



# INF4420a : Sécurité Informatique

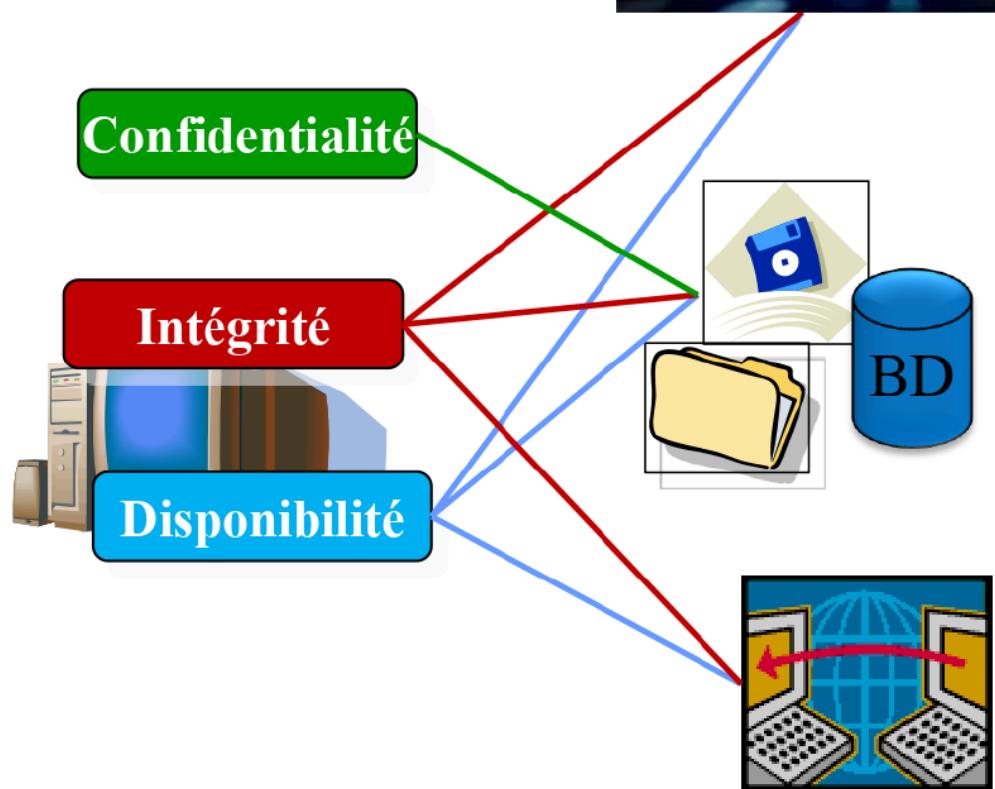
## Introduction : Concepts de base et motivation

Frédéric et Nora Cuppens



# Qu'est-ce que la sécurité informatique ?

- La sécurité informatique consiste en la protection
  - des systèmes,
  - des données et
  - des services
- Contre les menaces
  - délibérées
  - malveillantes
- Portant atteinte
  - confidentialité
  - intégrité
  - disponibilité





# Sûreté de fonctionnement vs. Sécurité informatique

- En anglais : deux termes pour « sécurité »
  - Safety
  - Security
  - Safety ≠ Security
- En français : un seul terme « sécurité »
  - Traduction de « safety » : Sûreté de fonctionnement (SDF)
  - Traduction de « security » : Sécurité informatique
    - Ou aussi « Sécurité des Systèmes d'Information » (SSI)



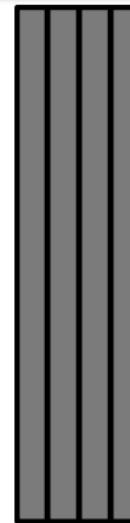
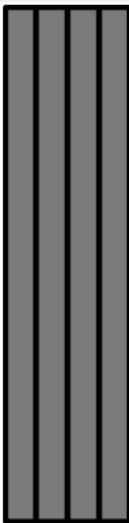
# Sûreté de fonctionnement vs. Sécurité informatique

- Différence entre la SdF et la SSI
  - La SdF traite des fautes accidentnelles (défaillances)
  - La SSI traite des fautes intentionnelles ou malveillantes (attaques)
- Mais il y a des liens entre SdF et SSI
  - Notamment une attaque (malveillante) peut causer une défaillance
  - En sens inverse, un attaquant peut profiter de (exploiter) une défaillance accidentelle pour réaliser une attaque
- Traduction de « Malveillant »
  - Malveillant = Malicious
  - Malicious ≠ Malicieux



# Sécurité informatique

## Sécurité



Disponibilité

Intégrité

Confidentialité

Propriétés de sécurité



# Rappel de génie logiciel

## Propriétés de safety et de liveness

- Propriété de liveness
  - Propriété de vivacité en français
  - Quelque chose de « bon » va arriver
  - Exemple : Ce vaccin est efficace contre la COVID 19
  
- Propriété de safety
  - Propriété de sûreté en français
  - Quelque chose de « mauvais » ne va pas arriver
  - Exemple : Ce vaccin n'a pas d'effet secondaire



# Propriétés de sécurité

- Disponibilité
  - Capacité d'un système informatique d'assurer ses fonctions sans interruption, retards ou dégradation, au moment où la demande en est faite
  - Propriété de liveness
- Capacité à rencontrer
  - les besoins et spécifications
  - les contraintes de temps, de performance et de qualité
- Applicable aux systèmes / données / services



# Propriétés de sécurité

- Disponibilité en temps fini
  - Ou propriété de disponibilité « faible »
  - Garantie que le système / la donnée / le service sera accessible une fois que la demande en est faite
  - Mais sans donner de garantie sur la durée que cela va prendre
  - Exemple : Le serveur web est disponible 7j/7 et 24h/24



# Propriétés de sécurité

- Disponibilité en temps borné (ou contraint)
  - Ou propriété de disponibilité « forte »
  - Garantie que le système / la donnée / le service sera accessible au bout d'une durée maximale spécifiée à l'avance
  - Exemple 1 : le dossier médical sera accessible au bout d'une durée maximale de 5s
  - Exemple 2 : le service de paiement en ligne sera accessible au bout d'une durée maximale de 10s
  - Pertinent notamment dans les systèmes temps-réel



# Propriétés de sécurité

- Intégrité
  - Propriété de safety
  - Il ne faut pas que quelque chose de « mauvais » arrive aux systèmes, données et / ou services
  - Nombreux sens possibles !



# Propriétés de sécurité

- Intégrité (des données)
  - Propriété associée aux données qui, lors de leur traitement ou de leur transmission, ne subissent aucune altération ou destruction volontaire ou accidentelle, et conservent un format permettant leur utilisation
- Intégrité (d'un système ou d'un service)
  1. Capacité du système ou du service à préserver l'intégrité des données qu'il gère
  2. Protection du système ou du service contre les dysfonctionnements, les agressions et les attaques



# Propriétés de sécurité

- Intégrité des données (au sens safety)
  - Exactitude
  - Précision
  - Cohérence
- Intégrité des données (au sens security)
  - Pas de modification non autorisée des données
  - Modification autorisées seulement
- Lien entre ces deux définitions

# Propriétés de sécurité

- Confidentialité
  - S'applique aux données
  - Assure que l'information n'est accessible qu'à ceux dont l'accès est autorisé
  - Propriété de safety
- Qui peut « voir » quoi ?
  - Fait référence à la notion de « Secret »
- Plusieurs dimensions
  - Intérêts publics
    - Exemple : secret militaire
  - Intérêts privées
    - Exemple : secret industriel et commercial
  - Vie privée
    - Privacy en Anglais
    - Protection des « données à caractère personnel »



# Propriétés de sécurité

- Une quatrième propriété de sécurité est aussi souvent utilisée : Auditabilité
  - Il s'agit de disposer de l'information nécessaire et suffisante pour attribuer (généralement *a posteriori*) la responsabilité d'un fait à une personne
- Plusieurs autres noms possibles
  - Traçabilité, Imputabilité, Preuve
- Le terme « Auditabilité » fait référence au besoin « d'auditer » les activités du système informatique
  - Disponibilité des journaux
  - Journaux doivent être intègres pour avoir valeur de preuve



# Cybersécurité

- Cyber sécurité : qu'est-ce qui change par rapport à la sécurité informatique ?
  - Changement de périmètre
  - Changement de paradigme



# Périmètre de la sécurité informatique

- Technologie de l'information
  - IT en Anglais : Information Technology





# Digitalisation de notre monde

Digitalisation de l'économie

Digitalisation de l'industrie

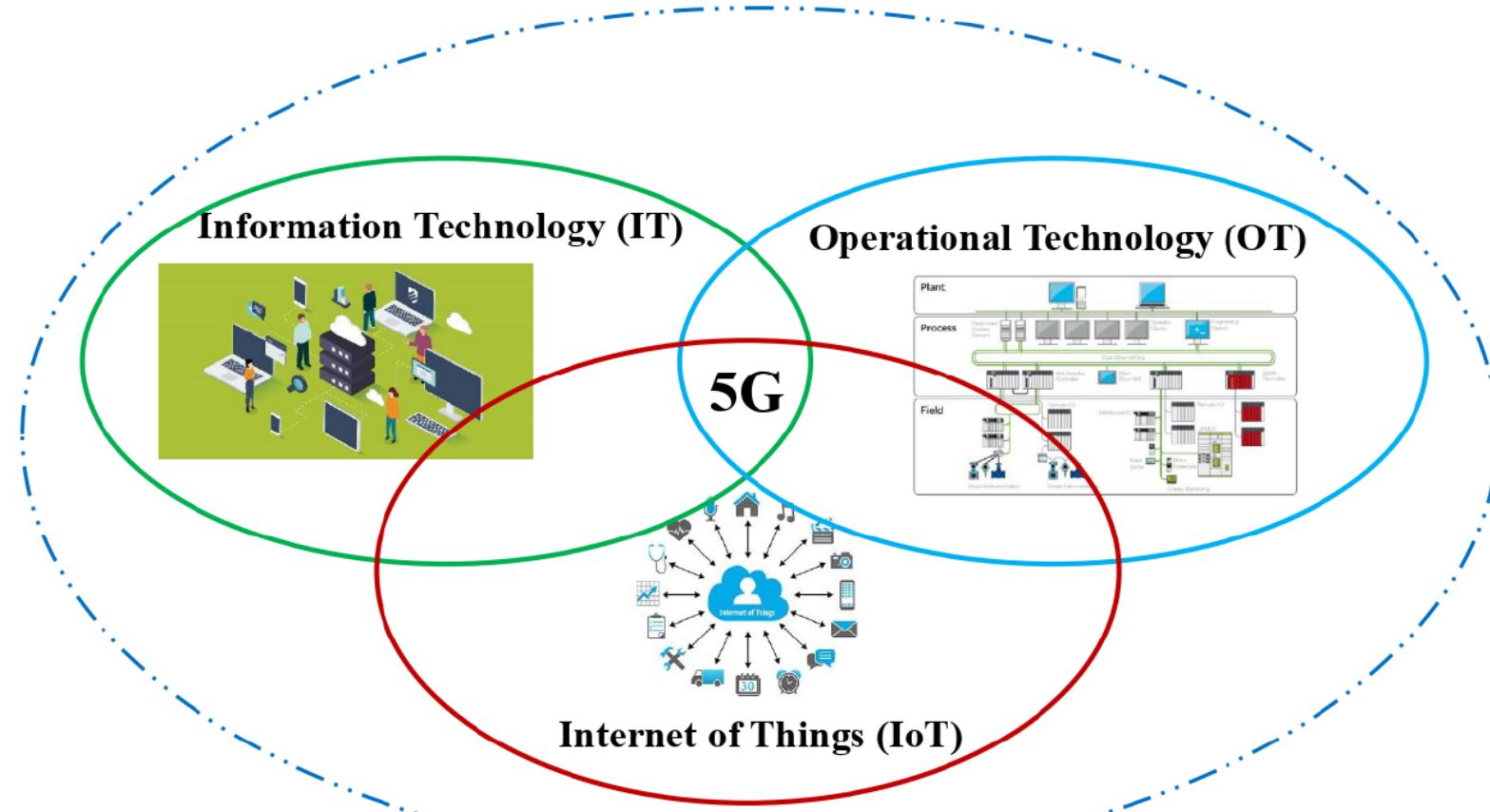
Digitalisation de la société



Changement de périmètre

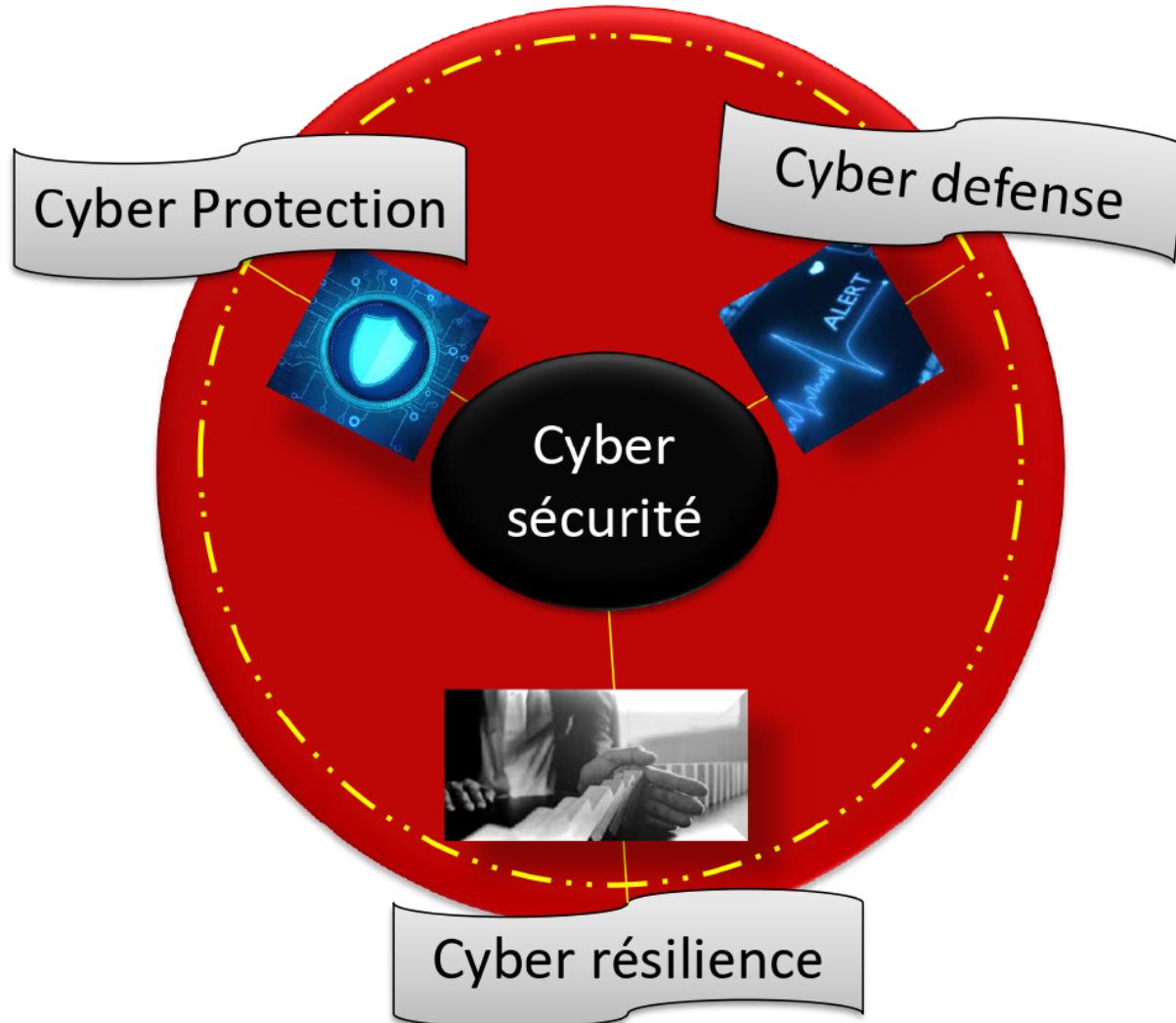


# Périmètre de la cybersécurité





# Paradigmes de la cybersécurité





# Paradigmes de la cybersécurité

- Cyber protection : définition et limites
- Définition
  - Moyens techniques, physiques et organisationnels pour protéger le système contre les cyber attaques
  - Exemples : chiffrement, contrôle d'accès, filtrage réseau, contrôle de flux, tatouage, anonymisation, ...
- Limites
  - Impossible d'assurer une protection à 100%
  - Faute de conception, d'implantation, d'utilisation
  - Evolutions techniques et technologiques
  - L'erreur humaine
  - Attaques internes



# Paradigmes de la cybersécurité

- Cyber défense : définition et limites
- Définition
  - Mesures techniques ou organisationnelles permettant la surveillance, l'appréciation de la sécurité et la réaction face aux cyber attaques
  - Exemples : IDS, SIEM, SOC
- Limites
  - Impossible d'assurer une détection à 100%
  - Attaques « zero day »
  - Techniques d'évasion
  - Attaques furtives

# Paradigmes de la cybersécurité

- Cyber résilience
- Définition
  - Capacité d'un système à résister à des cyberattaques qui réussissent

La question n'est pas :

Mon système va-t-il être attaqué ?

Mais,

Quand mon système va-t-il être  
attaqué ?

# Paradigmes de la cybersécurité

- Cyber résilience : d'autres propriétés
- Absorbabilité
  - Capacité du système à absorber les conséquence d'une attaque sous souffrir d'une défaillance complète
- Adaptabilité
  - Capacité du système d'ajuster son comportement en fonction des changements de l'environnement ou de sous-ensembles du système lui-même
- Recouvrabilité
  - Capacité du système de revenir dans un état normal

# Paradigmes de la cybersécurité

- Cyber résilience : quelques exemples de solutions
  - Diversification fonctionnelle
  - Défense en profondeur
  - Défense dynamique et adaptative

# Questions ?



# INF4420a: Sécurité Informatique

## Séance 2 : Analyse et gestion des risques

Nora Cuppens



# Contenu du cours

- Concept de menace
- Concept de vulnérabilité
- Concept de risque
- Evaluation des risques
- Réduction des risques
- Analyse de risques
- Méthodes d'analyse des risques



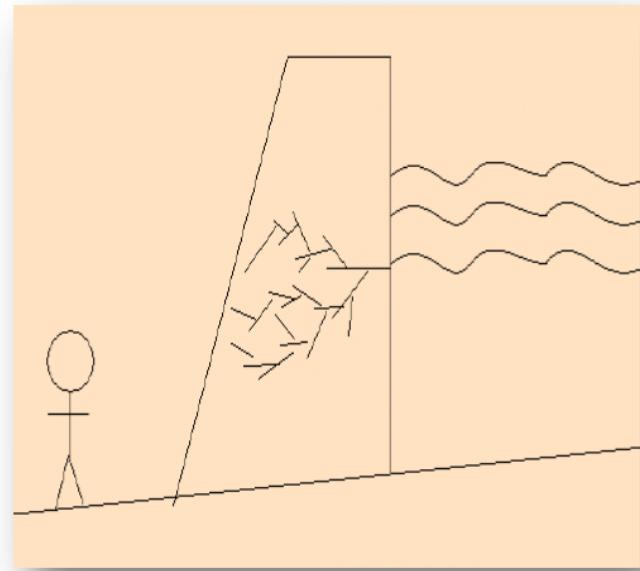
# Objectifs de la SSI

- Empêcher l'exploitation de failles (vulnérabilités) contre le système d'information par des acteurs malveillants (menace)



# Menace

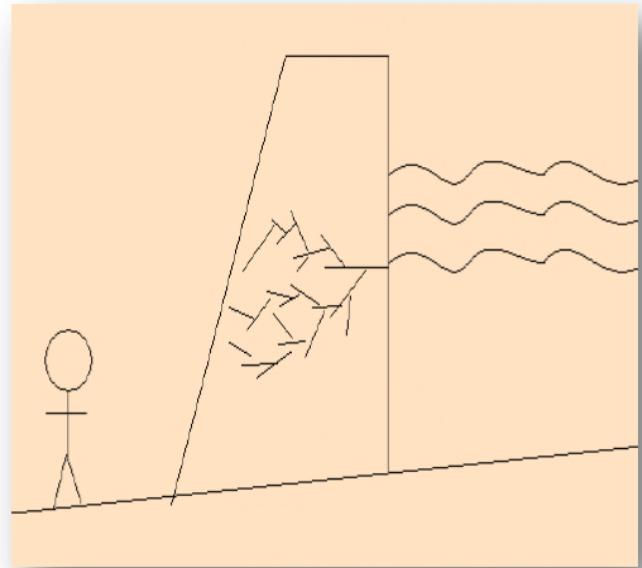
- **Bien**
  - Objet/personne ayant de la valeur
- **Acteur/agent de menace**
  - Objet/personne/entité qui met un bien à risque
- **Scenario**
  - Séquence d'évènement menant à la perte partielle ou totale de la valeur d'un bien





# Attaque

- Concrétisation de la menace
  - Acteur + Scenario
  - « Une méthode (COMMENT) par laquelle un acteur particulier (QUI) entreprend une action (QUOI) pour faire subir un dommage à un bien (POURQUOI) »





# La menace en sécurité informatique

- Quel type de menaces ?
  - Accidentelles (acteur inconscient ou absence d'acteur)
    - Catastrophes naturelles (« acts of God »)
      - feu, inondation, ...
    - Actes humains involontaires
      - mauvaise entrée de données, erreur de frappe, de configuration, ...
    - Performance imprévue des systèmes
      - Erreur de conception dans le logiciel ou le matériel
      - Erreur de fonctionnement dans le matériel
  - Malveillantes ou Délibérées (acteur conscient)
    - Attaque de déni de service (atteinte à la disponibilité)
    - Vol d'informations (atteinte à la confidentialité)
    - Modification non-autorisée des systèmes (atteinte à l'intégrité)
    - ...

