estado há sujeitos, objetos, uma matrix de acesso e a informação atual de acesso. Regras de transação de estados.

Simple security condition (no read up)

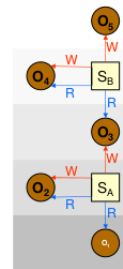
- S can read O iff $L(S) \geq L(O)$

*-property (no write down)

- S can write O iff $L(S) \leq L(O)$
- aka confinement property

Discretionary Security Property

- DAC-based access control



3.12 Modelo de Integridade Biba

Política de controlo de acesso para controlar flows de informação.

- Para reforçar o controlo da integridade dos dados;
- Usa níveis de integridade, não níveis de segurança;
- Parecido com **Bell-La Padula** com regras invertidas;

Simple Integrity Property (no read down)

- S can read O iff $I(S) \leq I(O)$

Integrity *-Property (no write up)

- S can write O iff $I(S) \geq I(O)$



3.13 Controlo de Integridade obrigatório Windows

Permite controlo de acesso obrigatório antes de avaliar DACLs

- Se não for permitido, DACLs não são avaliadas;
- Se for permitido, DACLs são avaliadas;

Labels de integridade

- Não confiável;
- Baixo (ou AppContainer);
- Médio;
- Médio Plus;
- Alto;
- Sistema;
- Processo Protegido;

Users

- Médios: users normais;
- Altos: users elevados;

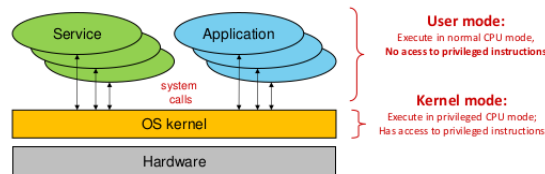
Processos de nível de integridade

- O mínimo associado ao owner e ao ficheiro executável;
- Processos de um user normalmente são **médio** ou **alto** (exceto ao executar ficheiros executáveis **Low**-labeled);
- Processos de serviço: **alto**;

Securable objects mandatory label

- **NO_WRITE_UP** (default)
- **NO_READ_UP**
- **NO_EXECUTE_UP**

4 Sistemas Operativos



4.1 Objetivos do Kernel

- Inicia os dispositivos (boot time);
- Virtualiza o hardware, modelo computacional;
- Reforça políticas de proteção e fornece mecanismos de proteção. Contra erros involuntários e atividades não autorizadas;
- Fornece o sistema de ficheiros, independente dos dispositivos de armazenamento usados;

4.2 Anéis de Execução

Diferentes níveis de privilégio, formando um conjunto de anéis. Usado pelos CPUs para prevenir código não privilegiado de correr opcodes privilegiados.

Hoje em dia, os processadores têm 4 anéis, mas apenas 2 são usados pelo OS, o 0 e o 3. O 0 é o mais privilegiado (supervisor/kernel mode) e o 3 o menos privilegiado (user-mode).

Transferência de controlo entre anéis requer gates especiais, os usados por system calls, syscalls (traps) e interrupções (interrupt gates).

4.3 Executar Máquinas Virtuais

Técnica comum: Virtualização baseada em software, com execução direta de código em user-mode (ring 3). Tradução binária de código privilegiado, os kernels do OS permanecem inalterados, mas não correm diretamente na host machine.

Virtualização assistida por hardware: Virtualização completa, pelo que existe um anel -1 por baixo do anel 0, que é usado pelo hypervisor. Esta forma pode virtualizar hardware para muitos anéis kernel 0. Não há necessidade de traduções binárias, o OS é mais rápido (quase performance nativa).

As máquinas virtuais implementam um mecanismo de segurança essencial: confinamento/isolamento. Implementam um domínio de segurança restrito para usar num pequeno conjunto de aplicações. Também fornece uma abstração comum com hardware comum (mesmo se for modificado).

