

Séance 10

Sécurité des systèmes d'exploitation



Ce(tte) œuvre est mise à disposition selon les termes de la Licence Creative Commons Attribution – Pas d'Utilisation Commerciale – Pas de Modification 4.0 International.

Rappels

- Gestion des **entrées/sorties** par le système d'exploitation
 - Aspects hardware avec contrôleur et driver de périphérique
 - Mécanisme de polling et d'interruption
 - Aspects software avec les différents types d'E/S
- **Sous-système E/S** du kernel d'un OS

Techniques de buffering, caching et spooling

Objectifs

- **Protection** d'un système d'exploitation
 - Authentification des utilisateurs et mot de passe
 - Techniques de contrôle d'accès et politiques de contrôle
- **Hardening** des systèmes d'exploitation

Recommandations du NIST pour hardener un OS

Protection

- Application du **principe du moindre privilège**
Ne donner que les privilèges nécessaires pour la tâche à réaliser
- **Minimiser les risques** en cas de vol des droits
Limiter les composants compromis en cas d'attaque
- Protéger le système d'exploitation d'**escalade de privilèges**
Un attaquant ne doit pas pouvoir gagner des droits

Domaine de protection

- Système informatique est une collection de **processus et objets**
 - Objets hardware : CPU, segment mémoire, imprimante...
 - Objets software : fichier, programme, sémaphore...
- Objets identifiés par **nom unique** agit comme TAD
 - Accès à un objet se fait via des opérations*
- Deux règles conditionnent l'**utilisation d'un objet** par processus
 - Accès aux objets pour qui le processus a l'autorisation
 - Et seulement ceux pour finir sa tâche (*need-to-know principle*)

Structure du domaine

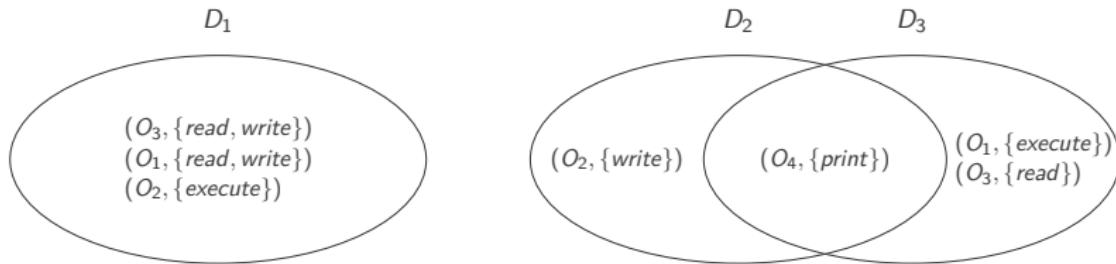
- Processus opère à l'intérieur d'un domaine de protection

Spécifie les ressources que le processus peut accéder

- Domaine définit ensemble objets avec types d'opérations

Ensemble de droits d'accès de la forme (objet, droit)

- Chaque utilisateur/processus/procédure peut être un domaine



Authentification



Authentification

- Pouvoir **authentifier** les utilisateurs d'un système
Vérifier que l'identité d'un utilisateur est authentique
- **Trois éléments** permettant une authentification
 - Une possession : une clé, une carte...
 - Une connaissance : un identifiant et un mot de passe...
 - Un attribut : une empreinte, une signature...
- **Deux étapes** pour un processus d'authentification
 - **Identification** : présenter un identifiant au système
 - **Authentification** : prouver lien entre entité et identifiant

Mot de passe

- Authentification avec un nom d'utilisateur et son **mot de passe**

Comparaison avec le mot de passe stocké dans le système

- Plusieurs **associations** possibles des mots de passe

- Un mot de passe pour chaque objet à protéger
- Protection d'un ensemble de droits d'accès par mot de passe

- OS limitent à **un seul** mot de passe pour tous les droits

Compromis entre sécurité et facilité d'utilisation

Vulnérabilité

- L'utilisation de mot de passe **pas du tout sécurisé**
 - Facile à deviner, peut être exposé, sniffé...
 - Transfert illégal vers un utilisateur non autorisé !
- Pin de quatre chiffres seulement 10000 possibilités
 - En moyenne, 5000 tentatives (en 5s si un test par ms)*
- Possibilité de voir un mot de passe lors de son **exposition**
 - Shoulder surfing, network sniffing, keylogger...*