➔ Sûreté

➔ Sécurité

# La menace en sécurité informatique

- Qui sont les « acteurs » ?
  - Catastrophes naturelles
  - Pirate/Hackers
    - "Script kiddies"
    - « Black hat » (et White Hat)
    - Professionnels
  - Concurrents
  - États étrangers
  - Crime organisé
  - Groupe terroriste
  - Compagnie de marketing
- Ceux à qui vous faites confiance...  Interne

 Externe

# Vulnérabilité

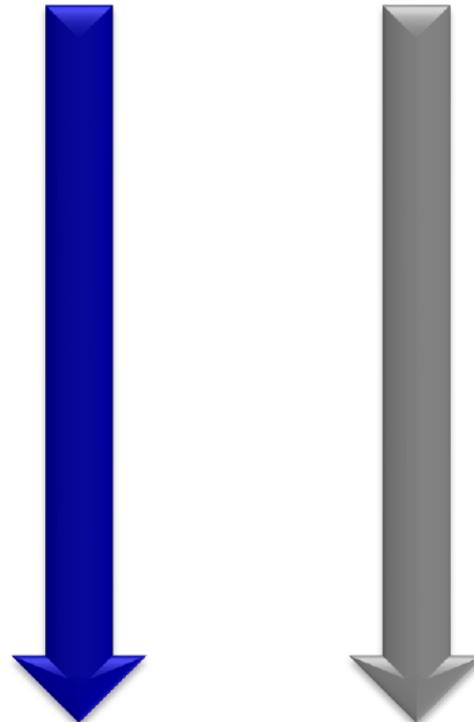
- Vulnérabilité
  - Faille qui offre l'opportunité de porter dommage à un bien
- Scénario
  - Exploitation d'une vulnérabilité par un acteur pour causer un impact
- Probabilité
  - Que la menace soit réalisée (dans une période de temps donné)
- Impact
  - Perte ou dommage à un bien

# Vulnérabilité - exemple

- **Biens**
  - Barrage et Vies humaines
- **Vulnérabilités**
  - Faiblesse du barrage et
  - Usure de la vanne ou
  - Vulnérabilité logicielle de la vanne
- **Menace**
  - Panne accidentelle de la vanne ou
  - Attaque terroriste
- **Risque**
  - Rupture du barrage et
  - Perte de vies humaines

# Vulnérabilité informatique

- Vulnérabilité = Faille
  - Causée par une erreur = bug
- Conception
- Implémentation
- Installation / Configuration
- Exploitation
- Mise à jour / Maintenance
- Suppression / Destruction





# Vulnérabilité informatique Security vs. Safety

- Partie vulnérabilité
  - Pas de différence significative entre sécurité informatique et sûreté de fonctionnement
- C'est la partie menace qui diffère
  - Menace accidentelle pour la sûreté de fonctionnement
  - Menace délibérée pour la sécurité informatique
- Conséquence très importante pour évaluer les risques





# Risque

- Définition qualitative
  - La prise en compte d'une exposition à un danger, un préjudice ou autre événement dommageable, inhérent à une situation ou une activité
  - Un risque correspond à la combinaison d'une vulnérabilité et d'une menace



# Risque

- Définition quantitative

Risque = probabilité \* impact  
= espérance de perte



- Le risque est inhérent à l'activité
  - Il est impossible de l'éliminer
  - On peut le « gérer » par
    - Réduction
    - Transfert
    - Acceptation
    - Arrêt de l'activité

**Il faut être conscient du niveau de risque avant de prendre une décision**

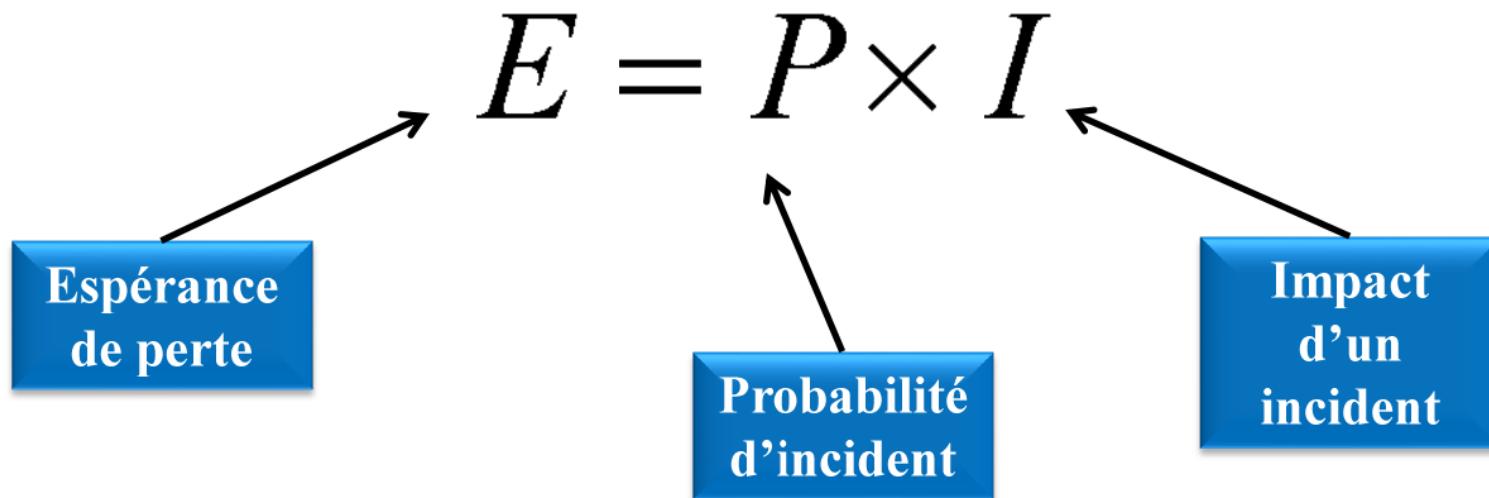
# Risque – « Reality Check »

- Il y a un risque s'il y a un enjeu réel relié au bien
  - Même si une vulnérabilité (scénario) et un acteur existent
    - Pas d'impact → pas de risque
- Il y a un risque si un scénario a une chance de se réaliser
  - Même si une vulnérabilité existe
    - Pas d'acteur → Pas de risque
  - Même s'il y a un acteur,
    - Pas de vulnérabilité → pas de scénario → pas de risque
- Mais comment gérer/estimer les risques potentiels ?
  - Vulnérabilité non encore identifiée
  - Menace non encore identifiée



# Risque

- Le risque informatique s'apparente à
  - une loterie où on ne peut pas perdre gagner !





# Risque

- Comment évaluer le risque
  - Impact
    - sous le contrôle du propriétaire du système à protéger
    - « facile » à évaluer
  - Probabilité
    - Risques naturels
      - Valeurs connues (statistiques, actuariat, historique de catastrophes, etc.)
      - Probabilité expérimentale ou fréquentielle
    - Risques délibérés
      - Acteur conscient et intelligent
      - Pas événement aléatoire

➔ Comment évaluer le risque délibéré ?

# Probabilité des risques délibérés

- Capacité
  - Savoir/connaissances ou accès au savoir
  - Outils
  - Ressources humaines
  - Argent
- Opportunité
  - Espace : avoir accès physique
  - Connectivité : existence d'un lien physique et logique
  - Temps : être « là » au bon moment
- Motivation
  - « À qui profite le crime ? » (Qui)
  - Que gagne l'attaquant ? (Quoi)
  - Combien gagne t-il ? (Combien)

*probabilité = capacité \* opportunité \* motivation*

# Probabilité des risques délibérés

*probabilité = capacité \* opportunité \* motivation*

- On obtient une mesure subjective
  - Repose sur l'expertise
  - Deux experts différents peuvent donner une évaluation différente
  - Contrairement à une probabilité fréquentielle qui peut être mesurée expérimentalement
- Il ne s'agit donc pas d'une valeur « absolue »
  - Une valeur de 0,5 de la probabilité ne veut rien dire
  - Mais cette valeur peut être utilisée pour faire des comparaisons
    - Une valeur de 0,6 représente une « probabilité » plus grande qu'une valeur de 0,4



# Évaluation et choix de contremesures

- Contremesure : Définition
  - Objet (ou processus) qui réduit le risque associé à une menace sur un bien

# Évaluation et choix de contremesures

- Réduction du risque
  - Motivation et impact ne changent pas
  - Réévaluation de capacité et opportunité => risque résiduel
  - réduction = risque initial (sans contremesures) –  
risque résiduel (après application efficace)
- Coût total
  - Coût d'installation (achat, installation, configuration)
  - Coût d'opération (licences, personnel supplémentaire)
  - Impact sur la performance des systèmes
  - Convivialité du système
  - Impact sur le processus d'affaires
  - Introduction de nouveaux risques ...

# Concepts et principes d'opération

- Efficacité des contremesures
  - Sensibilisation du personnel
  - Utilisation réelle des contrôles disponibles
  - Recouvrement des contrôles
  - Vérification administrative
- Principe de l'efficacité
  - Pour que les contremesures soient effectives, elles doivent être utilisées
  - Pour qu'elles soient utilisées, elles doivent être perçues comme étant faciles d'usage, et appropriées aux situations particulières



# Évaluation et choix – Principes fondamentaux

- Principe du point le plus faible
  - Une personne cherchant à pénétrer un système utilisera tous les moyens possibles de pénétration, mais pas nécessairement le plus évident ou celui bénéficiant de la défense la plus solide
- Principe de la protection adéquate (Gestion du risque)
  - La durée de la protection doit correspondre à la période pendant laquelle l'importance et la valeur sont présentes, et pas plus
  - Le niveau et le coût de la protection doivent correspondre à l'importance et à la valeur de ce qu'on veut protéger
- ➔ Choisir la contremesure avec le meilleur rapport « qualité » (réduction de risque) vs. « prix » (coût total)



# Moyens de protection - Types

- Exemples de contre-mesures
  - Chiffrement des données
  - Contrôles au niveau des logiciels
    - Programmés
    - Partie du système d'exploitation
    - Contrôle du développement des logiciels
  - Contrôles du matériel
    - Contrôle de l'accès au matériel: identification et authentification
    - Contrôles physiques: serrures, caméras de sécurité, gardiens, etc...
  - Procédures
    - Qui est autorisé à faire quoi?
    - Changement périodiques des mots de passe
    - Prise de copies de sécurité
    - Formation et administration

Politique  
de sécurité



# Méthodologie d'analyse de risque

1. Identifier la menace
  - Qui ou quoi ?
  - Comment (vulnérabilités) ?
2. Évaluer les risques
  - Probabilité
  - Impact
3. Considérer les mesures de protection par rapport au risque
  - Efficacité (risque résiduel)
  - Coût
  - Difficulté d'utilisation
4. Mettre en place et opérer les mesures protections
  - Modification et/ou installation
  - Changer les politiques
  - Éduquer les utilisateurs
5. Retourner à 1...





# Analyse de risque - Acteurs et responsabilités

- Responsable de sécurité informatique
  - Capacité et Opportunité
    - En analysant
      - Architecture des systèmes existants
      - Vulnérabilités connues et possible des systèmes
      - La nature technique de la menace
        - Outils existants
        - Techniques et méthode d'attaques
        - (Scénario=comment)
    - Probabilité des risques accidentels humains



# Analyse de risque - Acteurs et responsabilités

- « Stakeholders »
  - Description de la menace (quoi)
  - Motivation (qui)
    - Identification des acteurs : compétiteurs, opposants, etc.
    - Analyse d'objectifs et intentions des acteurs : « qu'est-ce qu'ils ont à gagner ? »
  - Impact (et alors)
    - « Combien ça coûterait si... »
    - Relié à la "valeur du remboursement" en assurances
    - Relié au concept d'exposition au risque en comptabilité



# Analyse de risque - Acteurs et responsabilités

- Spécialiste en risque ou en sécurité générale
  - Probabilité de risque accidentel naturel

# Analyse de risque

- Évaluer l'impact
  - Classification des actifs (les biens à protéger)
  - Échelle semi-objective
  - Chiffrage des impacts sur les « objectifs d'affaires »
- Le gestionnaire responsable du processus (propriétaire du système ou « stakeholder » en anglais) est la source de la classification puisqu'il est l'utilisateur du système
  - Ex. : le directeur de la paie est le propriétaire du système informatique qui génère la paie
  - Ex. : le directeur TI est le propriétaire du système informatique qui gère le VPN

# Analyse de risque

- Exemple d'échelle de cotation d'impact
- Échelle arbitraire (aurait pu être différente)
- Toujours conserver la même échelle pour comparer



Cote	Disponibilité/Intégrité	Confidentialité
1	Mineur : courte perte de disponibilité, petite perte monétaire, pertes de peu de données, etc.  (NON CRITIQUE)	Mineur : aucun impact relié si dévoilée à une tierce partie non autorisée  (SANS CLASSIFICATION)
2	Moyen: perte de disponibilité de quelques heures, perte monétaire moyenne, pertes de données peu dommageables etc.  (CRITIQUE)	Moyen: impact grave si dévoilée à un tierce partie non autorisée  (CONFIDENTIEL)
3	Majeur : arrêts de plusieurs jours, perte monétaires de plusieurs mois, pertes d'un large volume de données, etc.  (TRÈS CRITIQUE)	Majeur: impact très grave si dévoilée à une tierce partie non autorisée  (SECRET)
4	Catastrophique: arrêt indéfini, perte de millions de dollars, etc.  (VITAL; « MISSION CRITICAL »)	Catastrophique: impact extrêmement grave si dévoilée a une tierce partie non autorisée  (TRÈS SECRET)

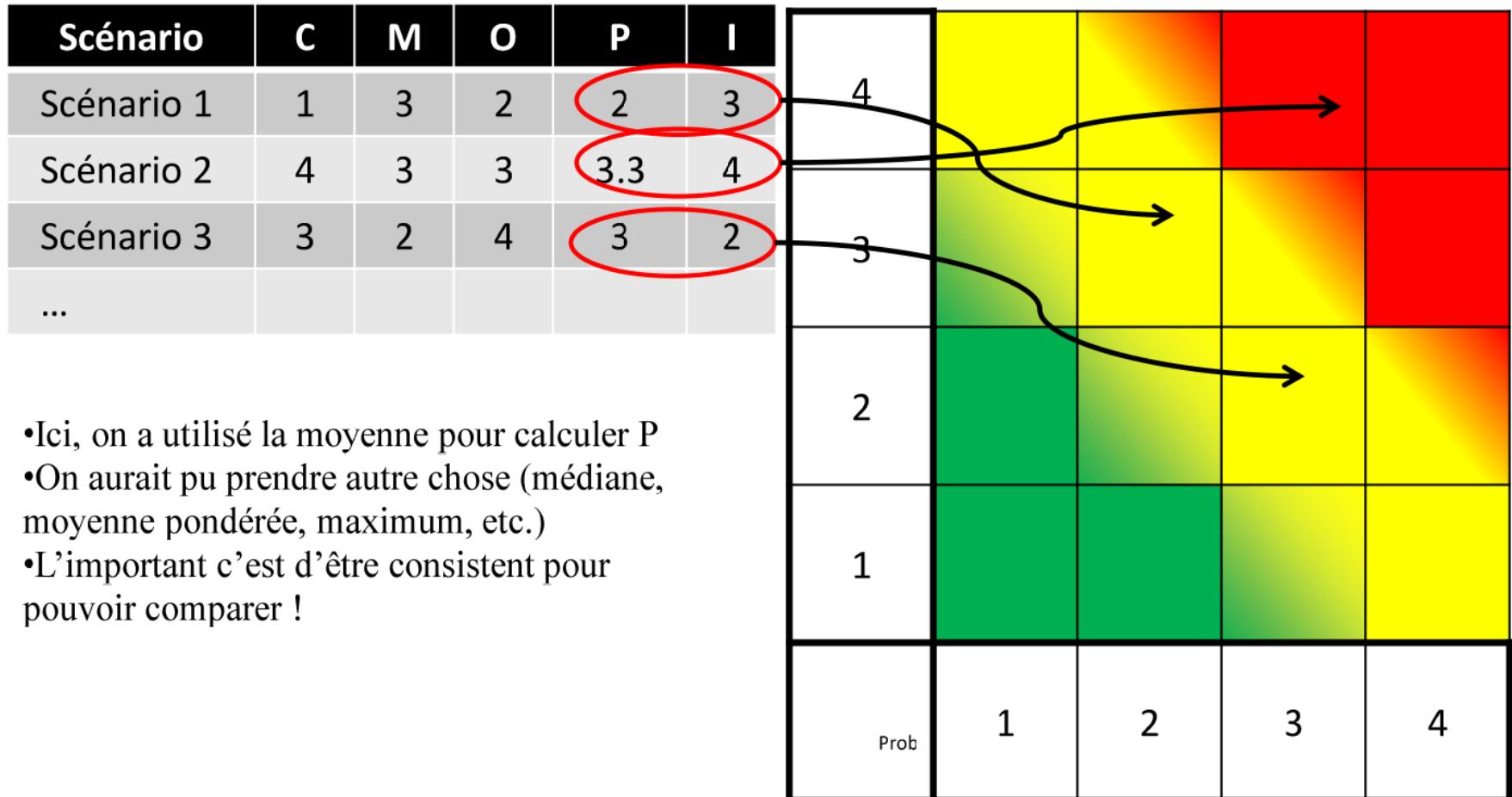


# Analyse de risque

- Évaluer la probabilité
  - Échelle objective pour les aléas (ex. : tables actuarielles)
  - Échelle subjective pour les risques délibérés
  - Chiffre les probabilités d'observer un impact dans un scénario précis



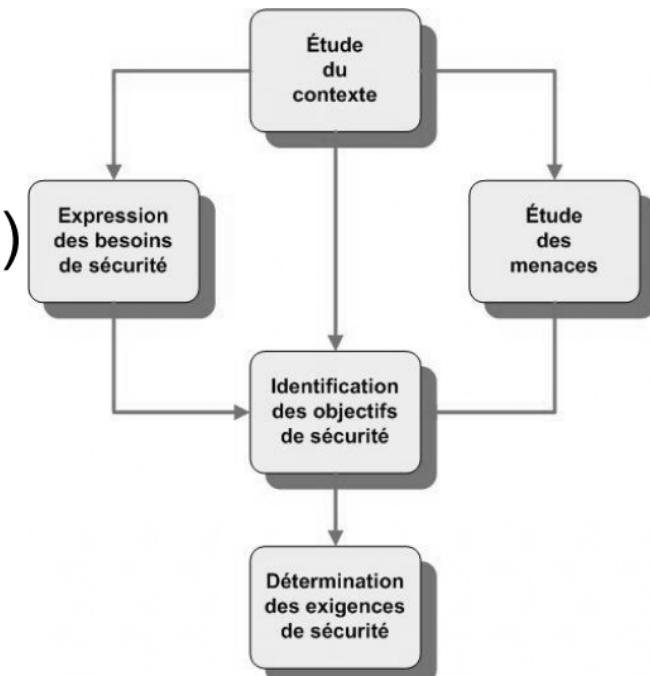
# Analyse de risque



- Ici, on a utilisé la moyenne pour calculer P
- On aurait pu prendre autre chose (médiane, moyenne pondérée, maximum, etc.)
- L'important c'est d'être consistent pour pouvoir comparer !

# Méthodes d'analyse de risque

- Exemples de méthodes d'analyse de risques
  - Méhari Méthode harmonisée d'analyse des risques (MEHARI)
    - CLUSIF (Club de la sécurité de l'information français)
    - CLUSIQ (Québec)
  - EBIOS (Expression des Besoins et Identification des Objectifs de Sécurité) (France)
    - Supportée par l'ANSSI
      - (Agence Nationale de la Sécurité des Systèmes d'Information)
      - Evolution EBIOS RM (Risk Manager)



# Méthodes d'analyse de risque

- Autres méthodes d'analyse de risques
  - CRAMM (Royaume-Uni)
    - Établir les objectifs de sécurité
    - Analyser les risques
    - Identification et sélection de contrôles
  - Octave (Etats-Unis)
    - Operationally critical threat, asset, and vulnerability evaluation
  - FAIR
    - Factor Analysis of Information Risk
    - Méthode reposant sur un taxonomie des facteurs de risques
  - RiskIT/COBIT
  - ...



# Normes ISO 27000

- Panorama des normes ISO 27000
- Famille de normes internationales de sécurité de l'information
- Principales normes

**27001**

- Systèmes de gestion de la sécurité de l'information

**27002**

- Code de bonnes pratiques

27004

- Mesures de gestion de la sécurité

**27005**

- Gestion des risques

**27035**

- Gestion des incidents de sécurité

**27037**

- Traitement des preuves numériques (*forensics*)

...

- ...

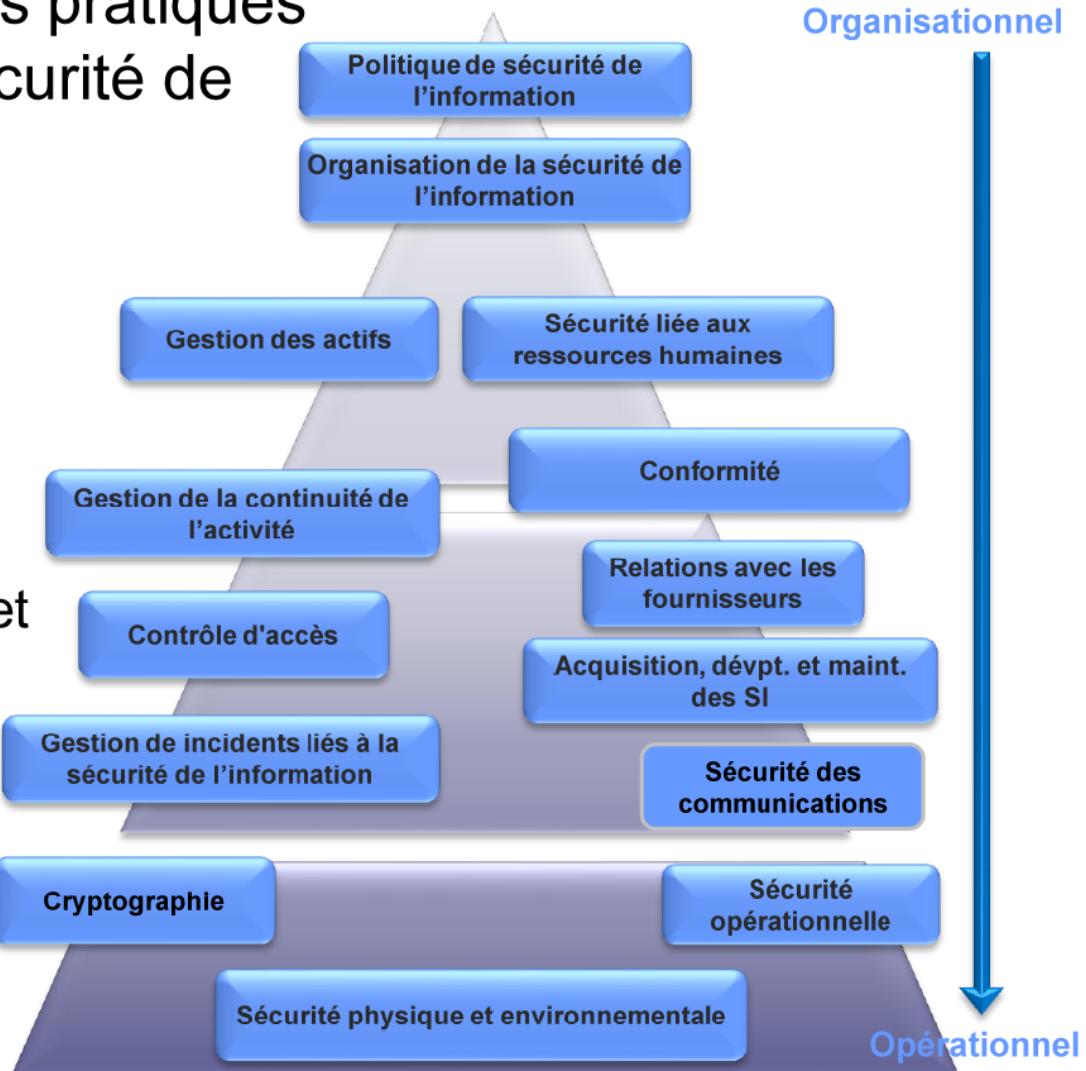


# Normes ISO 27000

- ISO 27001 : Système de Management de la Sécurité de l'Information
  - Certification ISO 27001 délivrée par un organisme certificateur accrédité
    - Démarche calquée sur ISO 9000 (Plan / Do / Check / Act)
    - Audit qui garantit que l'organisation a appliqué les exigences de la norme
    - Certification valable 3 ans, chaque année un audit de contrôle est effectué
    - Certification exigée pour accéder à certains contrats
      - Exemple : organisme payeur d'aides agricoles européennes
    - Pas de niveau minimum de sécurité à atteindre
      - Une entreprise peut donc être certifiée ISO 27001 tout en ayant défini un périmètre réduit et une politique de sécurité peu stricte

# Normes ISO 27000

- 27002 : Code de bonnes pratiques pour la gestion de la sécurité de l'information
- Approche globale de la sécurité des S.I.
- Composée de 114 mesures de sécurité réparties en 14 chapitres couvrant les domaines organisationnels et techniques
- Référentiel de mise en œuvre
  - « Check-list » en cas d'audit





# Normes ISO 27000

- 27002 : Code de bonnes pratiques pour la gestion de la sécurité de l'information
- Exemples de mesures du chapitre « Contrôle d'accès »
  - L'accès aux fichiers/répertoires doit être restreint conformément aux politiques de contrôle d'accès
    - Seuls les professeurs autorisés doivent pouvoir accéder à un répertoire contenant les épreuves des futurs examens/concours
    - Les propriétaires de l'information doivent vérifier les droits d'accès à intervalles réguliers
      - Le responsable des concours doit contrôler les droits d'accès au répertoire contenant les épreuves des futurs examens/concours pour s'assurer qu'il n'y a pas d'étudiants qui auraient été rajoutés
- Exemple de mesures du chapitre « Sécurité opérationnelle »
  - L'installation et la configuration de logiciels doivent être encadrés
    - Seuls les administrateurs doivent pouvoir installés un logiciel sur un poste
  - Des sauvegardes doivent être régulièrement effectuées et testées
    - Un espace de sauvegarde des données peut être mis à disposition des utilisateurs

# Normes ISO 27000

- 27005 : Gestion des risques
- La norme 27005 présente une démarche
  - Donne les lignes directrices relatives à la gestion des risques de sécurité
- Avantages
  - Utilisable seule
  - Plusieurs méthodes sont compatibles ISO 27005
    - Exemple : EBIOS RM
  - Méthode générique, peut être utilisée en toutes circonstances
- Limites
  - C'est plus une démarche qu'une vraie méthode
    - L'organisation doit définir sa propre approche
  - Tendance à l'exhaustivité
  - Accumulation de mesures techniques sans cohérence d'ensemble



# À la prochaine séance

Nora Cuppens



# INF4420a: Sécurité Informatique Cryptographie I

Frédéric Cuppens

*Nora Cuppens & José Fernandez*

# Aperçu du module – Cryptographie

- Définitions et histoire
- Notions de base (théorie de l'information)
- Chiffrement
  - Méthodes « classiques »
  - Chiffrement symétrique
  - Chiffrement à clé publique
- Cryptanalyse de base
- Autres primitives cryptographiques
  - Hachage cryptographique
  - Signature numérique
  - Infrastructure à clé publique (ICP)
- Principes d'applications de la cryptographie
- Risques résiduels d'applications de la cryptographie



# Cryptographie I (aujourd'hui)

- Définition et nomenclature
- Historique
- Théorie de l'information
  - Modèle de Shannon
    - Source d'information
    - Codage et compression
  - Entropie
- Chiffrement
  - Chiffrement et codage
  - Algorithmes « classiques »
- Cryptanalyse de base
  - Force brute
  - Reconnaissance de texte
  - Analyse de fréquences

# CRYPTOGRAPHIE I – INTRODUCTION ET HISTOIRE



POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE



# Définitions et terminologie

- Un peu de grec classique...
  - Kryptos = « caché », « secret »
  - Graphos = écriture
  - ⇒ Cryptographie
  - ⇒ Cryptanalyse
  - Logos = « savoir »
  - ⇒ Cryptologie
  - Stéganos = « couvert », « étanche »
  - ⇒ Stéganographie
- Un peu d'américain...
  - Alice
  - Bob
  - Ève
  - (Charlie)
  - Encrypt and Decrypt
- Un peu de français
  - Chiffrer et déchiffrer
  - Coder et décoder
  - Crypter et décrypter (!)
  - Irène !!! (l'ingénieure)
- Un peu de math...