Fornece mecanismos adicionais como, controlo de recursos, priorização de acesso a recursos, criação de imagens para análise e recuperação rápida para um estado conhecido.

4.4 Modelo Computacional

Conjunto de entidades (objetos) geridos pelo kernel do OS. Define a forma como as aplicações interagem com o kernel. Exemplos: Identificadores de user, processos, memória virtual, ficheiros e sistemas de ficheiros, ...

4.5 Identificadores de User (UID)

Para o kernel do OS, um user é um número, estabelecido no login, o User ID (UID).

Todas as atividades são executadas num computador em nome de um UID. Os UID's permitem ao kernel saber o que é permitido ou não a um user.

- Linux: UID 0 is omnipotent (root)
 - Administration activities are usually executed with UID 0
- macOS: UID 0 is omnipotent for management
 - Some binaries and activities are restricted, even for root
- Windows: concept of privileges
 - For administration, system configuration, etc.
 - There is no unique, well-known administrator identifier
 - Administration privileges can be bound to several UIDs
 - Usually through administration groups
 - Administrators, Power Users, Backup Operators

4.6 Identificadores de Grupo (GID)

OS também têm group identifiers. Um grupo é composto por 0 ou mais users e pode ser composto por outros grupos. Group ID: inteiro (Linux, Android, macOS), UUID (Windows).

Um user pode pertencer a vários grupos. Os direitos de um user são o direito do seu UID e dos seus GID's.

Em Linux, as atividades executam sempre por baixo do scope de um conjunto de grupos. 1 grupo primário (ownership dos ficheiros criados), múltiplos secundários (condicionam o acesso aos recursos).

4.7 Processos

Um processo define o contexto da atividade, para tomar decisões relacionadas com segurança ou outros propósitos (e.g scheduling).

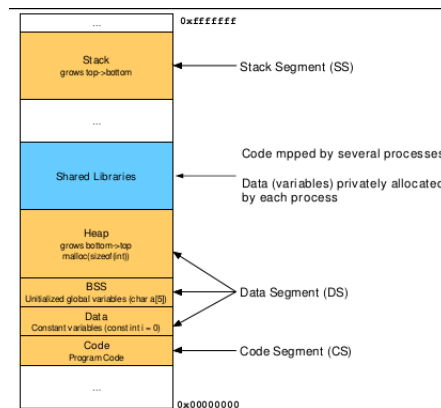
Contexto relacionado à segurança. Identidade efetiva (eUID e eGID), fundamental para controlo de acesso, pode ser o mesmo que a identificação do user que lança o processo. Os recursos usados são ficheiros abertos, áreas reservadas de memória virtual, ...

4.8 Memória Virtual

O espaço de endereçamento onde a atividade acontece, tem o tamanho máximo definido pela arquitetura do sistema ($32 \text{ bits} \rightarrow 2^{32}$, $64 \text{ bits} \rightarrow 2^{64}$). Gere em pequenos blocos de memória, páginas (4KiB).

Memória virtual pode ser escassa, uma vez que, apenas as páginas usadas devem ser alocadas.

Memória virtual é mapeada a RAM quando usada. A escolha de como gerir estes espaços é muito importante (evitar fragmentação, ...).

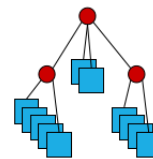


4.9 Sistema de Ficheiros

File System: objects

Hierarchical structure for storing content

- Provide a method for representing mount points, directories, files and links



Mount Point

- An access to the root of a specific FS
- Windows uses letters (A:, .. C:..)
- Linux, macOS, Android use any directory

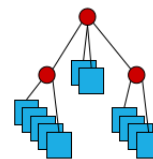
Directory (or folder)

- A hierarchical organization method
 - Similar to a container
- Can contain other directories, files, mount points, links
- The first (or top-most) is called by root

Links

- Indirection mechanisms in FS
- Soft Links: point to another feature in any FS
 - Windows: Shortcuts are similar to Soft Links, but handled at the application level
- Hard Links: provide multiple identifiers (names) for the same content (data) in the same FS
 - Usually allowed only for files