Sécuriser les mots de passe

- Les mots de passe doivent être gardé **secrets dans le système**
Stockage sécurisé, tout en pouvant vérifier le mot de passe
- Stockage sous **forme hachée** (hachage cryptographique)
Impossible de retrouver le mot de passe à partir du hash
- Attaque **brute-force** sur la liste des hashes
Il faut sécuriser la base de données des hashes
- Autoriser plus de caractères en permettant des **passphrase**

Schéma de mot de passe Unix (1)

- Ajout d'un **nouveau mot de passe** dans le système

Le sel choisi par le système est utilisé pour calculer le hash

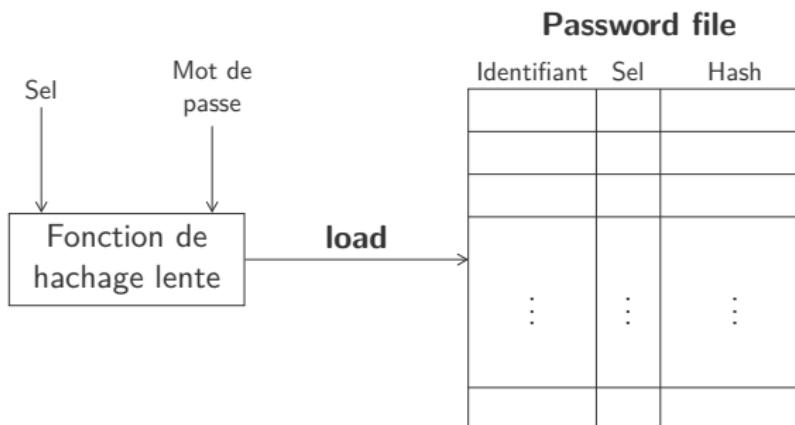
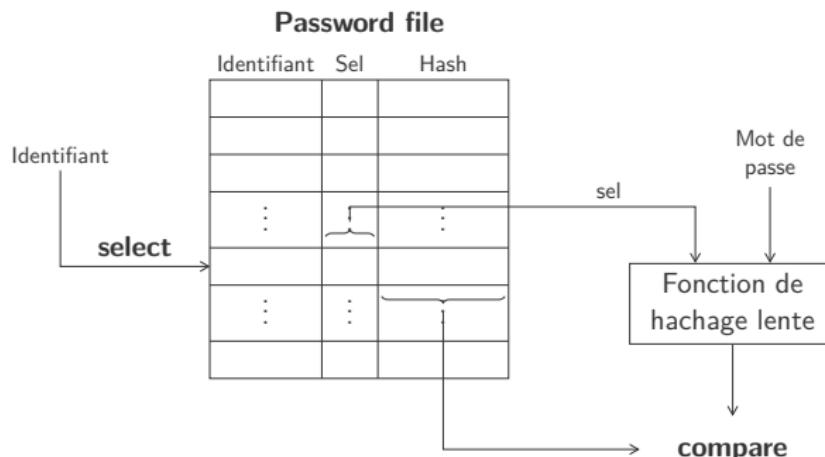


Schéma de mot de passe Unix (2)

■ Vérification d'un mot de passe

Dans le but d'authentifier un utilisateur



One-Time Password (OTP)

- Utilisation d'un ensemble de **mots de passe pairés**
 - L'utilisateur donne l'autre partie de la paire choisie par l'OS
 - Challenge de l'utilisateur qui doit y répondre correctement
- Utilisation d'un **algorithme** comme mot de passe
 - Partage d'un mot de passe symétrique pw jamais exposé
 - Le système présente un challenge ch à l'utilisateur
 - L'utilisateur calcule $H(pw, ch)$ et envoie au système
 - Le système calcule également et compare les résultats

Two-Factor Authentication

- Générateur de OTP à l'aide d'un dispositif physique
Après input de l'utilisateur, dispositif génère un OTP
- Utilisation de **deux composants** pour l'authentification
 - PIN code d'abord vérifié par le générateur d'OPT
 - Combinaison d'une possession et d'une connaissance

Donnée biométrique

- Utilisation de **mesures biométriques**
 - Lecteurs de paumes ou mains des utilisateurs
 - Map de t°, longueur/largeur des doigts, patterns de lignes...
- Lecteur d'**empreinte digitale** plus précis et moins cher
 - Patterns des crêtes des doigts comme séquence de nombres
 - Utilisation dans un système two-factor authentication