# Historique

- Les trois ères de la cryptographie
  - « Classique »
    - Jusqu'au masque jetable (chiffre de Vernam)
    - Chiffrement manuel → chiffrement faible
  - « Moderne »
    - Crypto électro-mécanique et WWII (voir applet Enigma)
    - Guerre froide ...
    - Crypto électronique et informatique – DES
    - Chiffrement par machines spécialisées → chiffrement plus complexes
    - Réservés aux organisations pouvant acquérir l'équipement

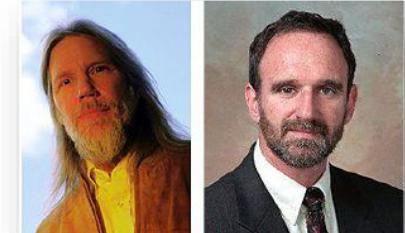


# Historique

- Les trois ères de la cryptographie (suite)

- « Âge d'or »

- Cryptographie à clé publique
    - 1976 - Whitfield Diffie & Martin Hellman
      - Introduise la notion de **cryptographie à clé publique**
      - Algorithme d'échange de clé (DH)
      - Introduise la notion de **signature numérique**
    - 1978 - Ronald Rivest, Adi Shamir, Leonard Adleman
      - Premier algorithme à clé publique (RSA)
    - 1973 – Clifford Cocks
      - Invente en parallèle un algorithme équivalent à RSA au sein du GCHQ
      - L'algorithme est classifié « TOP SECRET »
      - Existence dévoilée seulement en 1997





# Historique

- Les trois ères de la cryptographie (suite)
  - « Âge d'or » (suite)
    - « Démocratisation » de la cryptographie
      - Années 80
        - Cryptographie sur PC (PGP = Pretty Good Privacy))
      - Années 90 et 00
        - Levée des restrictions d'exportations de cryptographie
        - Internet et Web
          - Protocoles réseaux sécurisés : SSH, SSL/TLS, IPSEC, etc.
        - Infrastructures à clé publique et signature numérique
          - Transactions commerciales (bancaire et parabancaires)
          - Identité numérique
          - Cryptomonnaie
          - ...



# Historique

- Les trois ères de la cryptographie (suite)
  - Apocalypse « imminent » et ère post-quantique
    - 1984 – Charles Bennett et Gilles Brassard
      - Invention de la cryptographie quantique –
        - base sa sécurité sur les propriétés de la mécanique quantique
    - 1994 – Peter Shor (suivant les travaux de Dan Simon)
      - Découverte de la cryptanalyse quantique
        - Casse tous les algorithmes à clé publique connus
        - Nécessite d'un ordinateur quantique...
    - Années 10
      - Proposition d'algorithmes à clé publique « post quantiques »
        - Semblent résister à la cryptanalyse quantique
        - Peu pratiques à utiliser
        - Pas (encore) de standard établi
        - Adoption très lente...



# CRYPTOGRAPHIE I – THÉORIE DE L'INFORMATION – MODÈLE DE SHANNON



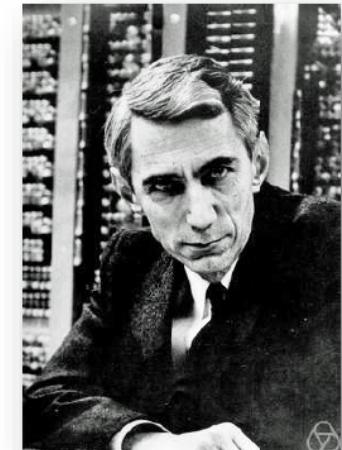
POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE



# Claude Shannon

- IIe guerre mondiale
  - Contribue aux efforts de cryptanalyse de guerre
- Père de la Théorie de l'information
  - 1948 – « A Mathematical Theory of Information »
- Fondement théorique de la cryptographie
  - 1945 – « A Mathematical Theory of Cryptography »
    - Classifié – basée sur ses travaux de cryptanalyse
  - 1949 – « Communication Theory of Secrecy Systems »
    - Version non-classifiée, publié dans Bell Technical Journal





# Claude Shannon

- Contributions
  - Introduit une définition mathématique de l'information
    - Source d'information
    - Modèle de Shannon – transmission d'information
  - Introduit la notion d'entropie (dans le contexte de l'information)
    - Définit le bit comme unité de mesure de l'information
    - Établit les limites fondamentales de la compression
    - Capacité maximale d'un canal de transmission (sans bruit)
    - Introduit une notion mathématique du bruit
      - Établit les limites fondamentales des codes correcteur d'erreurs
  - Introduit une théorie du « secret » en information
    - Modèle de Shannon révisé – transmission d'informations secrètes
      - Décrit le lien entre codage et chiffrement



# Théorie de l'information

- « Information »
  - Valeur instantanée d'une variable aléatoire qui est transmise vers un récepteur à travers un canal de communication
- Concepts importants
  - Variable aléatoire
  - Canal de communication – Transmission
    - ou
  - Moyen de stockage
- Exemples
  - La couleur du ciel (variable aléatoire) transmise via les ondes lumineuses (canal de transmission) vers votre œil (récepteur)
  - Contenu d'un fichier (variable aléatoire) transmise via le réseau téléphonique (canal de transmission) vers votre collègue (récepteur)

# Théorie de l'information

- L'information est un concept abstrait
  - la valeur de l'information dépend des attentes du récepteur
    - Couleur du ciel : est-ce vraiment une information ?
    - (théorie de la décision) Est-ce que la température du soleil est critique à ma décision d'investir dans une entreprise web ?
- Théorie de l'information (« Communication Theory »)
  - Ne s'intéresse pas à la sémantique de l'information (son « sens »)
  - S'intéresse à la quantité d'information qui manque au récepteur
    - Wheeler
      - *« information » in communication theory is not related to what you do say, but to what you could say*
  - Information = manque de connaissance du récepteur



# Théorie de l'information

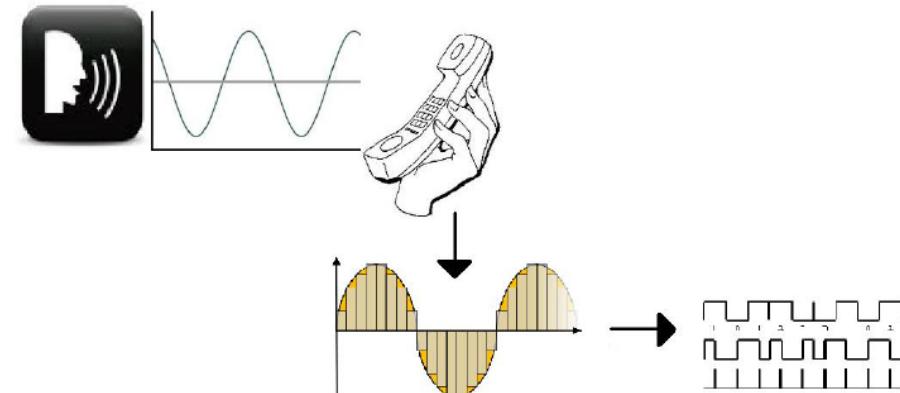
- Pour mesurer cette méconnaissance
  - Quantité d'information obtenue par observation directe de l'information obtenue/transmise
    - Représentée par la valeur de la variable aléatoire
    - Plus cette valeur est « aléatoire »
      - Plus grande est la méconnaissance du récepteur **avant** sa transmission
      - Plus sa transmission « ajoute » de l'information
    - Si cette valeur est peu aléatoire ou déterministe
      - Le récepteur a peu d'incertitude sur la valeur (méconnaissance faible)
      - La transmission n'apprend pas grand-chose au récepteur (valeur information de l'information faible)
  - Mesure mathématique d'information
    - Entropie de la variable aléatoire
  - Unité de mesure
    - Généralement le *bit*
      - Défini ainsi pour faciliter la représentation et calculs mathématiques



# Théorie de l'information

- **Source d'information**

- « Boîte noire »
- Produit des symboles
  - selon un processus stochastique
  - seront codés (transformés)
  - seront stockés ou transmis
- Variable aléatoire
  - associée au symbole produit
- Série de symboles
  - différente à chaque fois (« réinitialisation »)
  - produite selon le même processus stochastique



- **Source discrète**

- un symbole à la fois
- sur demande (« bouton »)

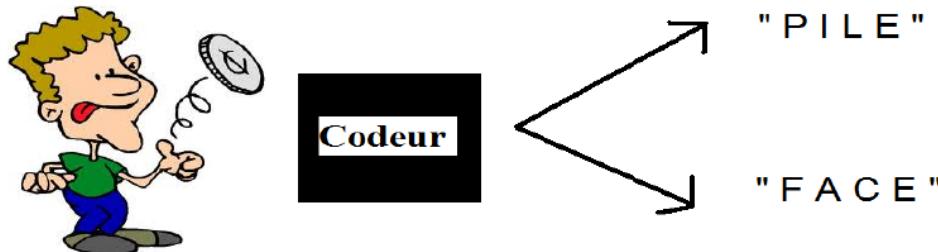


« a », « z », « θ », ...



# Théorie de l'information

- Transmission/stockage de l'information
  - La source produit de l'information « pure » sous forme abstraite
  - Ne peut pas être transmise ou stockée dans cet état « pur »
  - Doit avoir une représentation physique pour être transportée/stockée
- Codage
  - Processus de transformation de l'information
  - S'adapte au canal de transmission ou moyen de stockage
  - Permet au récepteur de reconvertisir (décoder) l'information dans une forme intelligible (même forme qu'à la source)



- Codage ≠ chiffrement
  - Le codage ne protège pas la confidentialité de l'information



# Théorie de l'information

- Compression et codage
  - Le codage peut permettre de faire de la compression
    - Moins de symboles utilisés dans la transmission/stockage que par la source
  - 1<sup>er</sup> théorème de Shannon (voir plus loin)
    - Établit limite de la compression sans perte d'information (*lossless compression*)
      - Codes de Huffman
      - Lempel-Ziv-Welch (LZW)
    - Ne s'applique pas à la compression avec perte (*lossy compression*)
      - MP3
      - JPEG
      - MPEG

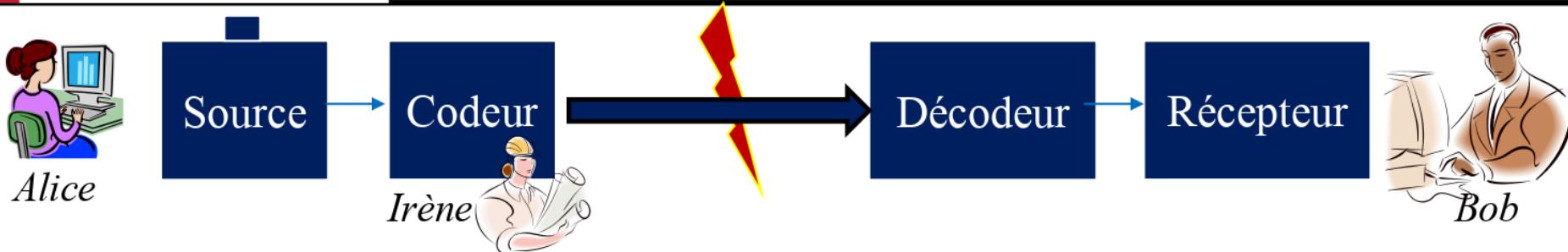


# Théorie de l'information

- Les composants que nous avons évoqués sont du côté de la source
  - On fait l'image miroir pour avoir les composants du côté du récepteur
  - Source – codeur  $\Rightarrow$  décodeur – récepteur
- On peut alors créer un modèle mathématique plus formel
  - ➔ le modèle de Shannon



# Modèle de Shannon



- **Source**
  - Produit des symboles d'un "alphabet" (  $\Sigma$  )
  - Fonctionne "sur demande" (d'où le "bouton")
- **Codage**
  - Regroupe et transforme les symboles de la source dans un format pouvant être transmis ou sauvegardé
- **Canal**
  - Peut introduire du bruit
    - symbole reçu  $\neq$  symbole transmis
- **Décodage**
  - Permet de reconstruire le message original
    - séquence des symboles de source



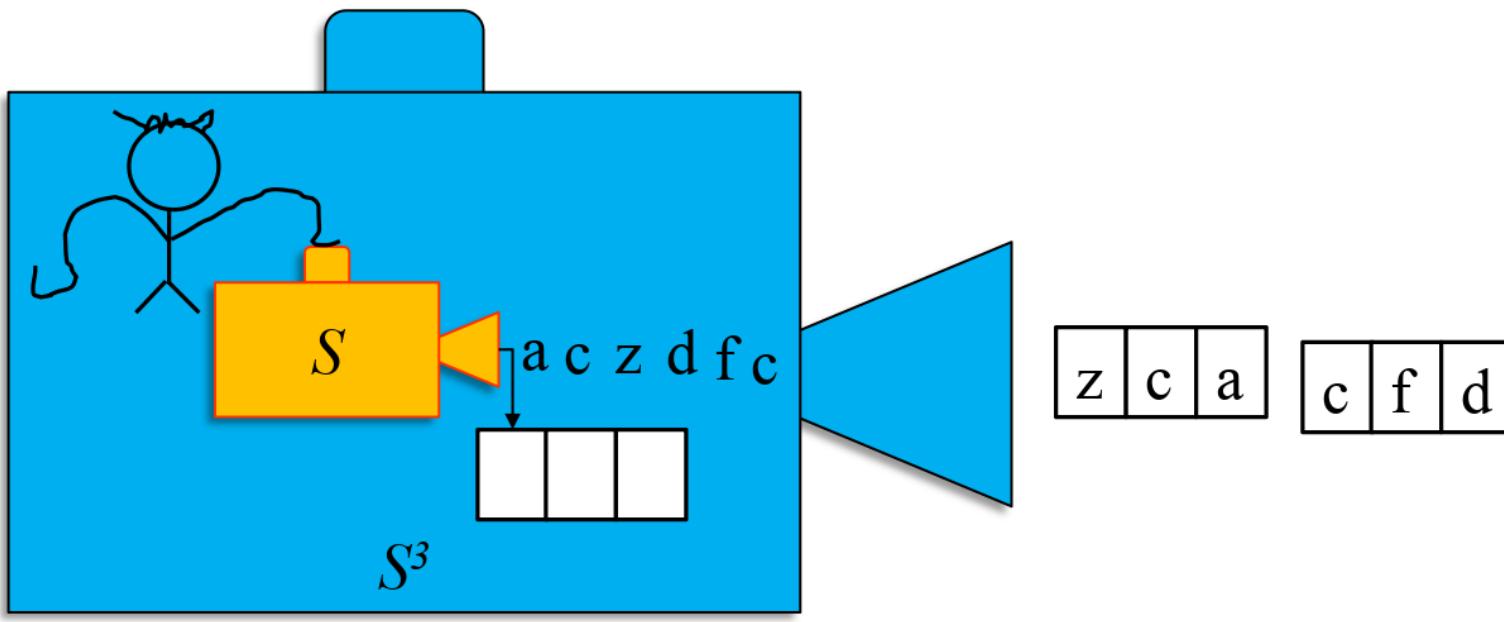
# Source d'informations

- Alphabet
  - Ensemble discret fini  $\Sigma = \{\sigma_1, \dots, \sigma_M\}$
  - Par convention taille de  $\Sigma$ ,  $|\Sigma| = M$
- Contrôle
  - Un "bouton" qui permet d'obtenir un symbole à la fois
- Principe de la boîte noire
  - Autre que le bouton et un nombre petit d'observations (symboles), on ne peut rien savoir sur le contenu ou fonctionnement de la source (sauf peut-être Alice, mais pas Ève, Irène ou Bob)
- Pourquoi cette abstraction ??
  - Permet de discuter de l'efficacité du codage (théorie de l'information)
  - Permet d'analyser correctement la résistance à certaines menaces
    - Algorithmes de chiffrement
    - Choix de mots de passe et phrases de passe
    - ...



# Sources dérivées

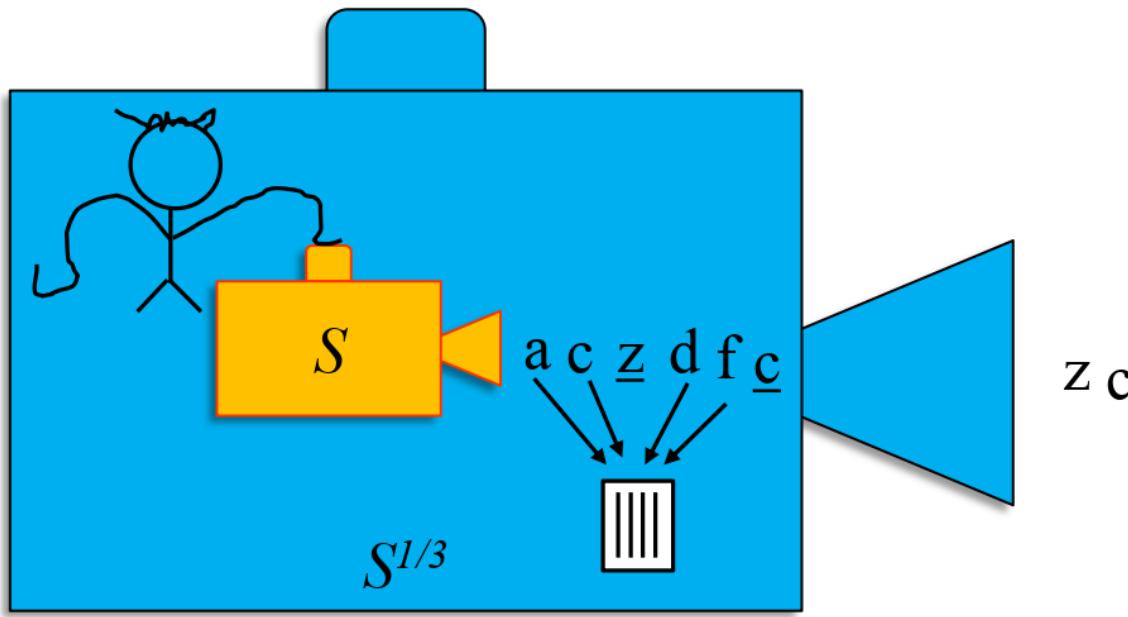
- Source par bloc
  - Étant donné une source  $S$ , et un entier positif  $b$
  - $S^b$  représente la source obtenue en encapsulant  $S$  par une boîte
    - qui mets  $b$  symboles de  $S$  dans un tampon (« buffer ») avant de les sortir
  - Noter que l'alphabet de  $S^b$  est maintenant  $\Sigma^b$





# Sources dérivées

- Source par échantillonnage
  - Étant donné une source  $S$ , et un entier positif  $b$ ,
  - $S^{1/b}$  représente la source obtenue en encapsulant  $S$  par une boîte
    - qui émet seulement le 1er symbole de chaque  $b$  symboles sortie de  $S$
  - L'alphabet de  $S^{1/b}$  est le même que  $S$ , soit  $\Sigma$



# Types de source d'information

- Déterministe
  - La boîte « connaît » à l'avance toute la séquence de symboles (potentiellement infinie...)
- Probabiliste
  - La boîte choisit les symboles au fur et à mesure selon une distribution de probabilité
    - **Processus markovien ou "sans mémoire"**
      - $p_i = \text{Prob}(S \Rightarrow \sigma_i), \forall 1 < i < M$
      - e.g.  $\text{Prob}(S^b \Rightarrow \sigma_i, \sigma_j) = p_i p_j$
    - **Processus non-markovien**
      - Les probabilités de symboles peuvent dépendre des symboles antérieurs sortis de la source...



