Módulo 3: Defensas, controles y contramedidas

Lección 2: Modelos de Control de Acceso y Análisis de Riesgos

Objetivos de la Lección

Al concluir la sesión, el estudiante podrá:

1. **Diferenciar** los modelos Discrecional (DAC), Mandatorio (MAC), Basado en Roles (RBAC) y Basado en Atributos (ABAC), indicando sus ventajas y limitaciones.
2. **Construir** matrices de control de acceso que asignen permisos de forma coherente con los principios de seguridad.
3. **Esbozar** un análisis de riesgos básico que relacione vulnerabilidades, amenazas y controles de acceso, estableciendo prioridades de mitigación.

Introducción a la Lección

El **control de acceso** determina *quién* puede *hacer qué* sobre *qué recurso*. Elegir el modelo correcto no solo evita accesos no autorizados, sino que también simplifica auditorías y reduce la exposición al riesgo. Después de revisar los cuatro modelos principales (Discrecional (DAC), Mandatorio (MAC), Basado en Roles (RBAC) y Basado en Atributos (ABAC)), aplicaremos una matriz de acceso a un caso práctico y finalizaremos con un bosquejo de análisis de riesgos que conecte controles y amenazas.

Desarrollo del Tema

Link: [Access Control Lists (ACLs) Explained for Beginners]

Control de acceso discrecional (DAC)

- **Definición:** es un modelo de seguridad donde **el dueño de un recurso decide quién puede acceder y qué permisos tiene**.
- **Base teórica (HRU 1976).** Los permisos se modelan con la *matriz de acceso* (S,O,R) (S,O,R): filas = sujetos, columnas = objetos, celdas = derechos (ej. Como en una base de datos). Las operaciones `grant` (Conceder permisos a un sujeto sobre un objeto) y `revoke` (revocar permisos) pueden ser realizadas por el *propietario* del objeto.
- **Características clave:** herencia simple, flexibilidad, permisos explícitos en ACL (Access Control Lists).
- **Ejemplo:** en un sistema UNIX, el usuario *alice* otorga `rw-` (**`read and Write`**) a *bob* sobre `project.txt`.
- **Ventajas:** facilidad de uso, granularidad al nivel de archivo.
- **Limitaciones:** difícil escalar en entornos grandes, más vulnerable a abusos o errores del usuario (ej. compartir un archivo con alguien no autorizado).
- **Listas ACL vs. Capabilities.**
    1. *ACL* (UNIX, NTFS): objeto (**recurso protegido**) mantiene lista de `(sujeto, derecho)`.
    2. *Capability list:* En este modelo, en vez de que **el objeto guarde la lista de quién accede** (ACL), es **el sujeto (usuario o proceso)** quien guarda **tokens de capacidad** que le otorgan acceso directo a ciertos objetos.
    3. (ex.: Capsicum, CHERI CPU).
- **Riesgo típico – "Trojan Horse Problem".** Si *alice* ejecuta un binario de *bob*, ese proceso hereda los derechos de *alice* y puede leer sus archivos.
- **Buenas prácticas**
    1. Habilitar bit `umask` apropiado 027 para evitar permisos grupales mundiales.
    2. Activar `setfacl` en Linux para granularidad sin cambiar dueño/grupo.

Control de acceso mandatorio (MAC)

- **Definición:** El **Control de Acceso Mandatorio (MAC, *Mandatory Access Control*)** es un modelo de seguridad donde **las reglas de acceso las define la política del sistema operativo o la organización, no el usuario propietario**.
- **Ejemplo e implementación:** sistema militar clasifica documentos como *Secreto, Confidencial, Público* y usuarios con nivel *Secreto* pueden leer todo, pero no degradar la clasificación.
- **Ventajas:** excelente para información altamente sensible; elimina “trust” en el usuario (cuando decimos “trust” nos referimos en que **la seguridad no depende de que el usuario actúe bien**, sino de que el sistema impone reglas estrictas que el usuario no puede alterar.).
- **Limitaciones:** muy rígido, costoso de administrar para datos dinámicos o colaboración ad-hoc. Cuando decimos **colaboración ad-hoc** esto se refiere a la capacidad de que **usuarios compartan información de manera espontánea o informal**, fuera de reglas predefinidas.
- **Modelos formales**
    - **Bell-LaPadula (BLP)** – diseñado para entornos militares/gubernamentales donde la prioridad es **proteger información confidencial**. Para esta plataforma existen dos reglas claves **no read up** y **no write down**.
        - **No read up** – un usuario con nivel bajo no puede leer información de nivel superior.
        - **No write down** – un usuario con nivel alto no puede escribir en un nivel inferior, para evitar filtración hacia abajo.
    - **Biba** – protege la **integridad de la información** (evitar corrupción o alteración indebida). Dos reglas claves *Simple Integrity* (“no read down”) y *Star*-Integrity (“no write up”).
        - **No read down** - un usuario en un nivel alto de integridad **no puede leer** datos de un nivel inferior.