Contrôle d'accès

Contrôle d'accès

- L'OS et le système de fichiers exercent un **contrôle d'accès**

Contrôle de l'accès aux fichiers ou aux ressources du système

- Plusieurs accès fournis après **login fructueux**

- Autorise l'accès à un ou des hôtes et à des applications
- Pas suffisant pour système qui contient beaucoup de données

- **Utilisateurs identifiés** de manière unique dans le système

- Profil avec opérations permises et fichiers accessibles
- L'OS peut imposer des règles sur base du profil utilisateur

Matrice d'accès (1)

- Modèle général de contrôle d'accès avec matrice d'accès

Deux dimensions avec utilisateurs et ressources

- Trois éléments de base composent la matrice d'accès

- **Sujet** peut accéder à des objets (processus)
- **Objet** dont on veut pouvoir contrôler l'accès (fichier...)
- **Droit d'accès** à l'objet pour le sujet (read, write, execute...)

- Droits pour des utilisateurs ou groupes

Les entrées de la matrice indique les droits accordés

Matrice d'accès (2)

- Matrice d'accès généralement **très creuse**

Deux manières concrètes de la décomposer plus efficacement

| | Fichier 1 | Fichier 2 | Fichier 3 | Fichier 4 |
|---------------|------------------------|------------------------|------------------------|------------------------|
| Utilisateur A | Propriétaire R W | | Propriétaire R W | |
| Utilisateur B | R | Propriétaire R W | W | R |
| Utilisateur C | R W | R | | Propriétaire R W |

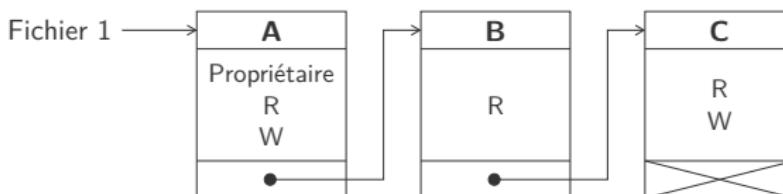
Access Control Lists

- Décomposition en colonnes donne **Access Control Lists**

Liste des utilisateurs et leurs droits, pour chaque fichier

- Possibilité d'avoir une règle **publique par défaut**

Droits définissables par utilisateur, groupe...



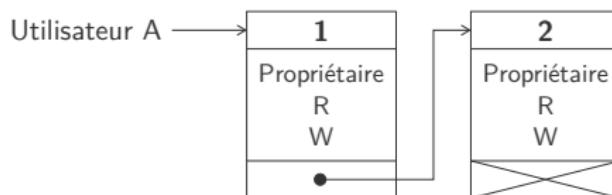
Capability Tickets

- Décomposition en lignes donne **Capability Tickets**

Liste des objets et autorisations, pour chaque utilisateur

- Possibilité de **prêter ou donner** ces tickets à d'autres

Plus grand souci de sécurité car dispersion des tickets



Politique de contrôle d'accès (1)

- Définition d'une **politique de contrôle d'accès** (ACP)

Types d'accès autorisés, dans quelle circonstance et par qui

- **Trois catégories** principales de politiques d'AC

- **Discretionary** (DAC) identité d'un demandeur et règle d'accès

Ce qu'un demandeur peut faire ou non

- **Mandatory** (MAC) labels de sécurité et autorisations

liste d'entités pouvant accéder à des ressources

- **Role-Based** (RBAC) rôles de l'utilisateur et règles par rôle

chaque rôle permet d'accéder à des ressources

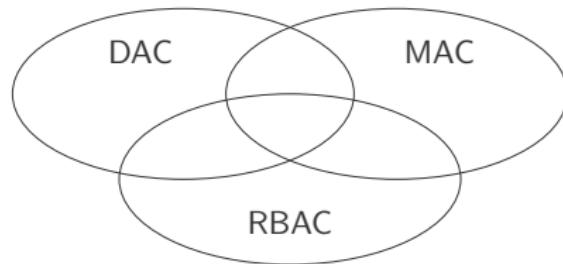
Politique de contrôle d'accès (2)