# Codage

- Translittération
    - Un codage traduit les symboles de source vers un autre « alphabet »  $T = \{\tau_1, \dots, \tau_N\}$ , (*Tau majuscule*)
  - Fonction de codage
    - $F : \Sigma \rightarrow T$ ,
      - $\tau = F(\sigma)$ , représente comment le symbole  $\sigma$  devra être transmis
  - Fonction de décodage
    - $F^{-1} : T \rightarrow \Sigma$ 
      - $\sigma' = F^{-1}(\tau')$ ,
        - Si  $\tau' \neq \tau$  alors  $\sigma' \neq \sigma$   
il y a eu erreur de transmission (bruit dans le canal)
        - Si  $\tau' = \tau$  alors  $\sigma' = \sigma$   
transmission sans erreur
      - Bob reçoit ce que Alice (source) a émis
- F est nécessairement une injection

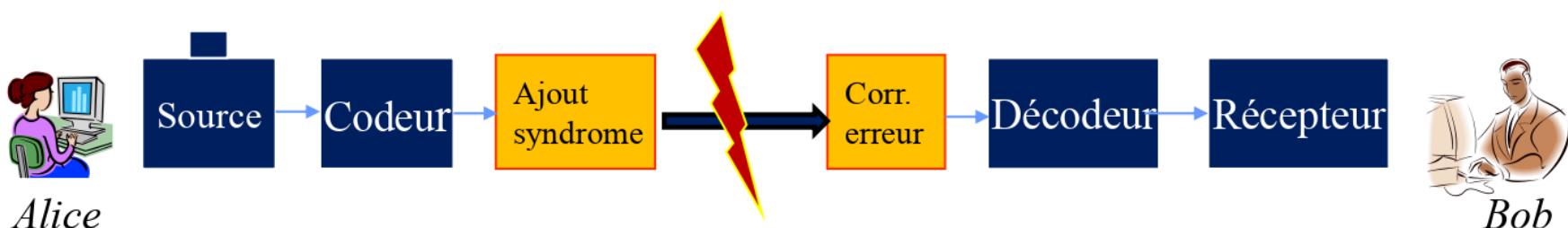


# Correction d'erreur

- Code correcteur d'erreur
  - L'introduction de bruit dans le canal est compensé en utilisant un code correcteur d'erreur dans le codage t.q.  
 $\text{Prob } (F^{-1}(\tau') = \tau) \rightarrow 1$ , où  $\tau'$  est le symbole reçu via le canal
- L'efficacité du code correcteur d'erreur
  - Dépend du niveau de bruit introduit par le canal
    - celui-ci peut être mesuré avec l'entropie de Shannon
  - Se mesure également en nombre de bits nécessaires par symbole de source, pour un code qui corrige « presque toutes les erreurs »
- 2e Théorème de Shannon
  - Établit le lien entre l'efficacité du code correcteur d'erreur et le niveau de bruit du canal

# Correction d'erreur

- En pratique, la correction d'erreur est souvent une étape distincte et séparée du codage
  - Chez Alice
    - Codage supplémentaire après codage initial
    - Ajout d'information supplémentaire (*syndrome*)
  - Chez Bob
    - Décodage initial avant décodage final
    - Analyse du syndrome et du message
      - permet de corriger les erreurs (avec haute probabilité)



# CRYPTOGRAPHIE I – THÉORIE DE L'INFORMATION – ENTROPIE



POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE

# Efficacité du codage - Compression

- Compression
  - Dans certaines circonstances, on voudrait pouvoir coder en utilisant moins de bande passante, p.ex. tel que  $N < M$
  - Efficacité du code
    - est mesurée en bits transmis par chaque symbole de source émis
  - 1er Théorème de Shannon
    - Efficacité maximum d'un code compresseur est approximativement égale à  $H(S)$
    - Il existe un code compresseur (sans erreur) avec efficacité  $H(S) + 1$
  - Qu'est-ce «  $H(S)$  » → L'entropie de la source  $S$



# Entropie de Shannon

- Définitions

- $H(S) = \sum_i p_i \log_2 1/p_i$



- Propriétés

- Fonction convexe

- $\Sigma = \{0,1\}$

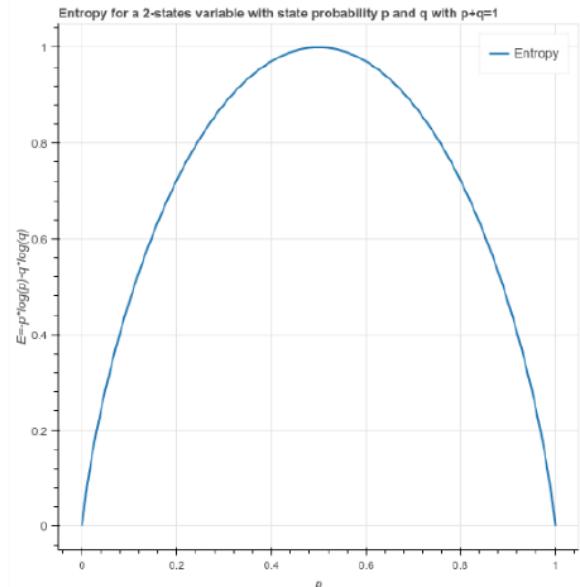
- Prob ( $S="0"$ ) =  $p$ , Prob ( $S="1"$ ) =  $q = 1-p$
    - Valeur minimale

- Prob ( $S="0"$ ) = 1; Prob ( $S="1"$ ) = 0
    - ➔  $H(S) = 0$  bit

- Valeur maximale
    - Prob ( $S="0"$ ) = Prob ( $S="1"$ ) =  $1/2$
    - ➔  $H(S) = 1$  bit

- $\Sigma$  arbitraire,  $|\Sigma| = N$

- Valeur minimale
      - Prob ( $S = \sigma$ )=1 pour un  $\sigma$  donné, Prob ( $S = \sigma$ ) = 0 pour tous les autres
      - ➔  $H(S) = 0$  bit
- Valeur maximale
      - Prob ( $S = \sigma_i$ ) = Prob ( $S = \sigma_j$ ),  $\forall \sigma_i, \sigma_j \in \Sigma$
      - ➔  $H(S) = \log_2 N$  bit





# Entropie

- Exemple de calcul d'entropie

- Pile ou face

- Alphabet {pile, face}
    - Probabilité d'occurrence des symboles ( $p_i$ ): chaque symbole équiprobable avec une probabilité de  $\frac{1}{2}$
    - $H(S) = \sum_i p_i \log_2 1/p_i$ 
      - pile :  $\frac{1}{2} \log_2 (1 / \frac{1}{2}) = \frac{1}{2} \log_2 2 = \frac{1}{2} * 1 = \frac{1}{2}$
      - face :  $\frac{1}{2} \log_2 (1 / \frac{1}{2}) = \frac{1}{2} \log_2 2 = \frac{1}{2} * 1 = \frac{1}{2}$
      - $H(S) = \frac{1}{2} + \frac{1}{2} = 1 \text{ bit}$

- Alphabet équiprobable

- Alphabet = {a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z}
    - Probabilité d'occurrence des symboles ( $p_i$ ):
      - chaque symbole équiprobable avec une probabilité de  $1/26$
    - $H(S) = \sum 1/26 \log_2 1/(1/26) = 26 * 1/26 * \log_{10}(26)/\log_{10} 2 =$   
 $= \log_{10}(26)/\log_{10} 2 = 4.7 \text{ bits}$  (on peut vérifier :  $2^{4.7} = 25.99$ )



# Analyse fréquentielle vs. entropie

- Problème
  - L'entropie de Shannon
    - est définie à partir de *probabilités*
    - s'applique seulement aux sources markoviennes
  - Comment calculer/utiliser l'entropie sur
    - des sources non-markoviennes ?
    - des textes/séquences finies de symboles ?
- « Solution »
  - Fréquence de symbole
    - Soit  $S_N = s_1, s_2, \dots, s_N$ ,  $s_i \in \Sigma$ , une séquence d'une source  $S$ , on définit:  
$$f_i(S_N) = \frac{|\{j \mid s_j = \sigma_i\}|}{N}$$
  - Pseudo-entropie
    - Définie/calculée à partir des fréquences (au lieu de probabilités)



# Pseudo-entropie

- Pour une séquence finie  $S_N$   $\Psi(S_N) = \sum_i f_i(S_N) \log \frac{1}{f_i(S_N)}$
- Pour une séquence  $S$   $\Psi(S) = \lim_{N \rightarrow \infty} \Psi(S_N)$
- Pour une source d'information quelconque  $S$ 
  - A chaque fois qu'on utilise la source « N fois »
  - On obtient une séquence  $S_N$  différente de longueur N
  - On calcule la pseudo entropie  $\Psi(S_N)$  de cette séquence, qui est elle-même une variable aléatoire
  - Sa valeur espérée  $\overline{\Psi(S_N)}$  représente une pseudo-entropie de la source sur des séquences de longueur N
- On considère alors la pseudo-entropie de la source comme étant la limite de cette valeur espérée

$$\Psi(S) = \lim_{N \rightarrow \infty} \overline{\Psi(S_N)}$$



# Entropie vs. pseudo-entropie

- Pour les sources markoviennes
  - La pseudo-entropie d'une séquence générée par la source va s'approcher de l'entropie
  - Cette convergence est bonne lorsque la taille de la sous-séquence est grande, parce que les fréquences  $f_i$  s'approchent des probabilités  $p_i$  (loi des grands nombres)
    - Quand  $N \rightarrow \infty$ , alors  $\Psi(S_N) \rightarrow H(S)$
    - Si  $N$  est trop petit, alors
      - déductions faites à partir des  $f_i$  non valable statistiquement  
→ cryptanalyse difficile (voir TP 1)
- Pour les sources non-markoviennes
  - L'entropie  $H(S)$  n'est pas vraiment définie,
    - On utilise  $\Psi(S)$  à la place (outil de calcul d'entropie TP1)
    - On écrira dans le reste du cours «  $H(S)$  »,  
mais on veut vraiment dire  $\Psi(S)$  ...



# Interprétation de l'entropie d'une source

- Interprétation de  $H(S)$ 
  - 1<sup>er</sup> théorème : Chaque symbole émis par  $S$  peut être codé individuellement avec en moyenne  $H(S)$  bits
  - Et si on permet que le codage regroupe 2 lettres à la fois ?
    - ➔ Par 1<sup>er</sup> théorème on peut coder chaque digramme (2 symboles) avec  $H(S^2)$  bits, soit  $H(S^2)/2$  bits par symbole
    - ➔ Mais si  $H(S^2)/2 \leq H(S)$ , donc on peut avoir un gain en compression
- Taux de compression
  - Sans compression
    - ➔  $\log N$  bits par symbole, dans le pire cas (entropie maximale)
  - Avec compression par bloc de  $b$  symbole
    - ➔  $H(S^b)/b$  bits par symbole
  - Taux de compression = 
$$\frac{H(S^b)/b}{\log N}$$



# Source markovienne vs. non markoviennes

- Source markovienne
  - Si  $S$  est markovienne, alors  $H(S^b) = b^*H(S)$ 
    - Conséquence: Aucun gain de compression en codant par bloc
    - Intuition: Il n'existe pas de corrélation entre les symboles (distribution de probabilité indépendante), et chaque symbole doit être codé individuellement
- Source non markovienne
  - En général  $H(S^b) \leq b^*H(S)$ 
    - Conséquence1: Il y a en général un gain de compression en codant par bloc
    - Intuition:
      - Les probabilités des symboles dépendent des symboles antérieurs
      - Cette « dépendance » statistique peut être exploitée par le codage pour réduire le nombre de symboles ou le nombre de bits dans leur codage
  - P.ex. en français
    - la lettre « u » suit (presque toujours) la lettre « q »
    - Un sujet est suivi d'un verbe, p.ex. « Je\_ » doit être suivi d'un verbe conjugué à la 1<sup>e</sup> personne du singulier
    - Le gain de compression devrait augmenter en considérant des tailles de blocs plus grandes

# Entropie du langage de la source

- En théorie,
  - plus la taille de bloc  $b$  est grande,  
plus le taux de compression est élevé (jusqu'à une certaine limite)
- Langage associé à une source  $S$ 
  - ensembles de chaînes finies générées par  $S$
- L'entropie  $H_L$  du *langage* associé à la source  $S$ ,

$$H_L(S) = \lim_{b \rightarrow \infty} \frac{H(S^b)}{b}$$

- est le minimum de bits nécessaires (en moyenne) pour coder chaque symbole de chaînes émises par  $S$ , même si on permet de coder avec des tailles de blocs arbitraires
- représente la limite ultime de compression

# CRYPTOGRAPHIE I – MODÈLE DE SHANNON RÉVISÉ

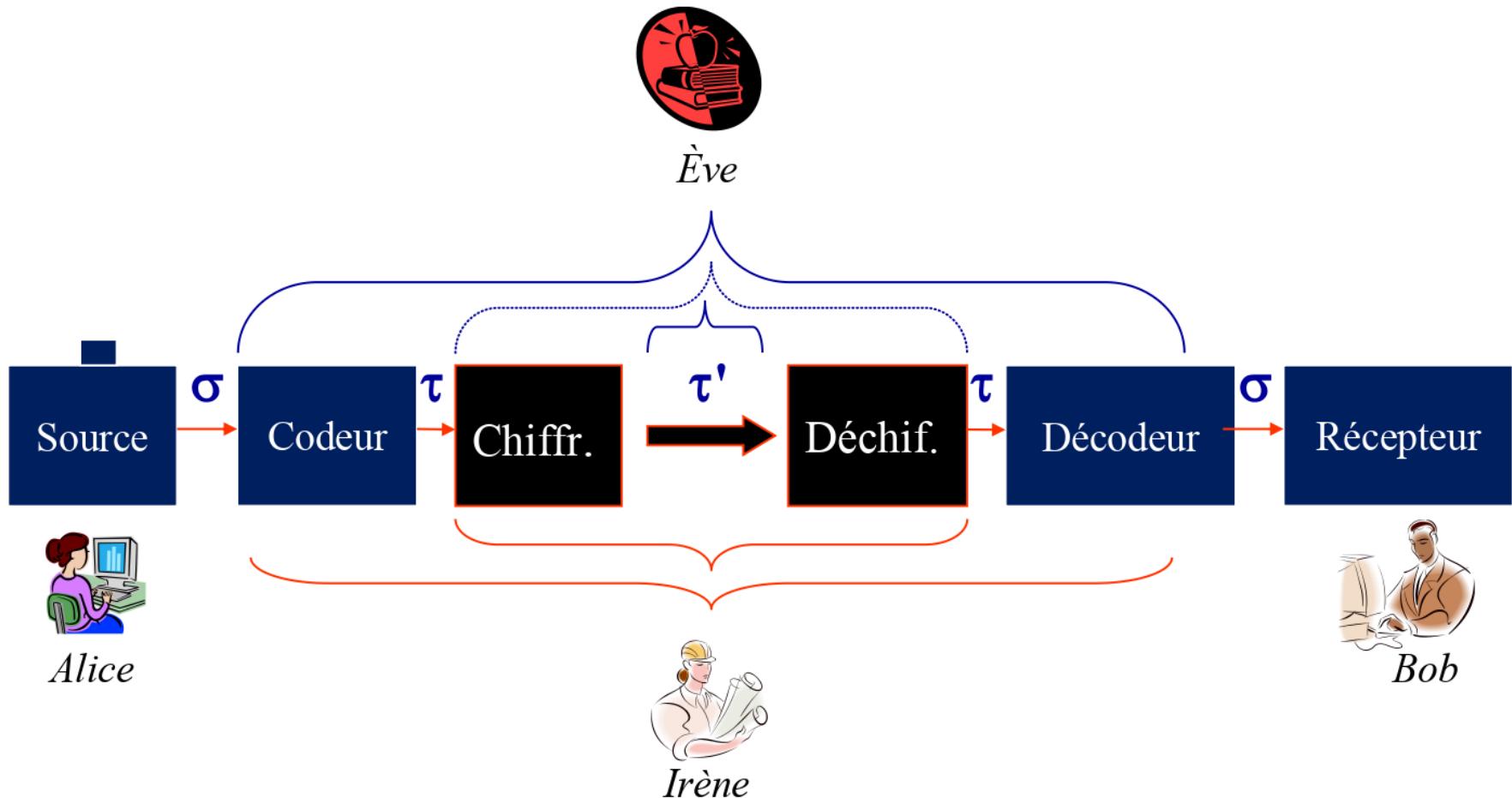


POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE



# Modèle de Shannon révisé





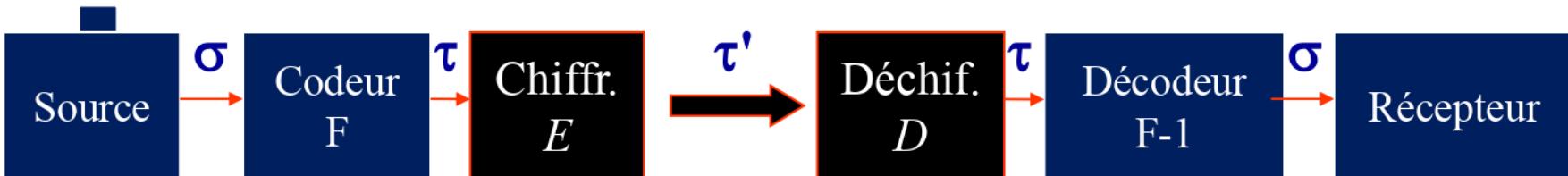
# Modèle de Shannon révisé

- Ève
  - Peut intercepter impunément tous les mots de codes  $\tau$  transmis sur le canal
- Irène
  - Choisit l'algorithme de chiffrement
  - Détermine la politique de choix et gestion de clés
  - Doit tenir en compte le codage
    - en considérant les caractéristiques de la source (DP, entropie, etc.)
    - en influençant le choix de codage (si possible)
    - en choisissant et adaptant l'algorithme de chiffrement en conséquence (choix de taille de clés, compression/décompression, etc.)
  - Pourquoi : voir TP 1...



# Algorithme de chiffrement – Concepts généraux

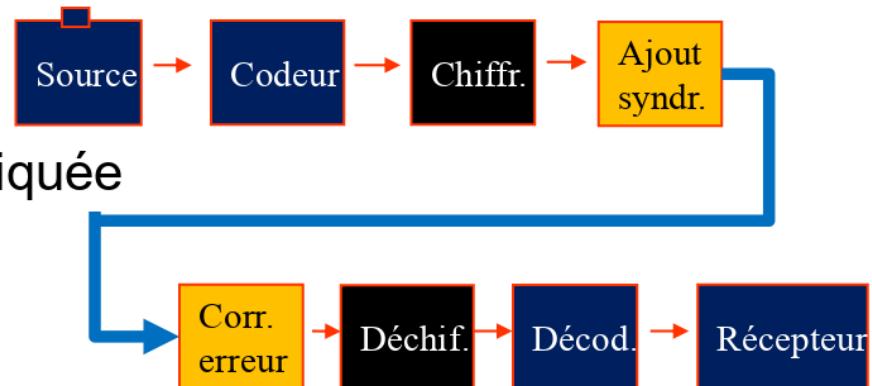
- Alphabet
  - Entrée :  $T$
  - Sortie : en général  $T$ , mais peut-être un autre alphabet  $T'$
- Fonction de chiffrement
  - Clé de chiffrement =  $k_e$
  - $\tau' = E(k_e, \tau) = E_{k_e}(\tau)$
- Fonction de déchiffrement
  - Clé de déchiffrement =  $k_d$
  - $\tau = D(k_d, \tau') = D_{k_d}(\tau)$



# Cryptographie et correction d'erreurs

- Ne corrige pas les erreurs

- Il faut donc que
  - $\tau = \tau'$  (pas de bruit), ou que
  - la correction d'erreur soit appliquée
    - après le chiffrement et
    - avant le déchiffrement



- La correction d'erreur

- Constitue une forme de protection de l'intégrité des messages
  - Protège contre des erreurs aléatoires (menace accidentelle)
    - P.ex. erreur de transmission due au bruit, interférence accidentelle, etc.
  - Ne protège pas contre la menace délibérée
    - P.ex. des erreurs introduites de façon « intelligente » par un acteur malveillant ayant accès au canal (Ève !)

# CRYPTOGRAPHIE I – CRYPTOGRAPHIE CLASSIQUE



POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE



# Algorithmes "classiques" mono-alphabétiques

- Algorithme de César
  - Source
    - texte en caractères latin
  - Codage
    - lettres → chiffres de 1 à 26  
(20 pour être historiquement exact)
  - Chiffrement
    - $x \rightarrow x+3 \bmod 26$
  - Clés
    - nil
- Algorithme de décalage
  - Source et codage
    - idem
  - Chiffrement
    - $x \rightarrow x + k \bmod 26$
  - Clés
    - $k \in \{1, \dots, 26\}$
- Algorithme de substitution
  - Source
    - Idem
  - Codage
    - aucun
  - Chiffrement
    - $x \rightarrow \pi(x)$
  - Clé
    - $\pi$  (une table de substitution)
- Algorithme afin
  - Source et codage
    - lettres en chiffres
  - Chiffrement
    - $x \rightarrow ax + b \bmod 26$
  - Clé
    - $(a, b)$  où  $a, b \in \{1, \dots, 26\}$



# Algorithmes classiques

- Algorithme de substitution
  - On prend un texte en clair et, pour chacune des lettres du texte, on utilise la lettre comme index dans une table de substitution ( $\pi$ ) pour trouver l'équivalent chiffré
  - La table de substitution représente la clé
  - « **H**ELLOWORLD » devient :

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
h	e	v	a	m	t	s	i	c	f	n	u	o	r	B	g	q	w	j	y	d	l	x	k	z	p

$\pi$

– « **0**imuubxbwua »



# Algorithmes classiques

- Algorithme de substitution
  - Avec la même clé et plus de texte

I	L	E	T	A	I	T	U	N	E	F	O	I	S	L	H	I	S	T	O	I	R	E	D	U	N	P	E	T	I	T	C	H	A	P	E	R	O	N	R	O	U	G	
c	u	m	y	h	c	y	d	r	m	t	b	c	j	u	i	c	j	y	b	:	w	m	a	d	r	g	m	y	c	y	v	i	h	g	m	w	b	r	w	b	d	s	m

- On remarque qu'il est difficile de faire la correspondance entre le texte original et le texte chiffré sans connaître la table de substitution  $\pi$  (la clé)
- Sans le texte en clair, il serait aussi difficile d'inférer  $\pi$  à partir du texte chiffré et il est nécessaire d'obtenir un « grand » (au moins une occurrence de 25 lettres sur 26) nombre de texte en clair pour reconstruire la clé
- L'algorithme classique de substitution possède donc des propriétés raisonnables de confusion



# Algorithmes classiques

- Algorithme de transposition (« bit shifting »)
  - On prend un texte en clair et on permute la position des lettres (ou des bits dans le cas moderne) entre elles en fonction d'une « clé »
  - Équivalent du jeu Charivari (allez voir sur Internet...)
  - Dans l'antiquité on utilisait un bâton autour duquel on enroulait une lanière de cuir (déguisée en ceinture) où était écrit le texte
- Exemple
  - Avec un « bâton » qui a une épaisseur de deux lettres, « HELLOWORLD » devient

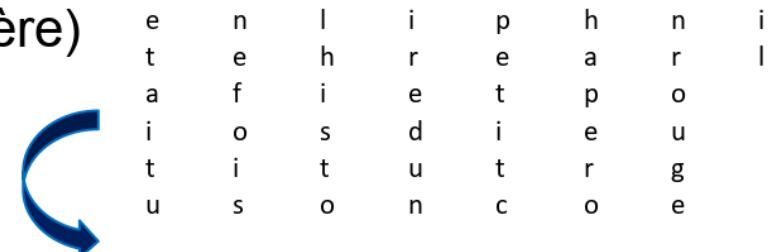
h		o	o	l
e		w	r	d

- « hloolelwrd »



# Algorithmes classiques

- Algorithme de transposition (« bit shifting »)
  - Chiffrons ce texte avec un « bâton » de taille 6 et commençons au 3<sup>e</sup> « trou de ceinture » (3<sup>e</sup> caractère)



e	n	l	i	p	h	n	i
t	e	h	r	e	a	r	l
a	f	i	e	t	p	o	
i	o	s	d	i	e	u	
t	i	t	u	t	r	g	
u	s	o	n	c	o	e	

I	L	E	T	A	I	T	U	N	E	F	O	I	S	L	H	I	S	T	O	I	R	E	D	U	N	P	E	T	I	T	C	H	A	P	E	R	O	N	R	O	U	G					
e	n	l	i	p	h	n	i	t	e	h	r	e	a	r	l	a	f	i	e	t	p	o	i	s	d	i	u	t	r	g	u	s	o	n	c	o	e	o	u	g	u	s	o	n	c	o	e

- Ici, il est « facile » d’inférer le texte original à partir du texte chiffré
- On peut « facilement » retrouver la clé à partir du texte chiffré
- La confusion est donc mauvaise
- Par contre, la disparition d’une lettre entraîne la modification de tout le texte chiffré qui suit
- La transposition amène donc une diffusion raisonnable



# Algorithme de Vigenère

- Algorithme de Vigenère
  - Source
    - Texte en caractères latin
  - Codage
    - lettres → chiffres de 1 à 26
  - Clé
    - $K = k_1 k_2 \dots k_m$ , mot/phrase de longueur  $m$
  - Chiffrement
    - $x_i \rightarrow (x_i + k_{i \bmod m}) \bmod 26$



*La trahison des images*, René Magritte 1929

C	E	C	I	N	E	S	T	P	A	U	N	E	P	I	P	E	...
S	E	X	Y	S	E	X	Y	S	E	X	Y	S	E	X	Y	S	...
3	5	3	9	14	5	19	20	16	1	21	14	5	16	9	16	5	...
+ 19	5	24	25	19	5	24	25	19	5	24	25	19	5	24	25	19	...
22	10	1	8	7	10	17	19	9	6	19	13	24	21	7	15	24	...
V	J	A	H	G	J	Q	S	I	F	S	M	N	U	G	O	X	...