        - **No write up** - un usuario en un nivel bajo **no puede escribir** en un nivel alto.
    - **Clark-Wilson** – diseñado para entornos **empresariales**, donde la seguridad se logra mediante **procesos bien definidos**. Para Clark-Wilson existen dos conceptos claves los cuales son: **Transacciones bien Formadas** y **Separación de Funciones.**
        - **Transacciones bien Formadas –** el acceso a los datos debe hacerse **solo a través de programas/aplicaciones autorizadas** que validan la operación.
        - **Separación de Funciones –** ninguna persona debe tener control total de un proceso crítico.
- **Desafíos operativos**
    - Alta tasa de falsos negativos (denegaciones legítimas) si la política no cubre todos los flujos de autenticación.
    - Complejidad de etiquetar dinámicamente objetos recién creados.

Control basado en roles (RBAC)

- **Definición:** es un modelo de seguridad donde los **permisos se asignan a roles**, (p. ej., *Administrador*, *Cajero*); luego los usuarios heredan permisos según su rol.
- **Pasos de diseño:**
    1. Identificar funciones de negocio.
    2. Mapear operaciones a roles.
    3. Asignar usuarios a roles.
- **Ejemplo:** un hospital crea roles *Médico*, *Enfermería*, *Recepción*—cada uno con conjuntos de operaciones predefinidos sobre historias clínicas.
- **Ventajas:** escalabilidad, alineación con organigramas, facilita principios de *least privilege*.
- **Limitaciones:** proliferación de roles (“role explosion”) si no se gestionan ciclos de vida y excepciones. Exceso de roles configurados (no se eliminan roles pasados, mala gestión y configuración de roles).

Control basado en atributos (ABAC)

- **Definición:** El **Control Basado en Atributos (ABAC, *Attribute-Based Access Control*)** es un modelo de seguridad en el que las decisiones de acceso se toman según **atributos** del:
    - **Usuario** (ej. cargo=profesor, edad≥18, departamento=IT).
    - **Recurso** (ej. tipo=documento, clasificación=confidencial).
    - **Acción** (ej. leer, escribir, borrar).
    - **Contexto** (ej. hora=9am–5pm, ubicación=campus).
- **Política típica:** “Permitir `read` a cualquier documento a un usuario dentro de un departamento, como por ej. Finanzas, y que este pueda acceder a dichos documentos hasta una hora en especifica, ej. 8:00am - 5:00pm.
- **Ejemplo práctico:** servicio en la nube verifica que `user.department=="Finance"` **y** `device.trustScore > 80` (el equipo tiene un puntaje de confianza mayor a 80, esto se identifica por varios factores tales como si el equipo tiene antivirus actualizado, tiene insertado los parches de seguridad adecuados, entre otros factores a medir) antes de conceder acceso.
- **Ventajas:** granularidad, adaptabilidad y soporte de iniciativas Zero-Trust.
- **Limitaciones:** complejidad de las reglas, necesidad de motores de política (e.g., XACML) y fuentes de atributos fiables en tiempo real.
- **XACML (eXtensible Access Control Markup Language)**
    1. **XACML (eXtensible Access Control Markup Language)** es un **estándar basado en XML** para definir y aplicar **políticas de control de acceso** de forma centralizada y flexible.
        a. **Puntos Principales:**
            i. Define **qué recursos** pueden ser accedidos, **por quién**, **qué acciones** puede ejecutar y **en qué condiciones**.
            ii. Separa la **decisión de acceso** (servidor de políticas, PDP) de la **ejecución** (aplicación o sistema, PEP).

iii. Permite expresar reglas complejas con atributos (usuario, recurso, acción, contexto).