- Avec MAC, une entité peut **donner des droits** à une autre

Si elle reçoit les droits de donner des droits

- Les trois politiques ne sont **pas mutuellement exclusives**

DAC traditionnel, MAC militaire et RBAC monte en popularité



Modèle du DAC

- Modèle pour DAC basé sur trois ensembles

Sujets, objets et de règles d'accès aux objets par les sujets

- Existence d'un état de protection d'un système

Spécifie les droits d'accès à un moment donné

- Trois exigences pour un modèle du DAC

- Représenter l'état de protection du système
- Imposer les droits d'accès
- Autoriser les sujets à modifier l'état de protection

Matrice de contrôle d'accès étendu

- Attributs d'accès dans **matrice de contrôle d'accès étendu**
 $A[S, X]$ spécifie droits d'accès pour sujet S sur objet X
- Droits de chaque type d'objet géré par un **module de contrôle**
Filesystem, process manager, access matrix monitor...

| Sujets | | | Fichiers | | Processus | | Disques | |
|---------|---------|-------------------|----------|----------------|-----------|--------|---------|--------|
| S_1 | S_2 | S_3 | F_1 | F_2 | P_1 | P_2 | D_1 | D_2 |
| control | owner | owner, control | read * | read, owner | wakeup | wakeup | seek | owner |
| | control | | write * | execute | | | owner | seek * |
| | | control | | write | stop | | | |

Vérification et modification des accès (1)

- **Trois étapes** lors d'une tentative d'accès
 - 1 Un sujet S fait une requête de type α pour l'objet X
 - 2 Message de type (S, α, X) envoyé au contrôleur de X
 - 3 Le contrôleur interroge la matrice A : $\alpha \in A[S, X]$?
- Ensemble de règles pour **modifier la matrice A**
 - Flag de copie (*) permet de transférer un droit
 - Attribut `owner` pour le propriétaire d'un objet
 - Attribut `control` indique le contrôle d'un objet

Vérification et modification des accès (2)

- Trois règles pour **transférer, donner et supprimer** des droits

Opérations réservées aux détenteurs de certains attributs d'accès

- Une règle de lecture des éléments **appartenus ou contrôlés**

| Règle | Commande (par S) | Autorisation | Opération |
|-------|---|---|---|
| R1 | transfer $\left\{ \begin{array}{c} \alpha^* \\ \alpha \end{array} \right\}$ to S, X | $\alpha^* \in A[S, X]$ | stocker $\left\{ \begin{array}{c} \alpha^* \\ \alpha \end{array} \right\}$ dans $A[S, X]$ |
| R2 | grant $\left\{ \begin{array}{c} \alpha^* \\ \alpha \end{array} \right\}$ to S, X | $owner \in A[S, X]$ | stocker $\left\{ \begin{array}{c} \alpha^* \\ \alpha \end{array} \right\}$ dans $A[S, X]$ |
| R3 | delete α from S, X | $control \in A[S, X]$ ou $owner \in A[S, X]$ | supprimer α de $A[S, X]$ |
| R4 | $w \leftarrow$ read S, X | $control \in A[S, X]$ ou $owner \in A[S, X]$ | copier $A[S, X]$ dans w |

Vérification et modification des accès (3)

- Deux règles pour **créer et supprimer** des sujets/objets

Aucune autorisation nécessaire pour la création

- Possibilité de créer des **hiérarchies de sujets**

Un sujet crée un autre lui donnant un sous-ensemble de ses droits

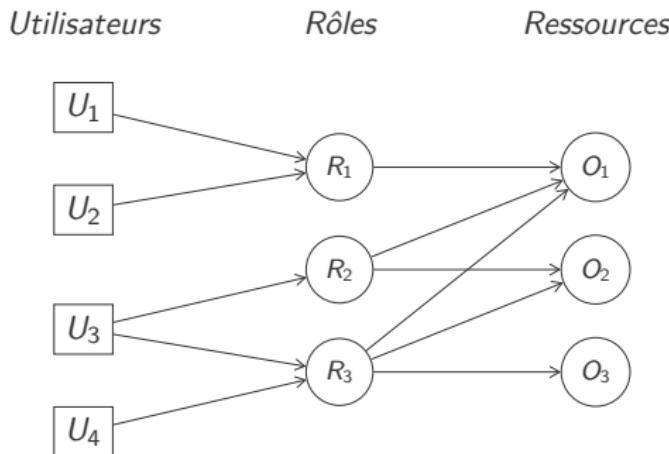
| Règle | Commande (par S) | Autorisation | Opération |
|-------|----------------------------|---------------------|---|
| R5 | create object X | — | ajouter colonne pour X dans A stocker <i>owner</i> dans $A[S, X]$ |
| R6 | destroy object X | $owner \in A[S, X]$ | supprimer colonne pour X dans A |
| R7 | create subject S | — | ajouter ligne pour S dans A exécuter create object S stocker <i>control</i> dans $A[S, X]$ |
| R8 | destroy subject S | $owner \in A[S, X]$ | supprimer ligne pour S dans A exécuter destroy object S |