# Masque jetable

- Connu sous le nom de « One-time Pad »
- Historique
  - Inventé par le capitaine Vernam (US Army Signal Corps) en 1919
  - Utilisée pour le Téléphone Rouge entre Moscou et Washington (guerre froide)
  - Utilisée par Che Guevara en Bolivie
- Fonctionnement
  - $\Sigma = T = \{0,1\}$
  - Algorithme : XOR bit-à-bit du message et de la clé
  - Clé
    - En « théorie »
      - chaîne de bits aléatoires, de longueur “infinie”
      - distribuée à l'avance (physiquement, etc.)
      - mauvaise diffusion et confusion, mais pourtant...
    - Seul algorithme avec « sécurité parfaite » (Shannon)
  - En « pratique »
    - chaîne de bits générée par un algorithme déterministe
      - Dépendant des messages/clés antérieurs
      - Générateur de nombres pseudo aléatoires (avec une « semence »)
    - au moins aussi longue que le message (pas de recyclage de clé)



# Confusion et diffusion

- Même dans les algorithmes modernes, la confusion et la diffusion sont deux propriétés recherchées
- Au sens strict
  - Confusion : propriété de rendre la relation entre la clé de chiffrement et le texte chiffré la plus complexe possible
  - Diffusion : propriété où la redondance statistique dans un texte en clair est dissipée dans les statistiques du texte chiffré
  - On remarque que les algorithmes classiques ne respectent pas tout à fait cette propriété
- Objectif
  - Empêcher de retrouver la clé à partir de paires texte chiffré et texte déchiffré (exemple : attaque à texte choisi)
  - Rendre plus difficile l'analyse fréquentielle (on verra plus tard)

# CRYPTOGRAPHIE I – CRYPTANALYSE DE BASE



POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE



# Méthodes de cryptanalyse de base

- Force brute
  - Essaie de toute les clés
    1. Déchiffrer le texte chiffré avec la clé à essayer
    2. Voir si le résultat est cohérent
    - Paramètre de difficulté
      - Taille de l'espace de clés $N$  bits de clés =  $2^N$  clés possibles
      - Génération de clés non-aléatoire/non-uniforme
        - Entropie de la source K générant les clés:  $H(K) \leq N$
    - Critère de reconnaissance ou de succès
      - Comment savoir si on a la bonne clé?
        - Patron ou format reconnaissable
        - Le texte « fait du sens »
        - Le texte « marche », e.g. mot de passe, etc.
      - Paramètre de difficulté
        - Entropie de la source du message
          - Entropie basse → moins de messages “valides” → plus facile
          - Entropie élevée → plusieurs messages “valides” → difficile



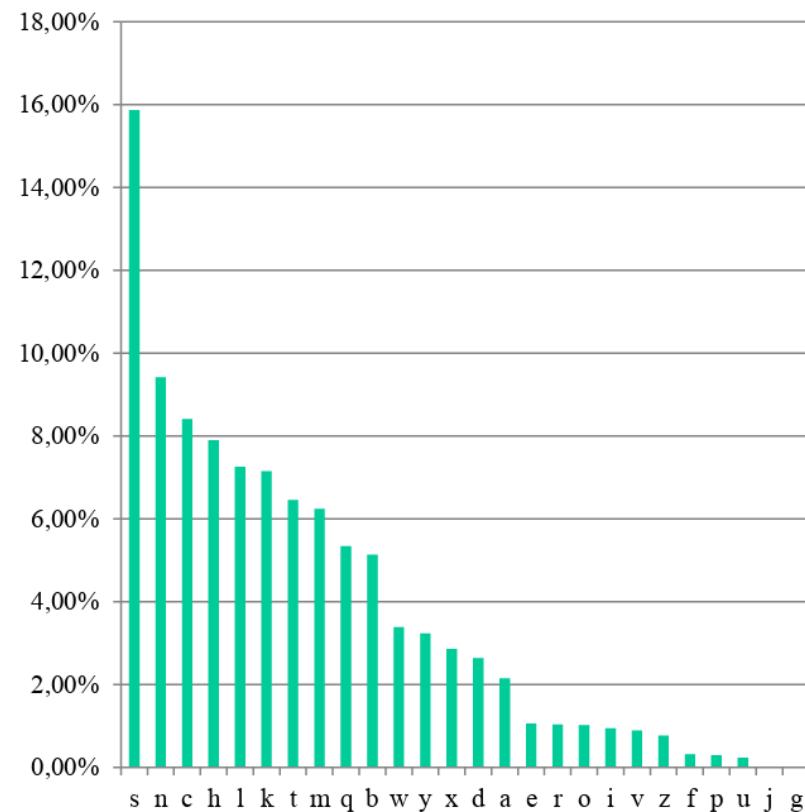
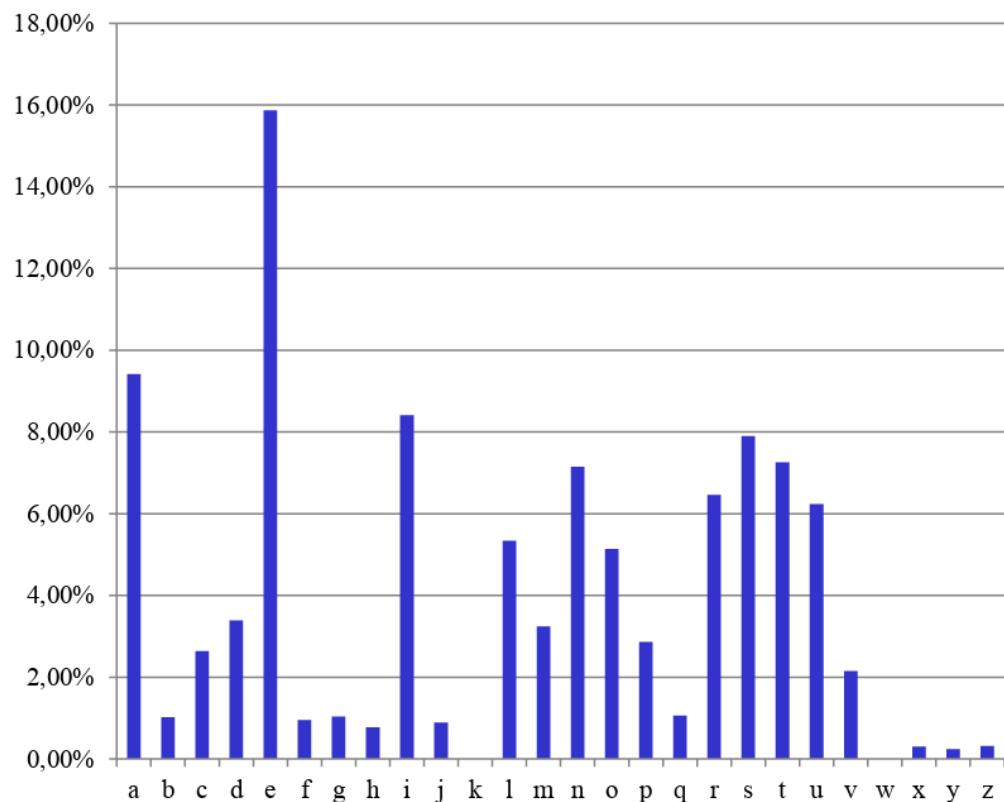
# Méthodes de cryptanalyse de base

- Analyse fréquentielle
  - Méthode
    1. Établir/retrouver fréquences des symboles de la source
    2. Calculer les fréquences des symboles chiffrés obtenus
    3. Comparer histogrammes de fréquences
    4. Établir relations entre symboles chiffrés et symboles de sources
    5. Essayer de déchiffrer le texte
  - Difficultés/précisions
    - Codage connu → possible d'inverser le codage
    - Paramètre de difficulté
      - Entropie de la source du message
        - » Entropie haute → histogramme « plat » → difficile
        - » Entropie basse → histogramme « escarpé » → plus facile
    - Variante - Analyse par bloc
      - Si entropie trop haute pour  $S$ , alors on essaie avec  $S^2$ ,  $S^3$ , ...
      - Compromis: taille de tableau de correspondance vs. entropie
      - Limite ultime = Entropie du langage



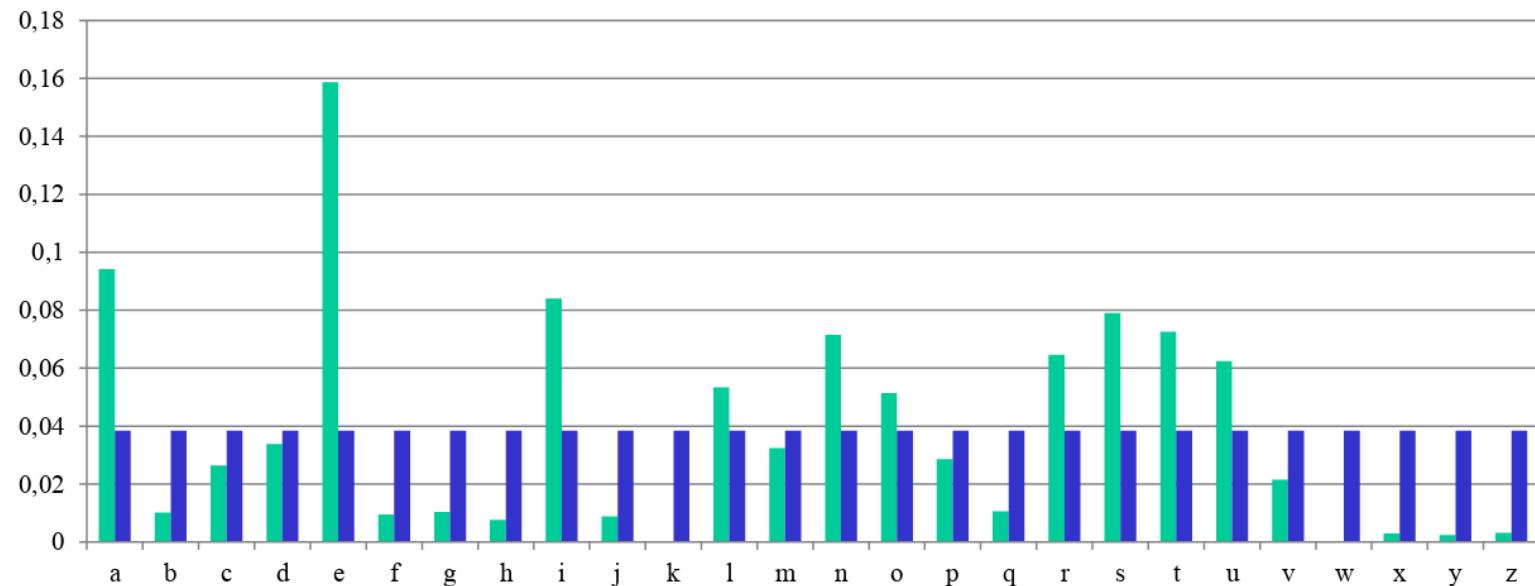
# Cryptanalyse fréquentielle

- Histogramme de fréquence par lettre en français
- Histogramme (ordonné) de d'un texte chiffré



# Cryptanalyse fréquentielle

- Si l'entropie était maximale (tous les caractères équiprobables), nous allons obtenir l'histogramme suivant



- Il est difficile de tirer des conclusions sur le texte original à partir du texte chiffré

# Cryptanalyse fréquentielle

- Une fois les caractères les plus probables démasqués, il devient difficile de déchiffrer les autres caractères
  - Il ne faut pas oublier que, pour un texte chiffré nous utilisons la PSEUDO-entropie, i.e. un estimateur statistique de l'entropie, on doit s'attendre à des déviations entre la valeur « observée » (proportion d'une lettre donnée) et la valeur « attendue » (fréquence d'usage dans le langage)
  - Les variations seront encore plus grande si l'échantillon est peu statistiquement représentatif (taille, type de langage, etc.)
- Rappel : l'entropie par bloc est donné par la formule suivante
$$\frac{H(S^b)/b}{\log_2 N}$$
  - En prenant des blocs de caractères, on peut obtenir plus d'information



# Cryptanalyse fréquentielle

- Tableau de la fréquence des digrammes en anglais

First Letter	Second Letter																										Space	Total	
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z			
A	2	144	308	382	1	67	138	9	322	7	146	664	177	1576	1	100	-	602	683	765	87	233	57	14	319	12	:50	7086	
B	136	14	-	-	415	-	-	-	78	18	-	98	1	-	240	-	-	88	15	7	256	1	1	-	13	-	:38	1417	
C	368	-	13	-	285	-	-	412	67	-	178	108	-	1	298	-	1	71	7	154	34	-	-	-	9	-	:47	2053	
D	106	1	-	37	375	3	19	-	148	1	-	22	1	2	137	-	-	83	95	3	52	5	2	-	51	-	:262	3770	
E	670	8	181	767	470	103	46	15	127	1	35	332	187	799	44	90	9	1314	630	316	8	172	106	87	189	2	:4904	11612	
F	145	-	-	-	154	86	-	-	205	-	-	69	3	-	429	-	-	188	4	102	62	-	-	-	4	-	:110	1561	
G	94	1	-	-	289	-	19	288	96	-	-	55	1	31	135	-	-	98	42	6	57	-	1	-	2	-	:86	1901	
H	1164	-	-	-	3155	-	-	1	824	-	-	5	1	-	487	2	-	91	8	165	75	-	8	-	32	-	:715	6733	
I	23	7	304	260	189	56	233	-	1	-	86	324	255	1110	88	42	2	272	464	558	5	165	-	15	-	18	-	:4	4501
J	2	-	-	-	31	-	-	-	9	-	-	-	-	-	41	-	-	-	-	-	56	-	-	-	-	-	:4	139	
K	2	-	-	-	337	-	-	-	127	-	-	10	1	82	3	1	-	-	50	-	3	-	-	-	-	8	-	:399	933
L	332	4	6	289	591	59	7	-	390	-	38	546	30	1	344	34	-	11	121	74	81	17	19	-	276	-	:630	3900	
M	394	50	-	-	530	6	-	-	165	-	-	4	28	4	289	77	-	-	53	2	85	-	-	-	19	-	:464	2160	
N	100	2	98	1213	512	5	771	5	135	8	63	80	-	54	349	-	3	2	148	378	49	3	2	2	115	-	:1182	5249	
O	65	67	61	119	34	80	9	1	88	3	123	218	417	598	336	138	-	812	195	415	1115	136	398	2	47	5	:294	5776	
P	142	-	1	-	280	1	-	24	97	-	-	169	-	-	149	64	-	110	48	40	68	-	3	-	14	-	:127	1337	
Q	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	66	-	-	-	-	-	:66	66		
R	289	10	22	133	1139	13	59	21	309	-	53	71	65	106	504	9	-	69	318	190	89	22	5	-	145	-	:1483	5124	
S	196	9	47	-	626	-	1	328	214	-	57	48	31	16	213	107	8	-	168	754	175	-	32	-	34	-	:2228	5292	
T	259	2	31	1	583	1	2	3774	252	-	-	75	1	2	331	-	-	187	209	154	132	-	84	-	121	1	:2343	8545	
U	45	53	114	48	71	10	148	-	65	-	-	247	87	278	3	49	1	402	299	492	-	-	-	1	7	3	:288	2678	
V	27	-	-	-	683	-	-	-	109	-	-	-	-	-	33	-	-	-	-	-	1	-	-	-	11	-	:864	664	
W	595	3	-	6	285	-	-	472	374	-	1	12	-	103	264	-	-	35	21	4	2	-	-	-	-	-	:326	2503	
X	17	-	9	-	9	-	-	-	10	-	-	-	-	-	1	22	-	-	-	23	8	-	-	-	-	-	:21	120	
Y	11	10	-	-	152	-	1	1	32	-	-	7	1	-	339	16	-	-	81	2	1	-	2	-	-	-	-	:1171	1827
Z	3	-	-	-	26	-	-	-	2	-	-	4	-	-	2	-	-	3	-	-	-	-	-	-	3	9	:54	54	
Space	1882	1033	864	515	423	1059	453	3388	237	93	352	717	876	478	721	588	42	494	1596	3912	334	116	1782	436	-	2	-	:19998	1E+05
Total	7069	1418	2059	3770	11645	1549	1906	6739	4483	31	932	3885	2163	5241	5781	1339	66	5129	5278	8536	2701	870	2507	121	1855	52	19974	1E+05	

- On remarque que certains digrammes sont plus fréquents que d'autres (ex. : HE avec 3155)
- On peut raisonner sur les digrammes de la même façon que sur les lettres

# Cryptanalyse fréquentielle

- Le fait que la source soit non-markovienne facilite le raisonnement sur les digrammes
  - Fréquence du digramme « Q U » en français : 1.6% (top 10 !)
  - Fréquence du digramme « Q U » source markovienne : 0.066%
  - L'entropie d'une source non-markovienne est moins élevée
  - (Les combinaisons « impossibles » sont plus instructives que les valeurs de fréquences des combinaisons possibles)
- On peut reprendre l'exercice avec des blocs de 3 lettres (trigrammes)
- Si on dispose des tables des fréquence, on peut considérer les  $n$ -grammes avec  $n$  qui tend vers l'infini
  - Encore plus de structures ! (mots, phrases, sens du texte, etc.)
  - Encore moins d'entropie

# cryptanalyse fréquentielle

- Il y a une limite théorique à la capacité de faire l'analyse fréquentielle en utilisant des blocs
  - Lorsque la taille de  $b$  (taille du bloc) tends vers l'infini, on obtient l'entropie du langage
  - Pour une source non-markovienne hautement structurée comme le langage humain, l'entropie par bloc diminue de façon asymptotique jusqu'à la limite
    - Plus de structure implique plus de différences entre les fréquences d'occurrence
    - Indique plus de compressibilité
    - Doit traiter une table de « symboles » de plus en plus grande !
  - Le tableau des digrammes est déjà difficile à interpréter (la fréquence maximale en anglais est de 3155/100 000)
    - Difficile de faire la distinction entre divers digrammes pratiquement équiprobales
    - Plus sensibles aux aléas de la statistique

# Cryptanalyse fréquentielle

- En pratique, pour compléter l'analyse, il est beaucoup plus ais  de vous fier   votre propre connaissance du langage, de la s mantique, du sujet du texte, etc.



# Contremesures et principes de base

- Maximisation de l'entropie sur symboles de codage
  - On ne peut pas
    - Changer la source (p.ex. pour réduire son entropie)
    - Choisir l'alphabet de codage (p.ex. déterminer par le chiffrement)
  - On peut
    - Ajuster le codage pour maximiser l'entropie
      - Faire du codage par bloc
      - Faire du bourrage (« padding ») avec des caractères aléatoires
      - Implémenter de la compression
    - Utiliser des algorithmes de chiffrement probabilistes
- Maximisation de l'entropie des clés choisies
  - Génération aléatoire
  - Contrôle sur la génération des clés (“souveraineté de clé”)

# Questions ?



# **INF 4420: Sécurité Informatique**

## **Cryptographie II**

Cuppens



# Aperçu – Crypto II

- Types de chiffrement
  - Par bloc vs. par flux
  - Symétrique vs. asymétrique
- Algorithmes symétriques modernes
  - DES
  - AES
- Fonctions de hachage
  - Propriétés
  - Obsolètes
  - Présentes et futures:
- Algorithmes à clé publique
  - Arithmétique modulaire
  - Notion de groupe



# Type de chiffrement – Bloc vs. Flux

- Chiffrement par bloc
  - Algorithme où chaque mot de code (un « bloc ») est codée avec la même clé  $k$
  - Pour la plupart des sources,  $|\Sigma| = M$  est petit, en conséquent  $|T| = N \gg M$ , de façon à éviter force brute
  - Codage
    - Doit « regrouper » symboles de  $\Sigma$  en blocs dans  $T$
    - Problème de « latency »
- Chiffrement par flux
  - Chaque mot de code est chiffré avec une clé différente
  - Les clés sont générés au « fur et à mesure »



# Type de chiffrement – Symétrique vs. Asymétrique

- Symétrique
  - Clé de déchiffrement = clé de chiffrement
  - La clé doit toujours être gardé secrète !!
- Asymétrique
  - Deux clés différentes
  - En général, il n'est pas possible de déduire clé de déchiffrement en connaissant clé de chiffrement, donc
    - Clé de chiffrement = Clé « publique »
    - Clé de déchiffrement = Clé « privée »
  - Facilite la gestion de clés
  - Permet plusieurs autres applications au-delà du chiffrement



# DES (Data Encryption Standard) – Historique

- Développée par le gouvernement américain
  - pour usage général par le public (intérêts privés commerciaux américains)
- Devis et spécifications en 1970
  - établie par le National Institute of Standards and Technology (NIST)
  - complètement spécifié et facile à comprendre
  - sécurité indépendante de l'algorithme lui-même (Principe de Kerkchoff)
  - disponible à tous et adaptable à diverses applications (usage « commercial »)
  - possibilité d'implantation économique en matériel et en logiciel
  - efficace d'utilisation, validable, et exportable
- Deuxième appel de propositions en 1974
  - Choix de l'algorithme "Lucifer" développé par IBM
  - Adopté comme « DES » le 23 novembre 1976

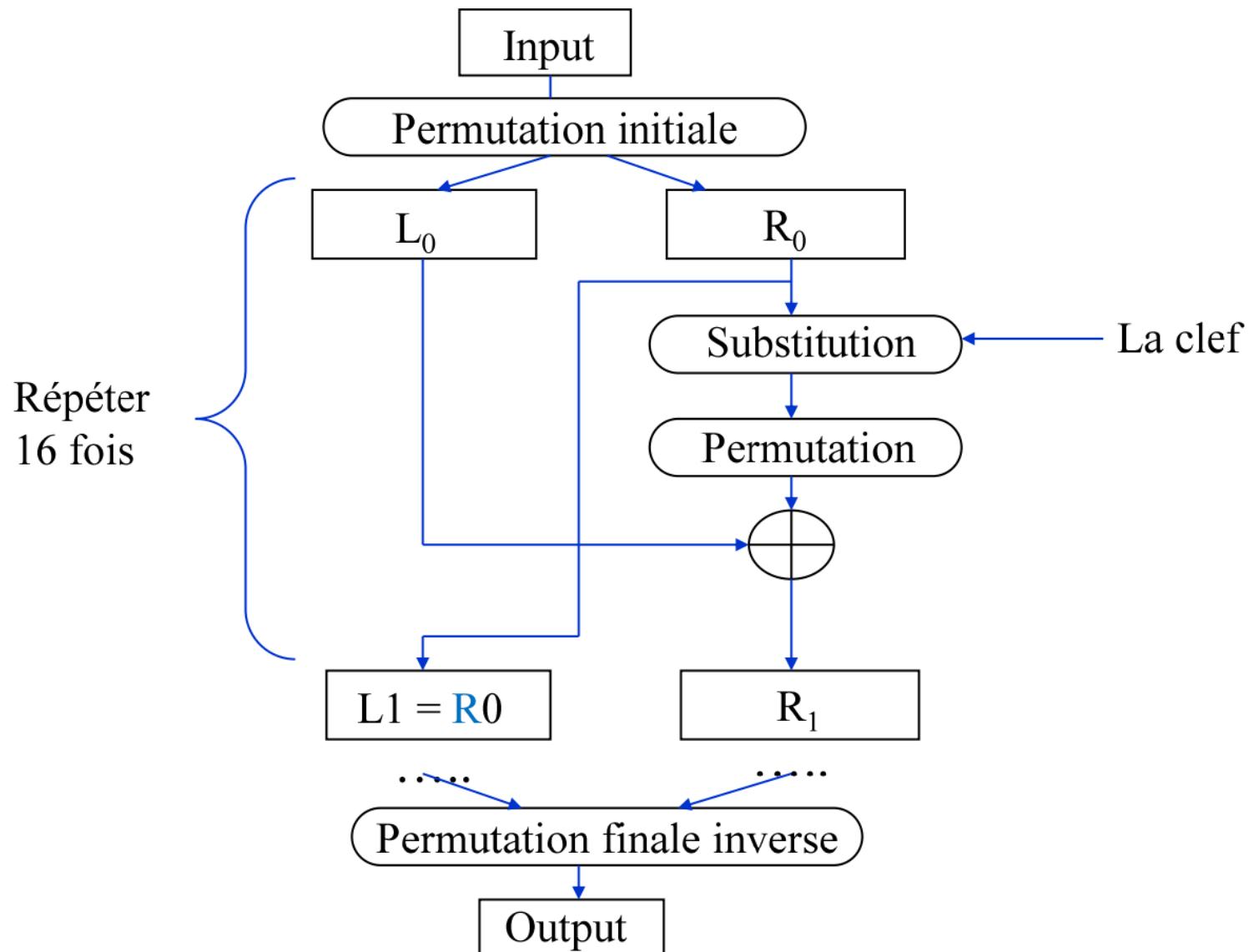


# DES Survol

- Application répétée (16 cycles) de
  - Substitution
    - changer systématiquement certains patrons de bits pour d'autres
  - Permutation
    - réarranger l'ordre des bits
- Arithmétique à 64 bits seulement; clef de 64 bits
- Chiffrement par blocs de 64 bits
- Objectifs de sécurité
  - Confusion
    - les bits d'output n'ont aucune relation évidente avec l'input
  - Diffusion
    - répartir les changements sur l'ensemble des bits du message
    - changement au bit  $i$  du message implique changement dans plusieurs bits du cryptogramme.