File System: files



Serve to store data on a persistent way

- But longevity is given by physical support and not by the file concept ...
- Erasing often means marked as deleted

Ordered sequences of bytes associated with a name

- The name allows you to retrieve/reuse these bytes later
- Its contents can be changed, removed, or added
 - As well as the name
- They have a protection that controls their use
 - Read, write, run, remove, lock, etc. permissions
 - The protection model depends on the file system

4.9.1 Sistema de Ficheiros: Mecanismos de Segurança

Mecanismos de proteção obrigatórios, para o owner, users e grupos são permitidos. Permissões são Read, Write e Run.

Mecanismos de proteção discricionários, regras específicas para users.

Mecanismos adicionais, como implicit compression, signature, encryption, ...

4.10 Canais de Comunicação

Permitem a troca de dados entre atividades distintas mas cooperativas.

Esta presente em qualquer sistema, todas as aplicações usam estes mecanismos.

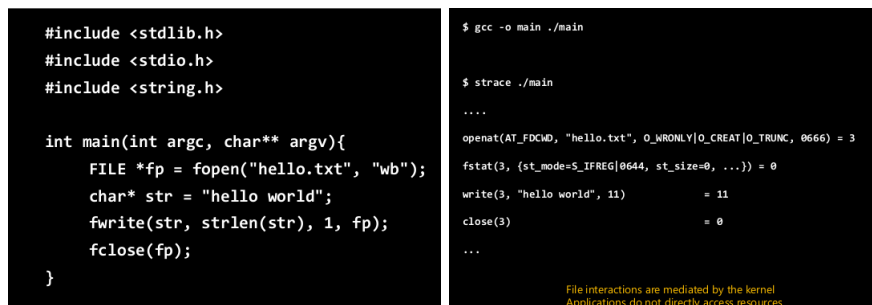
Processos no mesmo SO/máquina, através de pipes, UNIX sockets, streams, ... Comunicação entre processos e kernel, através de system calls, sockets.

Processos em máquinas diferentes, TCP/IP e UDP/IP sockets.

4.11 Controlo de Acesso

O kernel do OS é um monitor de controlo de acesso, controla todas as interações com o hardware. As aplicações nunca usam recursos diretamente e também controla todas as interações entre computational model entities.

Sujeitos são tipicamente processos locais, mas também mensagens de outras máquinas.



```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char** argv){
    FILE *fp = fopen("hello.txt", "wb");
    char* str = "hello world";
    fwrite(str, strlen(str), 1, fp);
    fclose(fp);
}
```

```
$ gcc -o main ./main
$ strace ./main
....
openat(AT_FDCWD, "hello.txt", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=0, ...}) = 0
write(3, "hello world", 11)      = 11
close(3)                        = 0
...
```

File interactions are mediated by the kernel.
Applications do not directly access resources

4.12 Controlo de Acesso Obrigatório

Há muitos casos de controlo de acesso obrigatório no OS, parte da lógica do modelo computacional, não são moldáveis por users ou administradores.

Exemplos em Linux: A root pode fazer tudo, os sinais para processos são enviados apenas pela root ou pelo owner.

Exemplos em macOS: A root faz quase tudo, mas não pode alterar binários e diretórios da Apple.

4.13 Controlo de Acesso Discrecionário

Users podem criar um conjunto de regras de controlo de acesso, podem ser definidas apenas pelo owner/user.

Exemplo:

- **Discretionary Access Control Lists (ACL)**, listas expressivas que limitam o acesso a recursos em Linux;
- **Linux Apparmor**, guarda settings em `/etc/apparmor.d` com limites de aplicações. As regras aplicam-se automaticamente aos processos, independentemente do user;
- **macOS sandboxd**, aplicações são lançadas em contextos isolados (sandbox), esta sandbox contém informação sobre o que entra/sai;