Modèle du RBAC

- Définition des droits en fonction des **rôles joués**
Un rôle représente une fonction dans une organisation
- **Chaque utilisateur** peut jouer un ou plusieurs rôles
Affectation statique ou dynamique des rôles
- **Recommandation NIST** pour utiliser les rôles
Security Requirements for Cryptographic Modules

Relation M2M

- Relations many-to-many utilisateur/rôle, rôle/ressource
 - L'ensemble des utilisateurs peut souvent changer
 - L'ensemble des rôles est souvent statique



Matrice de contrôle d'accès

- Deux matrices pour modéliser les droits selon RBAC

Rôles des utilisateurs et droits comme avec DAC

| | R ₁ | R ₂ | ... | R _n |
|----------------|----------------|----------------|-----|----------------|
| U ₁ | ✓ | | | |
| U ₂ | ✓ | | | |
| U ₃ | | ✓ | | ✓ |
| U ₄ | | | | ✓ |
| U ₅ | | | | ✓ |
| U ₆ | | | | ✓ |
| : | | | | |
| U _n | ✓ | | | |

| | R ₁ | R ₂ | ... | R _n | F ₁ | F ₂ | P ₁ | P ₂ | D ₁ | D ₂ |
|----------------|----------------|----------------|-----|-------------------|----------------|----------------|----------------|----------------|----------------|----------------|
| R ₁ | control | owner | | owner, control | read * | read, owner | wakeup | wakeup | seek | owner |
| R ₂ | | control | | | write * | execute | | | owner | seek * |
| : | | | | | | | | | | |
| R _n | | | | control | | write | stop | | | |

Hardening

Master

HARDENED

Hardening des OS

- Importance de **sécuriser le système d'exploitation** d'un OS
Car toutes les applications et services dépendent de l'OS
- **Condition minimum de base** au niveau de la sécurité
OS doit être bien installé, patché et configuré
- Configuration par défaut **facilite l'utilisation**
Maximum de fonctions pour l'utilisateur plutôt que sécurité
- Existence de **guide et checklist** pour les principaux OS
Et prendre en compte les besoins spécifiques de l'organisation

Recommandations NIST

- Installer et patcher le système d'exploitation
- Harderner et configurer l'OS selon les besoins en sécurité
 - Supprimer services, applications et protocoles inutiles
 - Configurer les utilisateurs, groupes et permissions
 - Configurer les contrôles de ressources
- Installer et configurer des outils de sécurité additionnels
Antivirus, firewall, IDS...
- Vérifier que les besoins sont remplis en testant la sécurité

Installation (1)

- Système connecté au réseau et non patché est **vulnérable**

Danger durant l'installation ou l'utilisation continue

- **Ne pas exposer** le système durant son installation

- Construction du système sur un réseau isolé ou limité
- Utilisation de média amovibles (DVDs, stick USB...)
- Installation avant déploiement sur localisation finale

- Installation du **minimum nécessaire**

Que des packages nécessaires au fonctionnement du système

Installation (2)

- Sécurisation du **processus de démarrage**
 - Ajuster les options et protéger les modifications du BIOS
 - Limiter les médias à partir desquels le système peut démarrer
- Sélection et installation des **drivers de périphériques**
 - Exécution en mode kernel avec tous les privilèges associés
 - Vérification de l'intégrité et de la source de ces drivers

Patch

- Maintenir le système d'exploitation **le plus à jour possible**

Installation de tous les patches liés à la sécurité

- **Outils d'installation** automatique des patches de sécurité

Téléchargement et installation, le plus régulièrement possible

- Systèmes nécessitant **disponibilité et bon uptime**

- Introduction d'instabilité non désirée par installation de patches
- Validation des patches sur un système test avant déploiement

Installation de logiciels

- Au moins il y a de software, au moins il y a de risques

Vulnérabilité, attaques, complexité de mises à jour...