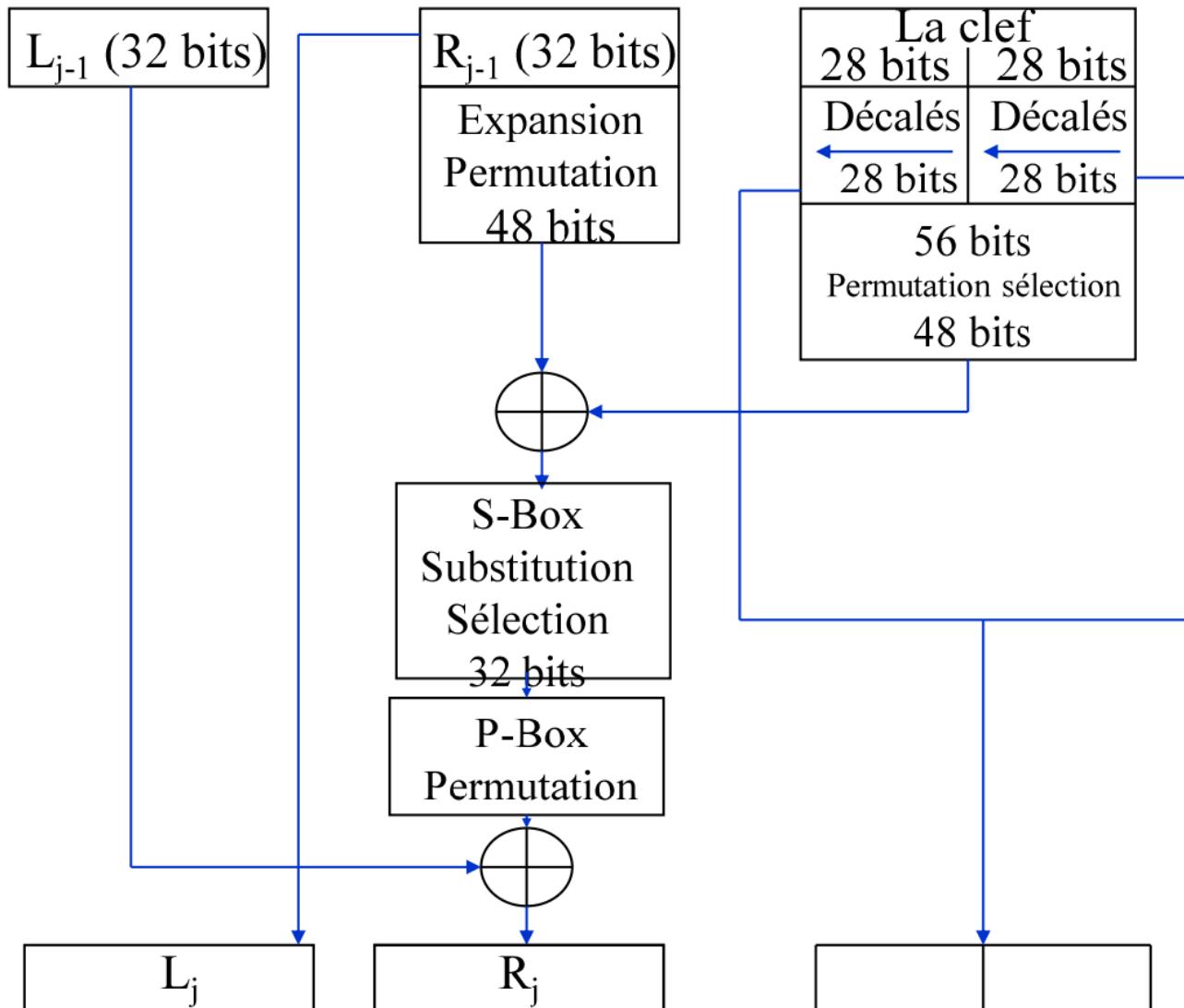


# Norme DES – Détails





# Norme DES – Détails d'un cycle





# Norme DES – Déchiffrement

- Chaque cycle de déchiffrement dérive du cycle précédent
  - $L_j = R_{j-1}; \quad R_j = L_{j-1} \oplus f(R_{j-1}, k_j)$
- Dans l'autre direction
  - $R_{j-1} = L_j; \quad L_{j-1} = R_j \oplus f(R_{j-1}, k_j)$
  - si on substitue:  $L_{j-1} = R_j \oplus f(L_j, k_j)$
- La procédure est donc réversible
  - la même fonction  $f$  est utilisée pour le déchiffrement
  - il suffit de prendre les 16 sous clefs dans l'ordre inverse



# S-box

- Quelle S-box est utilisée dépend de la clé de chiffrement
- Voici un exemple d'une S-box de DES (S5)

S5	Middle 4 bits of input							
	.0000.	.0001.	...	.1100.	.1101.	.1110.	.1111.	
0.....0	0010	1100	...	1101	0000	1110	1001	
<b>0.....1</b>	<b>1110</b>	<b>1011</b>	...	0011	<b>1001</b>	1000	0110	
1.....0	0100	0010	...	0110	0011	0000	1110	
1.....1	1011	1000	...	1010	0100	0101	0011	

- Exemple 6-bit string 0 1101 1 est converti en 4-bit string 1001



# Sécurité de la norme DES

- Questions relatives à la conception de l'algorithme
  - caractère confidentiel de la conception
  - présence de « trappes » (choix des s-box) ?
  - possibilité d'une faiblesse fondamentale ?
- Le nombre d'itérations (16) est-il suffisant
- La taille de la clef (56 bits) est-elle suffisante ?
  - originellement, Lucifer prévoyait 128 bits
  - possibilité d'une attaque force brute réussie
  - possibilité d'une attaque de type parallèle
  - possibilité de réussite d'une attaque de texte clair choisi
- Toutes ces questions avaient des réponses satisfaisantes



# Variantes de DES

- Double DES
  - Choisir deux clefs  $k_1$  et  $k_2$
  - Chiffrer deux fois:  $E(k_2, E(k_1, m))$
  - Est équivalent à un DES avec clé de 57 bits
    - 1 bit de clé supplémentaire...
    - ... seulement deux fois plus de travail pour briser
- Triple DES (ou 3DES)
  - Deux clefs
  - Trois opérations:  $E(k_1, D(k_2, E(k_1, m)))$
  - Équivalent à doubler la taille effective de la clé – 112 bits
  - Très robuste et effectif contre toutes les attaques faisables connues



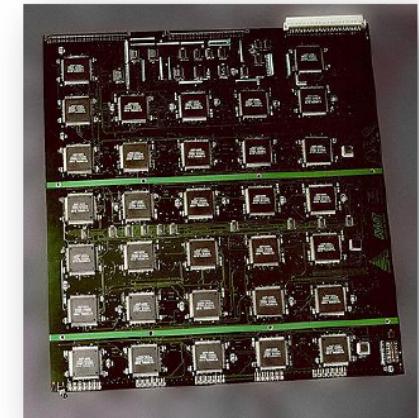
# Attaque par force brute – Limites ultimes

- Entropie
  - de la source de clé → nombres d'essais de clé
  - du texte clair (« plaintext ») → facilité de reconnaissance de la bonne clé
- Puissance de l'attaquant
  - Vitesse d'essai du matériel disponible
  - Nombre de machines disponibles (= budget total / cout par machine)
- Patience de l'attaquant
  - Combien de temps a-t-il ?  
(après combien de temps l'information n'est plus utile ?)
- Durée de vie de l'information
  - Après combien de temps l'information n'est plus confidentielle ?
- Loi de Moore
  - « À chaque 18 mois la puissance de calcul disponible pour un même budget double »  
→ Pour une sécurité équivalente on doit ajouter un bit de clé à chaque 18 mois...



# « DES is dead... »

- Attaque par force brute
  - 56 bits de clé ne sont pas suffisants aujourd'hui
  - COPACABANA
    - Cost-Optimized Parallel COde Breaker
    - Matériel spécialisé
    - Peut retrouver la clé en moins d'une journée !
    - Puissance
      - Nombre de secondes dans une journée =  $60*60*24 = 86\ 400 \sim 2^{18}$  s
      - Nombre d'essais =  $2^{56}$  clés
      - Vitesse d'essai =  $2^{56} / 2^{18} = 2^{56-18} = \underline{2^{38} \text{ clé/s}}$  (~ 256 Giga-clé/s)  
(gardez ce chiffre en tête!!)
    - COPACABANA est facilement reconfigurable pour d'autres algorithmes...
  - DES est obsolète, mais 3DES survi
    - demeure une norme acceptée quand même (pour le moment...)





# « Long live AES! »

- AES (Advanced Encryption Standard)
  - Nécessité de remplacer DES
  - Concours organisé en 1996 par le National Institute of Standards and Technology (NIST)
  - Cinq algorithmes finalistes internationaux
  - Choix final en 2001
    - Algorithme RIJNDAEL (pron. « *raïndol* »)
      - Proposé par cryptographes belges Joan DAEmen (Proton World Intl.) et Vincent RIJmen (Univ de Louvain)
  - Devient le Advanced Encryption Standard selon les normes :
    - USA - FIPS 197
    - Internationale – ISO/IEC 18033-3

# Rijndael (quelques détails)

- Algorithme itératif par bloc
- Plusieurs longueurs de clef et de bloc:
  - 128, (160,) 192, (224,) ou 256 bits (indépendantes l'une de l'autre)
- Une table d'état est utilisée
  - 4 rangées par  $N_b$  colonnes avec  $N_b = L_{\text{bloc}}/32$
- La clef est aussi représentée sous forme de tableau
  - 4 rangées par  $N_k$  colonnes avec  $N_k = L_{\text{clef}}/32$
- Le nombre de cycles (ou « rondes ») de transformation varie de 10 à 14 selon les valeurs de  $N_b$  et de  $N_k$
- On procède par une série de transformations/permutations/sélection
  - contrôlées par ces deux tableaux qui sont eux mêmes modifiés à mesure qu'on avance
  - Inspirées d'opérations sur  $\text{GF}(2^n)$
- Beaucoup plus performant que DES
  - utilisable pour une implantation en matériel.



# Rijndael/AES – Avantages et limites

- Principaux avantages
  - performance très élevée
  - possibilité de réalisation sur cartes à puces avec peu de code
  - possibilité de parallélisme
  - pas d'opérations arithmétiques: décalages et XOR seulement
  - n'est pas fondé sur d'obscures relations entre opérations
  - peu de possibilités d'insertion de trappes
  - possibilité de l'utiliser comme fonction de hachage
  - le nombre de rondes peut facilement être augmenté si requis
- Limites
  - le déchiffrement est plus difficile à implanter sur carte à puces
  - code et tables différents pour le chiffrement et le déchiffrement
  - dans une réalisation en matériel, il y a peu de réutilisation des circuits de chiffrement pour effectuer le déchiffrement



# Autres normes

- « International Data Encryption Algorithm » (IDEA)
  - proposé comme remplacement de DES
  - chiffre symétrique par bloc
  - blocs de 64 bits, clef de 128 bits
- Les autres finalistes AES
  - Tous: blocks de 128 bits, clés de 128, 192 ou 256 bits
  - Serpent:
    - Anderson, Biham, Knudsen,
  - Twofish/
    - Variante de Blowfish
    - Schneier et al.
  - RC6
  - Variante de RC5
  - Rivest, Robshaw (RSA)
  - Mars
  - Don Coppersmith (IBM)



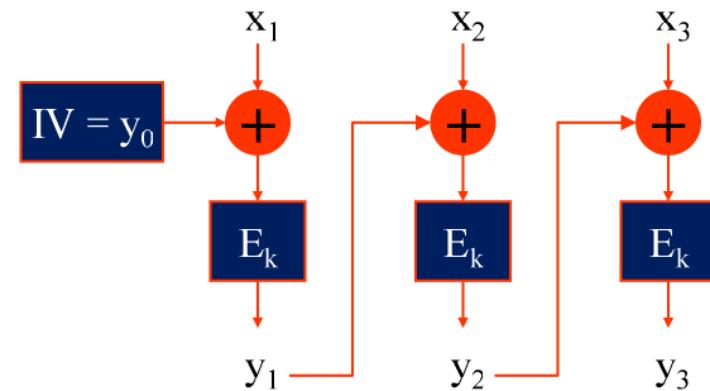
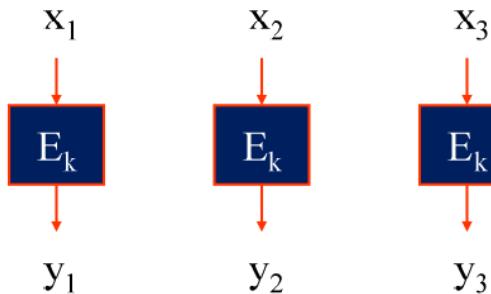
# Modes de chiffrement

- 4 modes de chiffrement
  - Historiquement introduit suite à la norme DES en 1981 (FIPS 81)
  - Définissent comment appliquer un algorithme de chiffrement par bloc pour la confidentialité de message de taille arbitraire
  - S'applique à n'importe quel algorithme de chiffrement symétrique
- À choisir selon critères d'application
  - Synchrone vs. asynchrone
  - Possibilité d'attaque par texte clair choisi, etc.



# Modes de chiffrement

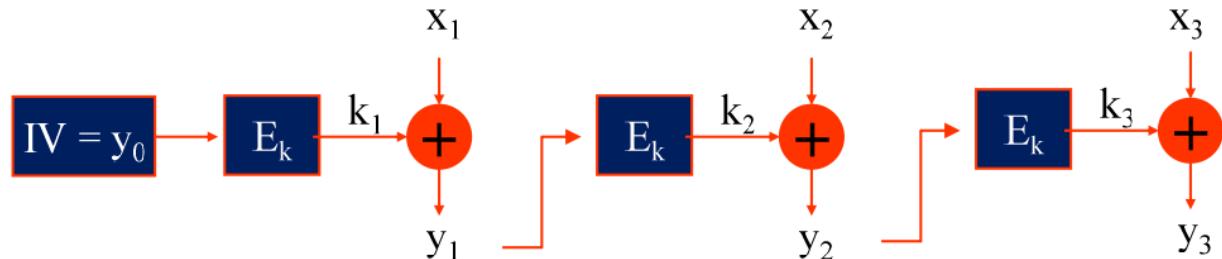
- Electronic Code Book (ECB)
  - Mode traditionnel
  - Chaque bloc chiffré indépendamment
  - La même clé est réutilisée
  - Permet transmission asynchrone
- Cipher Block Chaining Mode (CBC)
  - Chaque bloc XOR-é avec cryptogramme précédent
  - Utilise un vecteur d'initialisation (IV) comme paramètre cryptographique (pas nécessairement secret)



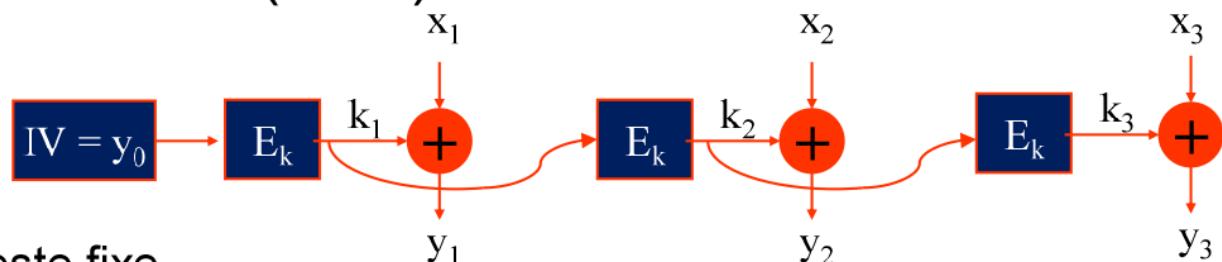


# Modes de chiffrement (suite)

- Cipher Feedback Mode (CFB)



- Clé pour DES reste fixe
- Clé pour XOR
  - cryptogramme antérieur chiffré par DES
- Output Feedback Mode (OFB)



- Clé pour DES reste fixe
- Clé pour XOR
  - Obtenu par application itérative de DES sur IV
  - Peuvent être générées d'avance (indépendante du message)

# L'algorithme RC4

- Caractéristiques
  - algorithme cryptographique par flux ("stream cipher")
  - inventé par Ron Rivest
  - secret commercial de RSA Data Security Inc.
  - dévoilé illégalement dans les mi-90
  - utilisé entre autres dans SSL (commerce électronique) et WEP
  - taille de clef variant de 8 à 2048 bits par intervalle de 8
  - génère une séquence pseudo aléatoire de bits
    - Utilisée comme « clé » pour le masque jetable (XORés avec le texte en clair)
  - le déchiffrement consiste à régénérer la séquence de bits et à inverser l'opération XOR
- Fiabilité
  - plusieurs faiblesses ont été identifiées
  - susceptible à une attaque par force brute si
    - la clef est choisie trop courte
    - le clés ou le IV sont mal choisi (basse entropie ou entropie nulle...)

# ALGORITHMES À CLÉ PUBLIQUE



POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE



# Notion de groupe

- Notion de groupe  $(G, \otimes)$ 
  - Un ensemble abstrait  $G$  sur lequel on a défini une opération abstraite «  $\otimes$  » avec certaines propriétés
    - élément identité :  $\exists 1 \in G$ , t.q.  $\forall a \in G$ ,  $a \otimes 1 = a$
    - Associativité :  $\forall a, b, c \in G$ ,  $a \otimes (b \otimes c) = (a \otimes b) \otimes c$ ,
    - Tout éléments à un inverse :  $\forall a \in G$ ,  $\exists a^{-1}$  t.q.  $a \otimes a^{-1} = 1$
    - (Commutativité):  $\forall a, b \in G$ ,  $a \otimes b = b \otimes a$   
on dit alors que le groupe est "abélien" ou "commutatif"
  - Exponentiation
    - $a^n = a \otimes a \otimes \dots \otimes a$ ,  $n$  fois  
où  $n$  est un entier et  $(G, \otimes)$  est un groupe abélien
  - Sous-groupe
    - Un sous-ensemble  $H$  de  $G$  est un sous-groupe de  $G$  si  $\forall a$ ,  $a^{-1} \in H$
    - Exemple : Sous-groupe cyclique
      - $\langle a \rangle := \{a^0, a^1, a^2, \dots\}$  est le sous-groupe cyclique de  $(G, \otimes)$   
« généré » par  $a$



# Calcul en arithmétique modulaire

- S'applique aux nombres entiers non-négatifs seulement
- $a$  modulo  $b$  est le reste entier de la division de  $a$  par  $b$
- Deux entiers sont équivalents modulo  $n$  si leurs modules sont égaux,

$x \equiv_n y$  si et seulement si  $(x \bmod n) = (y \bmod n)$

- Propriétés
  - Associativité:  $(a + (b + c)) \bmod n = ((a + b) + c) \bmod n$   
 $(a * (b * c)) \bmod n = ((a * b) * c) \bmod n$
  - Commutativité:  $(a + b) \bmod n = (b + a) \bmod n$   
 $(a * b) \bmod n = (b * a) \bmod n$
  - Distributivité:  $(a * (b + c)) \bmod n = ((a * b) + (a * c)) \bmod n$
  - Existence d'identités:  $(a + 0) \bmod n = (0 + a) \bmod n = a$   
 $(a * 1) \bmod n = (1 * a) \bmod n = a$
  - Existence d'inverses:  $(a + -a) \bmod n = 0$   
 $(a * a^{-1}) \bmod n = 1$  si  $\text{pgcd}(a, n) = 1$



# Arithmétique modulaire

## Inverses multiplicatifs

- L'inverse multiplicatif de  $a$  est  $b$  tel que  $a * b = 1$ 
  - exemples:  $(2 * 3) \text{ mod } 5 \Rightarrow 1$ ;  $(4 * 4) \text{ mod } 5 \Rightarrow 1$
- Pour un nombre premier  $p$ 
  - (Petit) Théorème de Fermat :
    - $a^p \text{ mod } p = a$ , donc  $a^{p-1} \text{ mod } p = 1$
    - $2^3 \text{ mod } 3 = 8 \text{ mod } 3 = 2$ ;
    - $4^5 \text{ mod } 5 = 1024 \text{ mod } 5 = 4$ ;  $4^4 \text{ mod } 5 = 256 \text{ mod } 5 = 1$ ;
  - si  $x$  est l'inverse de  $a$ 
    - $(a * x) \text{ mod } p = 1 = a^{p-1} \text{ mod } p$ , donc  $x = a^{p-2} \text{ mod } p$  ( $p$  premier)
- En général, pour un entier  $N$ 
  - Théorème d'Euler :
    - $a^{\varphi(N)} = 1 \text{ mod } N$
  - MAIS, l'inverse de  $a$  existe seulement si  $\text{pgcd}(a, N) = 1$



# Groupes pertinents

- Ensembles en arithmétique modulaire
  - $\mathbb{Z}_N = \{a \in \mathbb{Z} : 0 < a < N\}$
  - $\mathbb{Z}_p^* = \mathbb{Z}_p - \{0\} = \{a \in \mathbb{Z} : 0 < a < p\}$ , où  $p$  est premier
  - $\mathbb{Z}_N^* = \{a \in \mathbb{Z} : 0 < a < N, \text{pgcd}(a, N) = 1\}$
- Groupes pertinents
  - $(\mathbb{Z}_N, +)$  est un groupe commutatif
    - Tout  $a \in \mathbb{Z}_N$  a un inverse additif  $-a$
  - $(\mathbb{Z}_N^*, *)$  n'est pas un groupe
    - Si  $\text{pgcd}(a, N) = d > 1$ , alors  $a^*d = 0$ 
      - $a, d$  sont des diviseurs de 0
      - $a$  n'a pas d'inverse multiplicatif  $a^{-1}$  t.q.  $a^{-1}a = 1$
  - $(\mathbb{Z}_N^*, *)$  et  $(\mathbb{Z}_p^*, *)$  sont tous les deux des groupes commutatifs



# Groupe multiplicatif $Z_N^*$

- Combien d'éléments dans  $Z_N^*$ ?  
La fonction d'Euler  $\varphi(n)$  donne la réponse
  - si  $p$  est premier:
    - $\varphi(p) = p - 1$
    - $\varphi(p^k) = p^{k-1} (p - 1)$
  - si  $p, q$  sont relativement premiers
    - $\varphi(p * q) = \varphi(p) \varphi(q)$
  - en particulier si  $N = p * q$ , avec  $p$  et  $q$  premiers
    - $\varphi(N) = \varphi(p) * \varphi(q) = (p - 1) * (q - 1)$
- Quelle est la structure de  $Z_N^*$ ?
  - Tous les sous-groupes sont isomorphes à  $\langle a \rangle$  pour un  $a$  dans  $Z_N^*$
  - Les tailles (ou ordre) de ces sous-groupes sont les facteurs de  $\varphi(N)$