2. **Componentes:** PDP (Policy Decision Point), PEP (Policy Enforcement Point), PAP (Policy Administration Point), PIP (Policy Information Point).
3. Política ejemplo (pseudo-XACML):
   pgsql

```
Permit if user.department == "Finance"
          and resource.type  == "Invoice"
          and action in ["read","approve"]
          and environment.time < 18:00
```

Se permite acceso a facturas únicamente a usuarios del departamento de Finanzas, para leer o aprobar, y solo antes de las 6:00 pm.

- **OPA/Rego** – motor ligero de políticas para microservicios; las decisiones se exponen vía gRPC/REST.
- **Zero-Trust Alignment** – ABAC se combina con **Continuous Adaptive Risk & Trust Assessment (CARTA)**: cada request se evalúa en línea con señales de postura (EDR, contexto geográfico) antes de emitir `Deny` o `Permit`.

Construcción de una matriz de acceso (Caso práctico mini-ERP)

Objetos / Operaciones	Rol: Contador	Rol: Auditor	Rol: Administrador de IT
Facturas (create/read/update/delete)	C · R · U	R	C · R · U · D

Plan de cuentas (read/update)	R	R	R · U
Logs del sistema (read)	—	R	R
Usuarios (create/update/delete)	—	—	C · U · D

Leyenda: **C** = Create, **R** = Read, **U** = Update, **D** = Delete.

1. **Validar least privilege:** el contador no necesita acceso a logs.
2. **Identificar huecos:** el auditor requiere sólo lectura; impedir write arbitrario.
3. **Exportar a ACL o política ABAC:** convertir matriz en reglas JSON o XACML para el motor de autorización.

Análisis de riesgos y planificación

1. **Identificar activos:** datos financieros, logs, credenciales de usuario.
2. **Amenazas relevantes:**
    - Insider con permisos excesivos (violación de integridad).
    - Ataque externo por medio credenciales filtradas (confidencialidad).
3. **Vulnerabilidades:**
    - “Role explosion” permite asignaciones incorrectas.
    - Falta de MFA para cuentas de alto privilegio.
4. **Evaluar riesgo (probabilidad × impacto):**
    - *Insider* + permisos Contador → Impacto alto (puede modificar facturas), Probabilidad media.
    - Credenciales filtradas + acceso Auditor (solo lectura) → Impacto medio, Probabilidad alta.
5. **Planificar controles:**
    - Convertir Contador → ABAC con condición de hora/ubicación.
    - Activar MFA y rotación trimestral de roles.
    - Revisiones trimestrales de la matriz (control administrativo).

Resultado: **riesgo residual** aceptable, documentado en plan de seguridad y revisado por el CISO.

6. **Bajo un enfoque metodológico,** el análisis de riesgo y planificación trabaja con la metodología ISO 27005 y FAIR. A continuación, se muestra una tabla donde se presenta los pasos a exactitud que se deben de tener en cuenta dentro de un Enterprise o compañía cuando se analiza el riesgo a nivel cibernético, hardware y software de una red y sobre que acciones y herramientas sugeridas utilizar de acuerdo con el paso que corresponda, estos son:

Paso	Acción concreta	Herramienta sugerida
1. Identificación de activos	Enumerar tablas BD, APIs, microservicios	CMDB + OWASP ASVS
2. Amenazas & vectores	Insider, APT, error config.	MITRE ATT&CK Navigator
3. Vulnerabilidades	Falta de revisión de roles, token sin expiración	OpenSCAP, IAM Analyzer
4. Probabilidad (Loss Event Frequency)	Basada en histórico de alertas SIEM (+Bayes)	Splunk RiskIQ

Paso	Acción concreta	Herramienta sugerida
5. Impacto (Loss Magnitude)	$$ por minuto de downtime, multas GDPR	Cost Calculator
6. Riesgo inherente	Riesgo = Probabilidad × Impacto → mapa Heat 5×5	Excel/Power BI
7. Controles & coste	MFA admin = $2 K/año, motor ABAC = $10 K (Capital Expenditure ó Gasto de Capital)	ROI (Retorno de Inversión) Calculator
8. Riesgo residual	Reevaluar después del control aplicado	Dashboard GRC

- **Quick-win (acción de seguridad rápida y de alto impacto):** implementar *least-privilege* en 10 cuentas “Break-Glass” con llave FIDO2 reduce riesgo de insider >40 %.