- Trouver l'équilibre entre utilisabilité et sécurité

- Fournir les logiciels qui pourraient être utiles à un moment
- Limiter le nombre de logiciels installés

- Ne pas utiliser les configurations par défaut

Elles visent facilité d'utilisation et pas la sécurité

Désactivation de services

- Mieux vaut **ne pas installer** qu'installer puis supprimer
Les scripts de désinstallation ne sont pas toujours efficaces
- **Ne pas installer** un service meilleur que le désactiver
L'attaquant peut gagner des priviléges et le réactiver

Permission utilisateur (1)

- Tous les utilisateurs n'ont pas les mêmes accès
 - Contrôle d'accès aux données et ressources du système
 - Accès par rôle ou par contrôle obligatoire des droits
- Déterminer les catégories d'utilisateurs du système

Privilèges, informations accessibles, processus d'authentification
- Plusieurs niveaux de privilèges possibles
 - Administrateur, utilisateur normal, invités...
 - Ne donner des privilèges qu'à ceux qui en ont besoin

Permission utilisateur (2)

- N'activer ses privilèges que pour les opérations le nécessitant
Et faire le reste comme un utilisateur normal
- Diminution de la **fenêtre d'opportunité** d'exploit
 - Outil d'élévation de privilèges dans certains OS
 - Log complet de toutes les élévations
- Spécification des **méthodes d'authentification**
Localement sur la machine ou serveur d'authentification

Permission utilisateur (3)

- Sécurisation des **comptes par défaut**
 - Suppression des comptes par défaut qui sont inutiles
 - Modifier tous les mots de passe par défaut
- Protection des **comptes systèmes** qui gèrent les services
Supprimer la possibilité du login interactif
- Configurer toutes les **politiques liées aux credentials**
 - Liste des méthodes d'authentification acceptées
 - Complexité minimale des mots de passe

Contrôle de ressources

- Définition des **permissions** pour les données et ressources
 - Une fois les utilisateurs et groupes sont définis*
- Définition de **limitations**
 - Quel utilisateur peut exécuter quel programme
 - Limiter l'accès en lecture/écriture de répertoires

Outil de sécurité

- Installation et configuration d'**outils de sécurité**
 - Antivirus et firewall basé sur l'hôte
 - *Intrusion Detection/Prevention System (IDS et IPS)*
 - White listing d'applications
- **Outils tiers** ou faisant partie du système d'exploitation
 - À mettre à jour avec *les patches de sécurité*

Test de sécurité

- Vérifier que toutes les mesures ont **bien été implémentées**
Identifier des vulnérabilités possibles à corriger ou gérer
- Utilisation de **checklists** pour vérifier un système
 - Certains aspects peuvent être vérifiés par des programmes
 - Vérification après le hardening et puis régulièrement

Maintenance

- Continuellement **maintenir la sécurité** du système

Environnement change tout le temps, nouvelles vulnérabilités...

- **Recommandations** du NIST pour la maintenance

- Monitorer et analyser les informations logguées
- Faire des backups réguliers
- Récupération des compromis de sécurité
- Régulièrement tester la sécurité du système
- Patcher et mettre à jour tous les logiciels

Logging

- “*Logging is a cornerstone of a sound security posture.*”
 - Information à propos d'une mauvaise chose déjà produite
 - Identification des causes d'un problème qui s'est produit
- Importance de capturer les **bonnes informations** dans les logs
 - Information du système, du réseau, des applications
 - Outils de monitoring et d'analyse automatique des logs
- Prévoir suffisamment d'espace pour **stocker les logs**

Rotation des logs et système d'archivage

Backup et archive

- Maintenir l'**intégrité** du système et des données utilisateurs
Failures hardware/software, corruption accidentelle/délibérée
- Copie de données à intervalle régulier avec **backup**
Permet récupération de données perdues ou corrompues
- Stockage de copies sur de longues périodes avec **archivage**
Exigences opérationnelles ou légales sur des mois/années...

Crédits

- <https://www.flickr.com/photos/mawari/16021501609>
- https://www.flickr.com/photos/steve_chilton/3003413345
- <https://www.flickr.com/photos/jeremybrooks/2584557302>