# Exemples de $\mathbb{Z}_N^*$

- $N = 12 = 2^2 * 3$ 
  - $\mathbb{Z}_{12}^* = \{1, 5, 7, 11\}$
  - $\varphi(12) = \varphi(2^2) \varphi(3) = 2 * 2 = 4$ 
    - Le seul diviseur de  $\varphi(12)$  est 2
    - ➔ Tous les éléments ont un ordre 2 ou 4
  - En effet
    - $5^2 = 25 = 1 \rightarrow \langle 5 \rangle = \{1, 5\}$
    - $7^2 = 49 = 1 \rightarrow \langle 7 \rangle = \{1, 7\}$
    - $11^2 = 121 = 1 \rightarrow \langle 11 \rangle = \{1, 11\}$
  - Noter qu'aucun élément à ordre 4
- $N = 15 = 3 * 5$ 
  - $\mathbb{Z}_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$
  - $\varphi(15) = \varphi(3) \varphi(5) = 2 * 4 = 8$ 
    - Les seuls diviseurs de  $\varphi(15)$  sont 2 et 4
    - ➔ Tous les éléments ont un ordre 2, 4 ou 8
  - En effet
    - $2^1 = \underline{2}, 2^2 = \underline{4}, 2^3 = \underline{8}, 2^4 = \underline{1}$ 
      - ➔  $\langle 2 \rangle = \{1, 2, 4, 8\}$
      - ➔  $\langle 4 \rangle = \{1, 4\}$
      - ➔  $\langle 8 \rangle = \{1, 8, 4, 2\}$
    - $7^2 = 4, 7^3 = 28 = 13, 7^4 = 91 = 1$ 
      - ➔  $\langle 7 \rangle = \{1, 7, 13, 1\}$
    - $11^2 = 121 = 1$ 
      - ➔  $\langle 11 \rangle = \{1, 11\}$
    - $13^2 = 4, 13^3 = 52 = 7, 13^4 = 91 = 1$ 
      - ➔  $\langle 13 \rangle = \{1, 4, 7, 1\}$
    - $14^2 = (-1)^2 = 1$ 
      - ➔  $\langle 14 \rangle = \{1, 14\}$
  - Noter qu'aucun élément à ordre 8



# **INF 4420: Sécurité Informatique**

## **Cryptographie II**



# **INF4420: Éléments de Sécurité Informatique**

## **Identification, Authentification**

Nora Cuppens



# Contenu du cours

- Authentification et gestion des identités
- Authentification par mot de passe
- Gestion des mots de passe
- Authentification deux facteurs
- Authentification mutuelle



# Gestion des identités et des accès

- Authentification
  - « Est-ce que c'est la bonne personne/système ? »
  - Identification usager → système
  - Identification système → système
  - Identification système → usager
- Autorisation
  - « Est-ce que cette personne a le droit de faire ça ? »
  - Contrôle d'accès
    - Physique vs. logique
    - Objet protégés et modes d'accès
    - Modèles de contrôle d'accès

# Authentification des usagers

- Les 4 modes d'authentification d'un usager
- Par quelque chose qu'il connaît
- Par quelque chose qu'il possède
- Par quelque chose qu'il est (biométrie statique)
- Par quelque chose qu'il fait (biométrie dynamique)



# Authentification des usagers

- Par quelque chose qu'il connaît
  - mot de passe
  - phrase de passe
  - information personnelle
    - nom
    - date de naissance
    - No. d'ass. sociale
    - ...
- Par quelque chose qu'il possède
  - carte magnétique
  - carte à puce
  - « dongle »
  - dispositif « Secure ID»
  - dispositif ou carte RFID
- Par quelque chose qu'il est (biométrie statique)
  - empreintes digitales
  - géométrie de la main
  - rétine de l'œil
  - iris de l'œil
  - caractéristiques du visage
- Par quelque chose qu'il fait (biométrie dynamique)
  - signature
  - voix
  - rythme au clavier
  - géolocalisation



# Authentification par possession d'un objet unique

- Doit être vraiment unique
- Doit être difficile/coûteux à reproduire
- Possiblement indépendant d'une base de données
- Problème de gestion de ces objets
  - émission
  - contrôle de possession
  - perte ou vol
  - récupération
- Faiblesses
  - coût
  - possibilité de falsification
  - perte ou vol
- Cas particulier
  - Ré-authentification
  - détection de la présence continue de l'usager

# Authentification biométrique statique

- Empreintes digitales
  - bonne précision: expérience policière antérieure
  - la contrefaçon est possible
- Géométrie de la main
  - assez précise
  - contrefaçon ?
- Rétine de l'œil
  - la plus précise des méthodes biométriques
  - utilisation d'un laser : réticence des usagers
  - peut être affectée en cas de maladie de l'usager
- Iris de l'œil
  - très précise :
    - 266 caractéristiques =>  $10^{78}$  combinaisons
  - lecture jusqu'à un mètre
  - n'est pas affecté par l'Age, la maladie incluant la cataracte
- Caractéristiques du visage
  - se rapproche le plus de la méthode humaine
  - le taux de précision reste à améliorer
  - utilisé pour identifier des individus dans des lieux public

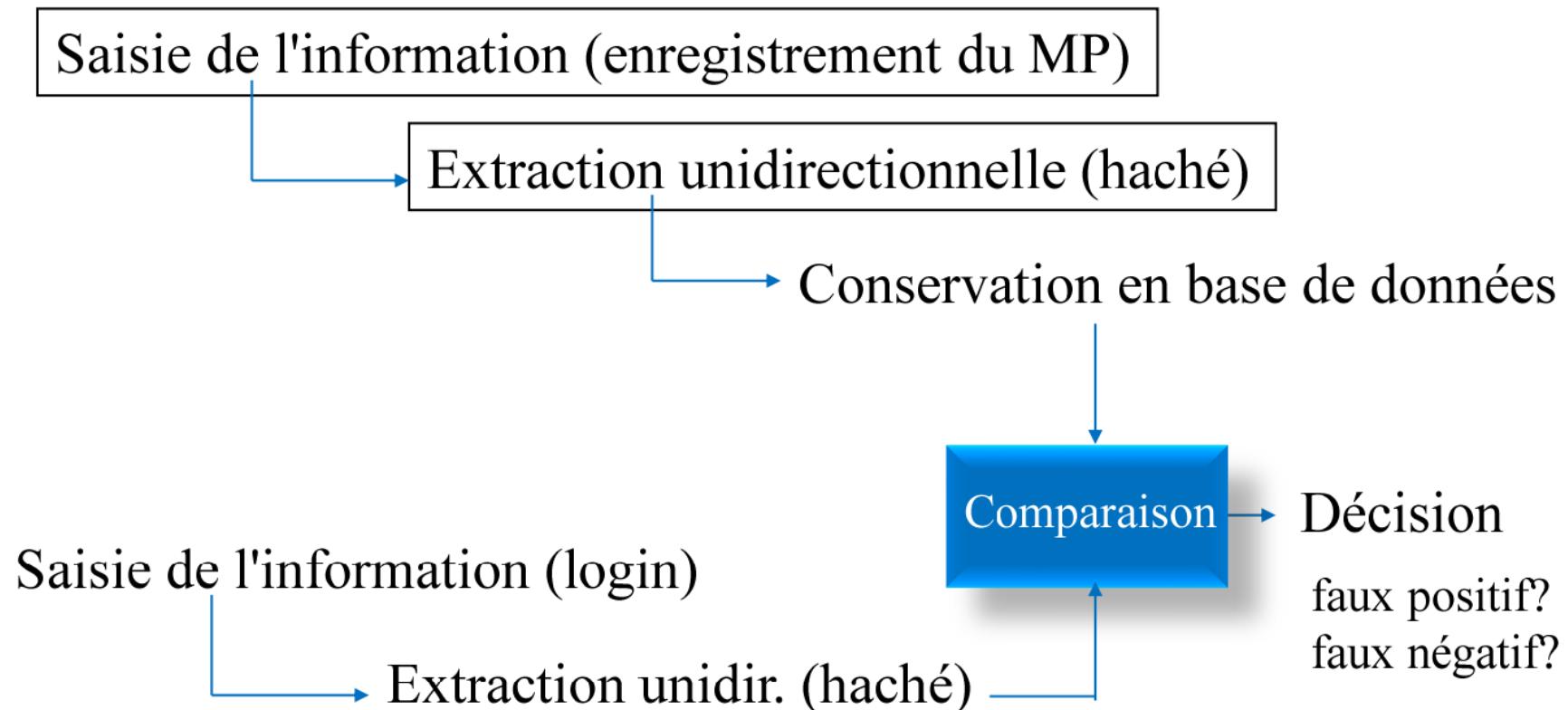


# Authentification biométrique dynamique

- Signature
  - tient compte de la dynamique du geste et non seulement de l'apparence de la signature
  - il faut entraîner le système: décision « floue »
  - la décision est sous forme d'une probabilité
- Voix
  - le moins précis de toutes ces méthodes biométriques
  - il faut entraîner le système
  - sensible à l'état de santé de la personne (laryngite)
  - contrefaçon facile
  - très accepté en général
- Rythme au clavier
  - pas encore très développé
  - décision « floue »
  - complètement transparent pour l'usager
  - surveillance continue tant que le clavier est utilisé
  - non-applicable si une interface graphique d'usager est utilisée



# Authentification par mot de passe - modèle général





# Attaque sur les mots de passe

- Techniques de base
  - Capturer et lire le fichier des mots de passe
    - surtout s'il n'est pas haché
  - Deviner le mot de passe
    - Online/Offline
  - Capturer le mot de passe
    - Enregistreurs de touches (« keyloggers »)
    - Post-it
    - Regard indiscret
      - (« Shoulder-surfing »)
  - Demander à l'usager
    - à son insu
      - Ingénierie sociale (« Social engineering »)
      - Supplantation
    - avec sa collaboration
      - \$\$\$
      - « Talents »

# Attaque sur les mots de passe

- Vulnérabilités de mots de passe
  - mots de passe probables
    - mots de passe courts
    - mots du dictionnaire
  - mots de passe avec lien à l'usager
    - information personnelle
    - information familiale

(Problème de base : faible entropie des choix de mots de passe !!! )

- mauvaise protection des fichiers de mots de passe
- pas d'authentification du système

# Attaques sur les mots de passe

Capturer le fichier des mots de passe hachés (attaque dictionnaire)

1. Construire une liste de mots de passe possible,  $w_1, \dots, w_t$
2. Pour chaque  $w_j$ , calculer  $h_j = H(w_j)$  et stocker dans une table T les paires  $(h_j, w_j)$ , trier par  $h_j$
3. Voler une base données de mots de passe hachés  $h_i = H(p_i)$
4. Chercher le mot de passe  $p_i$  correspondant à l'utilisateur  $u_i$  avec hash  $h_i$  en cherchant si  $h_i$  existe dans la table T. S'il existe, le mot de passe est  $w_j$

# Attaques sur les mots de passe

## Deviner le mot de passe

- Online: Pour un utilisateur connu  $U$ , essayer un ou plusieurs mots de passe séquentiellement ( $U, P_i$ ). Le serveur indique si le mot de passe est correct ou incorrect
  - Défenses: permettre un nombre limité d'essais, CAPTCHA, Ajouter un délai après chaque essai mauvais
- Offline: Comme online, mais plus difficile (impossible ?) de limiter le nombre d'essais
  - Défenses :
  - Fonctions hash spécialisée: Argon, bcrypt, scrypt
  - hash itératif (password stretching) –  $H^d(P_i)$ . Haché  $d$  fois le mot de passe avant de le stocker.  $d=1000$  limite la vitesse de l'attaque par une facteur de 1000.
  - Salt: ajouter une valeur de haute entropie  $s_i$  (p.ex. 128 bits), et stocker  $(u_i, s_i, H(p_i+s_i))$



# Mot de passe - Récupération

- MP temporaires et liens de récupération
  - Le serveur envoi par courriel un mot de passe temporaire (expiration après quelques heures ou après la première utilisation)
  - Communication par courriel n'est pas chiffrée !
- Question secrètes
  - Lors de l'enregistrement, l'utilisateur répond des questions prédéfinies.
  - Basse entropie pour les questions et souvent pour les réponses !



# Mot de passe - Récupération

Question 1 of 5

What is your favorite sport? ▾

Question 2 of 5

In what month is your best friend's bir... ▾

Question 3 of 5

What was the first major city that you ... ▾

Question 4 of 5

What is your favorite pizza topping? ▾

Question 5 of 5

What is your favorite warm-weather ac... ▾

Save my security questions

- Baseball
- Beach
- ✓ Biking
- Camping
- Canoeing
- Fishing
- Gardening
- Golfing
- Hiking
- Horseback riding
- Kayaking
- Mountain climbing
- Music festivals
- Paddleboarding
- Photography
- Picnicking
- Rafting
- Rock climbing
- Running
- Sailing
- Softball
- Sunbathing
- Surfing
- Swimming
- Tennis
- Wakeboarding
- Walking
- Waterskiing
- Windsurfing

# Gestionnaires de mots de passe

- Gestionnaires de mots de passe
  - Logiciel pour stocker des mots de passe (p.ex. 1password, LastPass, 1pass, Dashlane, KeePass)
  - Un mot de passe maître pour protéger plusieurs mots de passe
    - AKA « Single Sign-on » (SSO)
  - Avantages
    - Sécurité
      - mot de passe généré automatiquement avec plus d'entropie
    - Convivialité
      - auto-remplissage des champs de mot de passe (« auto-fill »)
      - génération automatique de mot de passe
  - Désavantages
    - Difficulté de synchronisation entre plusieurs PC,
    - Comment choisir un « bon » mot de passe maître?
    - Point de défaillance unique.
  - « Mettez tous vos œufs dans le même panier, et protégez bien ce panier ! » -- Andrew Carnegie, 1885

# Mots de passe – Contremesures

- Algorithme unidirectionnel
  - Introduire une variation aléatoire pour permettre plus d'une variation possible (« salt » en Unix)
  - Utilisation de fonction de hachage cryptographique sécuritaire
- Choix du mot de passe
  - Utiliser plus que 26 caractères: majuscule, minuscules, chiffres et symboles spéciaux
  - Utiliser un mot de passe suffisamment long (« phrase de passe »)
  - Éviter des mots de passe qui sont des mots du dictionnaire(s)
- Politique de gestion de mot de passe
  - Expiration des mots de passe
  - Mots de passe à usage unique : Un mot de passe = Un système
  - Contrôle des mécanismes de mise à zéro (« password reset »)
  - etc.

MOTS DE PASSE = TALON D'ACHILES !!!

# Authentification à deux facteurs (2FA)

- Idée
  - Combiner au moins deux facteurs d'authentification
  - En anglais : Two Factor Authentication (2FA) ou multi-factor
  - Chaque facteur a besoin d'une attaque différente
- Méthode « simple »
  - MP + {biométrie OU jeton} + ....
  - Tous les facteurs vérifiés localement par serveur d'authentification
- Méthode « threshold »
  - M de N facteurs doivent être corrects
- Méthode « OTP »
  - MP + Mot de passe à usage unique (One-Time Password)
  - Deux méthodes possibles
    - Méthode locale
    - Méthode à distance (remote)



# OTP - Méthode locale

- Dispositif
  - Porte clé avec écran (p.ex. Secure ID)
  - Calculatrice avec écran et clavier (pour entrer NIP)
  - Plateforme mobile avec application sécurisée
  - Chaque dispositif a un ID unique (un par compte)
- Méthode

## 1. Enregistrement du device

Secret  $S$  généré à partir du device ID et une clé maître,  
e.g.  $S = h(K, ID)$

## 2. OTP généré localement par le dispositif

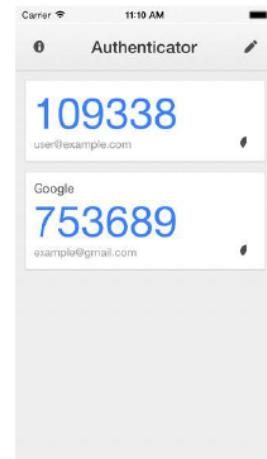
$$OTP = h(S, timestamp)$$

## 3. OTP envoyé par usager via Internet

## 4. Serveur vérifie

Identifie bon ID à partir de nom d'usager

Calcule  $S$  et  $h(S, timestamp)$  et vérifie égal à  $OTP$  envoyé





# OTP – méthode « remote »

- Dispositif
  - Téléphone cellulaire
- Méthode
  1. Usager se connecte en indiquant usager et MP
  2. Serveur calcule OTP aléatoire
  3. Serveur obtient no. de téléphone d'usage sur BD
  4. Serveur envoie OTP au téléphone par un autre canal (e.g. SMS), aussi appelé « side channel »
  5. Usager entre OTP et envoie via Internet
- Avantages
  - Force Ève à intercepter deux canaux indépendants
- Désavantage
  - Force l'usager à être sur réseau cellulaire ou avoir un accès à un deuxième canal



# Signaux vs. Facteurs d'authentification

- Des signaux d'authentification peuvent être envoyés sans participation de l'utilisateur p.ex:
  - Adresse IP
  - Cookies
  - Géolocalisation
  - Caractéristiques du matériel ou logiciel
- Les signaux peuvent augmenter l'assurance d'une authentification, mais ne peuvent pas être utilisés comme facteur indépendant



# Authentification dans les réseaux

- Problématiques additionnelles
  - Interception de la session d'authentification
  - Supplantation du système
  - Replay attacks
  - Session hi-jacking
  - Chess-master attack
    - L'attaquant est au milieu de la communication entre le client et le serveur
    - L'attaquant rejoue le trafic comme une partie d'échecs contre deux adversaires différents

# Système de « challenge-response »

- La possession d'une information  $I$  authentifie l'utilisateur au système
- Au lieu de dévoiler  $I$ ,
  1. Le système émet un « challenge »
  2. L'utilisateur répond au « challenge » avec une réponse, que seul quelqu'un connaissant l'information  $I$  peut calculer
  3. Le système vérifie que la réponse est bonne
- Avantages
  - Protège contre l'interception
- Désavantages
  - Le système doit connaître  $I$
  - Vulnérable à l'attaque de supplantation
  - Vulnérable à l'attaque de « replay »

# Preuves à connaissance nulle

- « Zero-Knowledge Proofs », en anglais
- Protocoles permettant à un démonstrateur  $P$  de prouver à un vérificateur  $V$  qu'il connaît quelque chose ou est capable de réaliser une tâche, sans que  $V$  n'apprenne rien d'autre que ce fait
- Sécurité basée sur des problèmes calculatoires difficiles :
  - Coloriage de graphe (NP-complet)
  - Isomorphisme de graphe (NP)
  - Calcul de résidu quadratique modulo  $N = p \cdot q$  (NP)



# Preuves à connaissance nulle

- Exemple simple de « Zero-Knowledge Proofs »
  - Une personne A non voyante possèdent deux billes de couleurs différentes
  - A rencontre une autre personne B
  - A utilise le protocole suivant pour savoir si B est un voyant ou un non voyant
- Étape 1 : A présente l'une des deux billes à B et lui demande de regarder la couleur de la bille
- Étape 2 : A présente ensuite l'une des deux billes à B et lui demande si c'est la même bille que la première fois
- A répète N fois l'étape 2
- Conclusion
  - B a une chance sur  $2^N$  de réussir le protocole
  - Si N est grand, la probabilité est faible que B soit non voyant
  - Le protocole est à connaissance nulle car A n'apprend rien d'autre que le fait « B est voyant »

# Preuves à connaissance nulle

- Applications en authentification
  - P détient une information I qui l'authentifie auprès du système
  - V émet un « challenge » aléatoire, que seul quelqu'un connaissant I peut résoudre
  - V ne connaît pas I
- Avantages
  - Résout le problème de supplantation
- Désavantages
  - Vulnérable au session hijacking et attaque « chessmaster »
  - Requiert une capacité de calcul chez l'utilisateur



# Authentification mutuelle

- Définition
  - L'authentification mutuelle ou bidirectionnelle désigne le fait que deux parties s'authentifient en même temps



# Authentification mutuelle

- Exemple simple d'authentification mutuelle
  - Double poignée de main « défi – réponse »
  - A et B possèdent un secret partagé S
  - Le client A envoie une valeur de défi unique  $sc$  au serveur B
  - B envoie une valeur de défi unique  $cc$  à A
  - A calcule  $cr = \text{hachage}(sc + \text{secrète})$  et envoie à B
  - B calcule  $sr = \text{hachage}(cc + \text{secrète})$  et envoie à A
  - B calcule la valeur attendue de  $cr$  et vérifie que A a répondu correctement
  - A calcule la valeur attendue de  $sr$  et vérifie que B a répondu correctement



# Authentification mutuelle

- KERBEROS

- Système de cryptographie à base de clés symétriques
- Kerberos repose sur un tiers de confiance, appelé Key Distribution Center (KDC)
- Un KDC se compose d'un serveur d'authentification (AS), qui authentifie un utilisateur, et d'un serveur d'octroi de tickets (TGS)



# Authentification mutuelle

- KERBEROS (suite)
  - Kerberos partage avec chaque client du réseau une clé secrète faisant office de preuve d'identité
  - Les clés secrètes partagées permettent l'authentification mutuelle des clients
  - L'authentification proposée par le serveur Kerberos a une durée limitée dans le temps,
    - Pour éviter les attaques par rejeu

# Authentification mutuelle

- L'authentification mutuelle est par défaut dans certains protocoles
  - IPSEC via IKE (Internet Key Exchange)
  - SSH
- Facultatif dans d'autres
  - Exemple : SSL-TLS



# **INF4420: Éléments de Sécurité Informatique**

## **Identification, Authentification**

Nora Cuppens



POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉnierie

# INF4420a: Sécurité Informatique

## Cryptographie III



# Aperçu – Crypto III

- Cryptographie à clé publique
  - Algorithme RSA
  - Algorithme de El-Gamal
    - Chiffrement à courbe elliptique (ECC)
  - Algorithmes post-quantiques
- Autres primitives cryptographiques
  - Hachage cryptographique
  - Signature numérique et infrastructures à clé publique (PKI)
  - Échange de clés
    - Diffie-Hellman
    - Cryptographie quantique
- Méthodes cryptanalytiques (suite)
  - Type d'attaques
  - Attaques quantiques
  - Complexité
- Principes d'utilisation de la cryptographie
  - Gestion de clés (privées et publiques)
  - Risques résiduels liés à l'utilisation de la crypto



# RSA : préliminaires

- Deux clefs:
  - clef publique : une paire  $(e, N)$
  - clef privée : une paire  $(d, N)$
- Première étape: choisir  $N$ 
  - Choisir  $n$ , la taille de  $N$  en fonction de sécurité requise
    - Selon le niveau de sécurité désiré
    - Tailles typiques  $n = (1024), 2048, 3072, 4096, 8192$
  - Trouver deux nombres premiers  $p$  et  $q$  de taille  $n/2$
  - Calculer  $N = p * q$
- Choisir un entier  $e$  de taille  $n$ 
  - relativement premier à  $\varphi(N) = (p-1)*(q-1)$  i.e.  $\text{pgcd}(e, \varphi(N)) = 1$
- Choisir  $d$  tel que
  - $e * d \equiv 1 \pmod{(p-1)*(q-1)}$
- À la fin, on n'a plus besoin de conserver  $p$  et  $q$

# RSA : Chiffrement et déchiffrement

- Alphabet de codage
  - $T = \mathbb{Z}_N^*$ 
    - généralement représentés par des chaînes de  $n$  bits
- Fonction de chiffrement
  - $x \in \mathbb{Z}_N^* \rightarrow y = E_e(x) = x^e \bmod N$
- Fonction de déchiffrement
  - $y \in \mathbb{Z}_N^* \rightarrow x' = D_d(y) = y^d \bmod N$
- Est-ce que  $x' = x$  ?
  - $D_d(E_e(x)) = (x^e)^d \bmod N = x^{ed} \bmod N = x \bmod N$   
(parce que  $e^*d = 1 \bmod \varphi(N)$  et théorème de Euler)