    1. **Least-privilege**

        - Principio de **mínimos privilegios**: cada cuenta solo debe tener los permisos estrictamente necesarios.
        - Reduce la superficie de ataque y el abuso de permisos excesivos.

    2. **Cuentas Break-Glass**

        - Son cuentas de **emergencia** con permisos de administrador total (ej. para usar si los sistemas de autenticación normales fallan).
        - Normalmente están bloqueadas o bajo control estricto y solo se usan en situaciones críticas.

    3. **Llave FIDO2**
        - Autenticación fuerte con un **dispositivo físico** (ej. YubiKey).
        - Protege las cuentas de accesos indebidos, incluso si la contraseña es robada.

    4. **Reducción de riesgo de insider >40 %**
        - Al aplicar estas medidas (least-privilege + FIDO2 en cuentas críticas), se **mitiga significativamente el riesgo de abuso interno** (empleados malintencionados o comprometidos).
        - El “>40 %” es una estimación típica en modelos de riesgo: significa que con esta acción simple ya se reduce casi la mitad del riesgo asociado a insiders.

- **Plan de mejora de madurez (CMMI-style (Capability Maturity Model Integration)):**
    1. **Inicial:** DAC (Control de Acceso Discrecional) esporádico.
        - Los permisos se otorgan de manera manual y poco uniforme (ej.: el administrador da acceso caso por caso, sin reglas claras).
    2. **Definido:** RBAC documentado.
        - Se establecen roles (ej.: administrador, desarrollador, usuario de solo lectura) y se documentan las políticas de quién puede hacer qué. **Mejora en la estandarización:** todos los usuarios de un mismo rol tienen permisos uniformes.
    3. **Gestionado:** ABAC + reviews trimestrales.

        - Se añaden atributos dinámicos (ej.: hora del día, ubicación, tipo de dispositivo, nivel de riesgo).
        - Se implementa un ciclo de gestión: revisiones periódicas (cada trimestre) para validar que los accesos siguen siendo adecuados.
    4. **Optimizado:** ABAC + motor RBA + IaC-policies.
        - “Motor RBA” puede referirse a Risk-Based Access - el acceso se adapta según el nivel de riesgo en tiempo real (ej.: si te conectas desde una red desconocida, pide MFA extra).
        - IaC-policies = Infrastructure as Code policies - las políticas de acceso se definen como código (ej.: en Terraform/OPA/CloudFormation), lo que permite consistencia, trazabilidad y despliegue automatizado.
        - Es el estado más avanzado: políticas inteligentes, revisiones automáticas, seguridad dinámica.

Relación con Otros Conceptos

- Los modelos de acceso aplican los conceptos de **autenticación** vistos en la lección anterior: una credencial robusta es inútil si los permisos son excesivos.
- El análisis de riesgos aquí es un antecedente para el **módulo de Defensas y Contramedidas** donde se detallarán técnicas de mitigación más avanzadas (firewalls, IDS, criptografía).

Resumen de la Lección

Distinguimos los modelos DAC, MAC, RBAC y ABAC, enfatizando cuándo y por qué elegir cada uno. Demostramos la construcción de una matriz de acceso en un mini-ERP y empleamos esa matriz para identificar riesgos y priorizar controles. El control de acceso no es estático: debe evaluarse periódicamente a la luz de cambios organizacionales y del panorama de amenazas.

Actividad de la Lección

Workshop – De la matriz al riesgo (90 min)

1. Grupos de tres reciben un escenario (universidad, clínica, startup SaaS).
2. Elijan un modelo primario (RBAC o ABAC) y diseñen una matriz simplificada.
3. Localicen al menos dos riesgos derivados de permisos excesivos o reglas ambiguas.
4. Propongan controles de corrección y redacten un mini-informe (máx. 2 pág.).
5. Suban el PDF al LMS y preparen un pitch de 5 min para la próxima clase.

Referencias Adicionales

- Pfleeger, C. P., Pfleeger, S. L., & Coles-Kemp, L. (2023). *Security in Computing* (6.ª ed.). Addison-Wesley.
- Stallings, W., & Brown, L. (2017). *Computer Security: Principles and Practice* (4.ª ed.). Pearson.
- Bishop, M. (2018). *Computer Security: Art and Science* (2.ª ed.). Addison-Wesley.
- NIST SP 800-162 (2019). *Guide to Attribute Based Access Control*.
- NIST SP 800-53 Rev 5 (2020). Controles AC-x para sistemas federales.