# L'algorithme RSA : exemple

- On utilise des valeurs petites pour illustrer
  - Soit  $p = 11$  et  $q = 13$ , alors
    - $N = p * q = 143$  et  $\phi(N) = (p-1) * (q-1) = 120$
  - $e$  doit être relativement premier à  $(p-1) * (q-1)$ :
    - Exemple 1 : soit  $e = 11$ 
      - l'inverse de 11 mod 120 est aussi 11:  $11 * 11 = 121 = 1 * 120 + 1$  (pas très astucieux ! Cas atypique ...)
      - Pour chiffrer un mot de code  $x = 7 \rightarrow y = 7^{11} \text{ mod } 143 = 106$
      - Pour déchiffrer  $y = 106 \rightarrow x = 106^{11} \text{ mod } 143 = 7$
      - La clef publique est  $N = 143$ ,  $e = 11$
    - Exemple 2 : soit  $e = 17$ 
      - Alors  $d = 113$ , parce que  $17 * 113 = 1921 = 16 * 120 + 1$
      - Pour chiffrer un mot de code  $x = 7 \rightarrow y = 7^{17} \text{ mod } 143 = 50$
      - Pour déchiffrer  $y = 50 \rightarrow x = 50^{113} \text{ mod } 143 = 7$
      - La clef publique est  $N = 143$ ,  $e = 17$



# L'algorithme RSA : exemple

- Avec des petites valeurs, on peut retrouver  $d$ 
  1. En factorisant  $N$ 
    - 143 ne peut pas être autre chose que  $11 * 13$
    - on peut le retrouver en 11 essais
  2. En calculant  $\phi(N)$
  3. En retrouvant la clé privé  $d$ 
    - en résolvant équation  $e * d = 1 \text{ mod } \phi(N)$ , avec l'algorithme d'Euclide



- Plus grand commun diviseur (pgcd)
  - soit deux nombres  $a$  et  $b$ ; le plus grand entier qui divise  $a$  et  $b$  sans reste est le pgcd de ces deux nombres
- Algorithme d'Euclides
  - Calcul du pgcd
    - Pour  $\text{pgcd}(3615807, 2763323)$   
 $3,615,807 = 1 * 2,763,323 + 852,484 \quad a = m * b + r$   
 $a \leq b; \quad b \leq r$   
 $2,763,323 = 3 * 852,484 + 205,871$   
 $852,484 = 4 * 205,871 + 29,000$   
 $205,871 = 7 * 29,000 + 2,871$   
 $29,000 = 10 * 2,871 + 290$   
 $2,871 = 9 * 290 + 261$   
 $290 = 1 * 261 + 29$   
 $261 = 9 * 29 + 0$     - $\text{pgcd}(3615807, 2763323) = 29$
  - Permet également de calculer les inverses multiplicatifs dans  $Z_N^*$
  - Complexité =  $O(n^3)$

## Exponentiation rapide

Permet de calculer  $x^e \bmod N$  en temps  $O(n^3)$



# Comment trouver un nombre premier très grand

- L'algorithme standard - Crible d'Ératosthène
  - On examine tous les nombres en éliminant ceux qui sont un produit de deux facteurs différents de 1
- Test de primalité probabiliste
  - Tout nombre premier doit satisfaire deux conditions:
    - $\text{pgcd}(p, r) = 1$  et  $J(r, p) \equiv r^{(p-1)/2} \pmod{p}$
    - $r$  est un nombre plus petit que  $p$
    - $J$  est la fonction de Jacobi
  - |  $1 \text{ si } r = 1$   
 $J(r,p)= J(r/2)*(-1)^{(p*p -1)/8} \text{ si } r \text{ est pair}$   
 $J(p \bmod r, r) (-1)^{(r-1)*(p-1)/4} \text{ si } r \text{ est impair et } \neq 1$
  - si c'est satisfait, la probabilité que  $p$  soit premier est 50%
  - si on répète le test  $k$  fois avec succès, la probabilité est  $1-2^{-k}$



# Problème du log discret

- Propriétés mathématiques de  $Z_p$ 
  - Tous les éléments de  $Z_p$  ont des inverses multiplicatifs, sauf 0
    - Donc,  $Z_p^* = Z_p - \{0\}$
  - Il existe des éléments  $g$  dit *générateur* ou *racine primitive* tel que :  
 $Z_p^* = \{g^0, g^1, \dots, g^{p-1}\}$ 
    - Notes :
      - Il est possible de vérifier en temps polynomial si un élément  $g$  est un générateur.
      - Il existe un très grand nombre de générateurs dans  $Z_p^*$
  - Définition : Le logarithme discret en base  $g$  de  $a \in Z_p$  est l'entier  $x$  tel que  $a = g^x \bmod p$
  - Hypothèse calculatoire : Il n'est pas possible de calculer le log discret en temps polynomial sans connaître la factorisation de  $p-1$ .



# Algorithme de El-Gamal

- Génération de clé
  1. Trouver un grand entier premier  $p$  tel que  $p-1$  a au moins un grand facteur premier (donc difficile à trouver)
  2. Choisir au hasard un générateur  $g$  de  $Z_p^*$  et un entier  $d$
  3. Calculer la valeur  $e = g^d \text{ mod } p$
  4. Clé publique =  $(p, g, e)$ , Clé privé =  $d$
- Chiffrement/Déchiffrement – pour un message  $x \in Z_p^*$ 
  - Choisir un entier  $k \in [0..p-1]$  au hasard
  - $E(k,x) = (y_1, y_2) = (g^k \text{ mod } p, xe^k \text{ mod } p)$
  - $D(y_1, y_2) = y_2 / y_1^d \text{ mod } p$



# Algorithme de El-Gamal

- Intuition
  - Le message  $x$  est "masqué" dans  $y_2$  en le multipliant par  $e^k$  par
  - La partie  $y_1$   
**fournit à qui connaît  $d$ , l'information nécessaire pour reconstruire  $x$  en "divisant" par  $y_1^d$  (en réalité, calculer son inverse et multiplier)**



# Algorithme de El-Gamal – Notes importantes

- Méthode de chiffrement dite "probabiliste"
  - il n'existe pas de chiffrement unique pour un même  $x$ .
- Choix de  $k$ 
  - Ce n'est pas une clé *strictu sensu*
    - Il n'est pas nécessaire que Bob connaisse  $k$  en avance pour déchiffrer  $x$
  - C'est plutôt un *masque*
    - Qui ne doit pas être dévoilée (!)
  - Très important de choisir un  $k$  différent à chaque fois
    - Un mauvais choix de  $k$  (toujours pareil, basse entropie) rend possible des attaques à texte clair choisi (dangereuses si entropie après codage basse)



# Notion de groupe - rappel

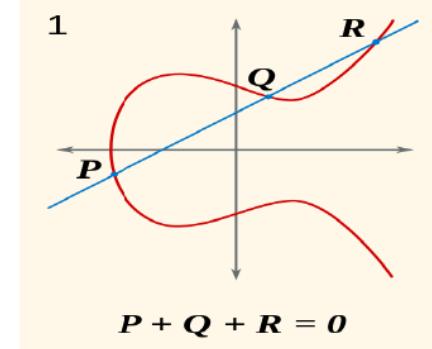
- Notion de groupe  $(G, *)$ 
  - Un ensemble abstrait  $G$  sur lequel on a défini une opération abstraite " $*$ " avec certaines propriétés :
    - Il existe un élément identité  $1$  :  $a * 1 = a$
    - Associativité :  $a * (b * c) = (a * b) * c$
    - Tout élément à un inverse :  $a * a^{-1} = 1$
    - (Commutatif) :  $a * b = b * a$ , on dit que le groupe est "abélien" ou "commutatif"
  - Définition (exponentiation) :
    - $a^n = a * a * \dots * a$ ,  $n$  fois où  $n$  est un entier et  $(G, *)$  est un groupe abélien
  - Note: On peut définir le problème de log discret sur n'importe quel groupe
- Exemples
  - $\mathbb{Z}_p^*$ , Corps fini (corps de Galois), Courbe elliptique

# El-Gamal sur Corps de Galois

- **GF( $2^n$ )**
  - Corps de Galois (corps fini) de taille  $2^n$
  - Se base sur l'arithmétique modulaire avec des polynômes de degré  $n$
  - Toutes les coefficients des polynômes sont binaires
    - toute l'arithmétique est binaire
- **El-Gamal sur GF( $2^n$ )**
  - Chiffrement et déchiffrement très efficaces
  - Surtout utilisée sur des plateformes matériel
  - Peu utilisé aujourd'hui



- Courbe elliptique
  - Se base sur l'opération de "somme" de points sur une courbe elliptique
  - Combiné avec des opérations sur des corps de Galois
- ECC
  - Permet un niveau équivalent de sécurité avec des tailles de clés entre 6-12 fois plus petites (p.ex. 256 bit EC  $\cong$  3072 bit RSA)
    - ➔ meilleure performance de chiffrement et déchiffrement
    - ➔ taille réduite des signatures numériques pour applications avec peu de bande passante





# ECC - suite

- Standard NIST
  - ECDH (échange de clés)
  - ECDSA (signature numérique)
  - Pas utilisé pour le chiffrement (clé symétrique)
- Paramètres recommandés
  - Taille de clé 256 (SECRET) ou 384 bit (TOP SECRET)
  - Courbes elliptiques
    - En principe, n'importe quelle courbe peut être utilisée
    - Certaines courbes seraient moins “robustes” que d'autres
    - Standard NIST
      - Le choix de courbes possible est limité
      - “Théorie de la conspiration”
        - » *Les courbes auraient été choisies par la NSA/NIST avec des “trappes” permettant de déchiffrer sans connaître la clé (à la S-box dans DES....)*

- Caractéristiques
  - Une généralisation de l'algorithme de Shor (Mosca, Ekert 1998) permet de retrouver un groupe “caché” sous-jacent (“hidden subgroup”) en temps polynomial quantique
    - ➔ Impose une restriction sur les algorithmes post-quantiques
    - ➔ Élimine d'emblée plusieurs algorithmes à clé publique
- Type (selon l'origine)
  - Algorithmes « pré-quantique »
    - Anciens algorithmes à clé publique (années 80)
    - Avaient été mis de côté car moins pratiques
    - Maintenant sorti de l'armoire et « recyclés » (p.ex. McEliece)
  - Nouveaux algorithmes
    - Invention plus récente (années 90 et 2000)
    - Souvent (mais pas tout le temps) motivée par la menace quantique (p.ex. basés sur les réseaux)

# Cryptographie post-quantique (PQC)

- Types (selon le problème calculatoire sous-jacent)
  - Basé sur les réseaux euclidiens (« lattice-based »)
  - Basé sur les polynômes multivariés
  - Basé sur codes correcteur d'erreurs
  - Basé sur les fonctions de hachage....
- NIST Post-Quantum Project
  - Concours lancé en 2016 pour trouver un standard de PQC
  - Round 1 – 60+ candidats proposés
  - Round 3 (juil 2020) –
    - Chiffrement : 4 finalistes choisis
      - McEliece, CRYSTALS-KYBER, NTRU, SABER
    - Signature numérique : 3 finalistes choisis
      - CRYSTALS-DILITHIUM, FALCON, Rainbow
- Tests en utilisation réelle
  - Test de performance (2016) réalisé par Google sur Google Chrome
  - Implémentait TLS avec la proposition de PQC CECPQ1, développé par Google
  - Résultats encourageants

# Cryptographie post-quantique (PQC)

- Problème principaux
  - Algorithmes récents ou peu utilisés
    - ➔ Peu de scrutin de la résistance aux attaques classiques ou quantiques
  - Taille de clés plus élevées
    - Certificats et signatures plus grands
    - Peu pratiques pour des applications avec bande passante limitée

Algorithme	Type	Clé publique
<i>256-bit Elliptic Curve</i>	Classique	32 B
<i>3072-bit Discrete Log</i>	Classique	384 B
<i>SIDH</i>	Isogénie	751 B
<i>Streamlined NTRU Prime</i>	Réseau	1232 B
<i>Quasi-cyclic MDPC-based McEliece</i>	Code correcteur	1232 B
<i>New Hope</i>	Réseau (RLWE)	2 KB
<i>NTRU Encrypt</i>	Réseau	6130 B
<i>Random Linear Code based encryption</i>	RLCE	115 KB
<i>Rainbow</i>	Polynôme multivar.	124 KB
<i>Goppa-based McEliece</i>	Code correcteur	1 MB



# Hachage cryptographique

- Objectif : Intégrité
  - S'assurer qu'un message n'a pas été modifié de façon non autorisée une fois qu'il a été terminé par son auteur légitime
- Fonctions de hachage cryptographique  $h$ 
  - Une fonction  $h( )$  est dite de hachage cryptographique si à partir d'un message  $x$  elle produit un *haché* (ou *hachage*)  $h(x)$ , avec ces propriétés
    1. **(à sens unique)** : il est très difficile de trouver un  $x$  à partir d'un  $h$  donné, tel que  $h = h(x)$
    2. **(absence de collision)** : il est très difficile de trouver deux messages  $x$  et  $y$ , tel que  $h(x) = h(y)$
    3. **(effet d'avalanche)** : il est très difficile de trouver à partir d'un  $x$  donné de trouver un autre  $x'$  « similaire » tel que  $h(x) = h(x')$
  - Notes
    - 3 implique 2 (trivial), 3 implique 1 (pas trivial)
    - En anglais,  $h(x)$  est appelé "hash", MAC (pour *Message Authentication Code*), "message digest" ou simplement "digest"
    - Ne pas confondre avec les fonctions de hachage universelles, utilisées par exemple dans la construction de compilateur, les structures de données et algorithmes aléatoires, etc.



- Fonctions obsolètes

- MD4 (128 bit)

- Conçu par Rivest (de RSA)
    - Ressemble un peu à DES
      - Plusieurs rondes de coupage, transposition, permutation, et autre opérations binaires.

- MD5 (128 bit)

- Version amélioré de MD4
    - Usage très répandu
    - Utilisé par le programme linux md5sum
    - **Collisions en quelques heures!**

- SHA-1 (160 bit)

- Conçue par la NSA
    - Collisions possibles
    - Remplacé officiellement depuis 2011

- Fonctions recommandées

- SHA-2

- Famille de 5 fonctions introduites en 2001
    - Conçue par la NSA
    - Taille de haché : 224, 256, 384, 512
    - Similaire en structure à MD5 et SHA1
    - Aucune vulnérabilité connue (2020)

- SHA-3

- Compétition du NIST en 2006
    - Besoin d'avoir une famille de fonction différente de MD5/SHA1/SHA2 (au cas où...)
    - Algorithme KECCAK choisi (Bertoni, Daemen, Peeters, van Aasche)
    - Taille de haché : 224, 256, 384, 512



# Intégrité de message avec haché (MAC)

- Modèle théorique
  - Modèle de Shannon révisé - version « intégrité »
    - Ève peut intercepter tout message et le lire
    - Ève peut modifier le message partiellement ou entièrement sans que Alice ou Bob s'en rende compte
      - Attaque *par personne interposée*
        - » En anglais « Man-in-the-middle » (MITM) ou « person-in-the-middle » (PITM)



- Objectif
  - Alice et Bob veulent s'assurer que toute modification d'un message original  $x$  soit détectable par Bob



# Intégrité de message par MAC

- Protocole « canonique » (de base)
  1. Alice calcule le haché  $h = h(x)$  de son message  $x$ 
    - $h$  est le « message authentication code » (ou MAC) pour  $x$
  2. Alice transmet le message  $x$  par le canal de communication
  3. Alice transmet le haché  $h(x)$  soit
    - a) En utilisant un canal de transmission alternatif qu'Ève ne peut pas modifier (« out-of-band » transmission)
    - b) Sur le canal principal, mais en utilisant une méthode d'authentification qu'Ève ne peut pas falsifier (p.ex. reconnaissance vocale, etc.)
  4. Bob reçoit un message  $x'$  (possiblement modifié)
  5. Bob calcule  $h(x')$  et compare avec  $h(x)$  reçu
    - ➔ Si  $h(x') = h(x)$  alors avec très haute probabilité  $x' = x$  et aucune manipulation n'a eu lieu

# Intégrité avec clé secrète partagée

- Si Alice et Bob
  - Ne dispose pas d'un moyen d'authentification,
  - Ne dispose pas d'un canal alternatif sécurisé (non accessible à Ève),
  - Mais ils peuvent partager une clé secrète au préalable, alors
- Protocole MAC à clé partagée
  - 0. Alice et Bob partage une clé secrète  $K$  connue d'eux seulement
  - 1. Lorsqu'Alice veut transmettre  $x$ , elle calcule  $h = h(K \parallel x)$
  - 2. Alice transmet  $x$  par le canal principal
  - 3. Alice transmet  $h$  par le même canal que  $x$
  - 4. Bob calcule  $h' = h(K \parallel x')$  à partir du message reçu  $x'$
  - 5. Si  $h = h'$  alors très probablement  $x = x'$  et le message n'a pas été modifié
- Avantages
  - Ève ne peut pas « tromper » Alice et Bob sans connaître la clé  $K$
  - Pas besoin d'un 2e canal sécurisé



# Protocole standardisé (HMAC)

- Protocole standardisé (standard HMAC)
  1. Alice calcule  $\text{HMAC}(K, x)$  où
    - $\text{HMAC}(K, x) = h(K' \oplus opad) \parallel h(K' \oplus ipad) \parallel x$
    - $K'$  est la clé dérivée
      - $H(K)$ , si  $K$  est plus grande que la taille du bloc (déterminée par la fonction de hachage)
      - $K$ , dans le cas contraire
    - $opad$  est une constante de la taille du bloc répétant le byte  $0x5C$
    - $ipad$  est une constante de la taille du bloc répétant le byte  $0x36$
  2. Bob calcule et vérifie  $\text{HMAC}(K, x') = \text{HMAC}(K, x)$
  - Proposé par Bellare, Cannaeti, Krawczyk en 1996
  - Adopté comme standard par le NIST en 2002
  - Plus sécuritaire que le protocole « simplifié » présenté antérieurement



# Signature numérique

- Objectifs
  - Authenticité :
    - Pouvoir prouver qu'un document électronique a bel et bien composé et "signé" par son préputé auteur.  
=> Il ne doit pas être possible pour personne de falsifier la signature d'autrui.
  - Intégrité :
    - Pouvoir prouver que le document n'a pas été modifié depuis qu'il a été signé par son auteur légitime.  
=> Il ne doit pas être possible pour une autre personne que l'auteur de changer le document après sa signature sans violer la condition d'authenticité.
  - (Non-répudiabilité)
    - Empêcher qu'un auteur légitime puisse *a posteriori* nier qu'il est l'auteur et signataire d'un document qu'il a bel et bien signé  
=> Il ne doit pas être possible de "répudier" une signature faite par soi-même



# Signature numérique avec crypto à clé publique

- Signature
  - Pour signer un message  $x$ 
    1. Ajouter au message un préambule  $T$ , p.ex.  
*"Le document qui suit a été signé par José M. Fernandez, en date du ..."*  
 $x' = T \parallel x$
    2. Utiliser la clé privée  $d$  pour produire la version signé  $y$  du document en utilisant la clé privée et l'algorithme de déchiffrement:  
 $y = D(x', d)$  p.ex.  $y = (x')^d \bmod N$  avec RSA
  - Vérification
    - Pour vérifier un document  $y$  :
      1. Utiliser l'algorithme de chiffrement avec la clé publique  $e$  du présumé auteur pour obtenir  $x' = E(y, e)$
      2. Vérifier si  $x'$  est bel et bien un message "légitime" (bien formaté, a un préambule, qui a du sens, etc.). Si oui, accepter la signature.
  - Notes
    - *Pourquoi un préambule?*
      - Parce qu'il est possible pour un malfaiteur de falsifier une signature sur un message aléatoire ("garbage"), mais il ne lui est pas possible de le faire sur un message déterminé de son choix (p.ex. ayant un préambule raisonnable en français)
    - *Authenticité de la clé publique ?*
      - Comment s'assurer que le vérificateur a la bonne clé publique  $e$  qui correspond vraiment à l'auteur ?

# Signature numérique avec hachage

- Signature
  - Pour signer un message  $x$ 
    1. Calculer le haché  $h(x)$  du message avec une fonction de hachage cryptographique
    2. Utiliser la clé privée  $d$  pour  $h(x)$  comme avant pour obtenir la signature
      - $\text{sig}(x) = D(h(x), d)$
    3. Le document signé contient :  $(x, \text{sig}(x))$
  - Vérification
    - Pour vérifier un document  $(y, s)$ 
      1. Calculer le hachage  $h(y)$  de  $y$
      2. Obtenir la valeur  $h'$  en chiffrant la signature  $s$  avec la clé publique  $e$ ,  
$$h' = E(s, e)$$
      3. Accepter la signature si  $h' = h(y)$
    - Avantages
      - Plus rapide
      - La signature est indépendante du message lui-même

# Échange de clés – Diffie-Hellman

- Objectifs
  - Alice et Bob n'ayant pas échanger de clés auparavant désirent établir un canal privé
- Conditions et préalable
  - Ils ont accès à un canal "public" (non sécurisé)
  - Ils peuvent s'authentifier mutuellement (p.ex. par la voix)
- Protocole de Diffie-Hellman
  - Se base sur la difficulté du log discret
  - Permet à Alice et Bob de générer une clé dans  $[0..p-1]$  connue de personne d'autre
  - En l'absence d'authentification ...
    - ➔ Vulnérable aux attaques "man-in-the-middle"



# Échange de clés - Diffie-Hellman (suite)



Alice



Bob

Choisi

- $p$  premier
- $g$  générateur de  $Z_p^*$
- $a$  aléatoire dans  $[1..p]$

$p, g, g^a \text{ mod } p$

Calcule

- $K = (g^b)^a \text{ mod } p$

$g^b \text{ mod } p$

Choisi

- $b$  aléatoire dans  $[1..p]$

Calcule

- $K = (g^a)^b \text{ mod } p$

Données  $x$  chiffrées par  $E_K(x)$



# Cryptographie quantique

- Algorithme d'échange de clé
  - Proposé en 1984 par Brassard et Bennett
  - Principe de base
    - 1 bit codé sur un seul photon
- Avantages
  - Sécurité basée sur la mécanique quantique
    - Les propriétés quantiques d'un photon ne peuvent pas être reproduites parfaitement
    - Toute observation du photon pour extraire de l'information le détruit
      - ➔ Est « post-quantique » de façon démontrable
- Désavantages
  - Nécessite une infrastructure dédiée spéciale
    - Réseau de fibre optique
    - Liens satellite par laser
    - Cher à implémenter
- Situation actuelle
  - Course aux armements : crypto quantique par satellite



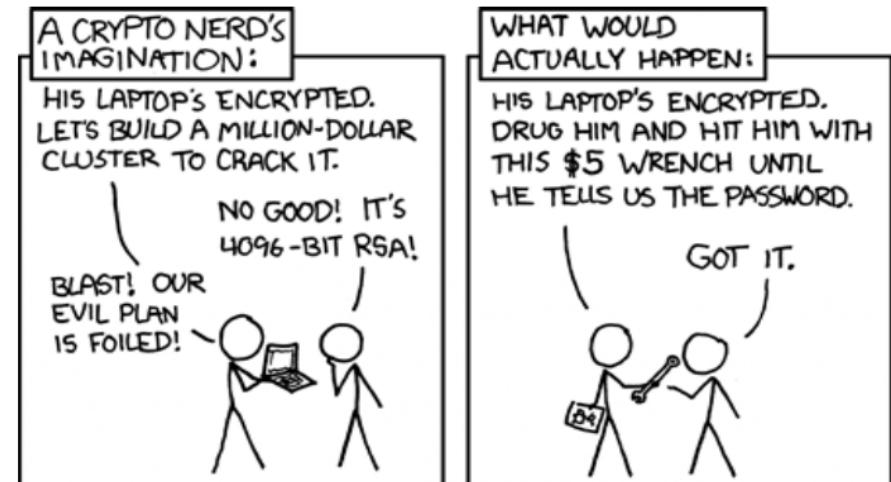
# Attaques cryptographiques

- Types d'attaques  
(en fonction des données disponibles pour l'attaquant) :
  - Texte chiffré seulement (*known ciphertext*)
    - uniquement accès au texte chiffré sans pouvoir chiffrer des messages (p.ex. document chiffré intercepté sur le réseau)
  - Texte connu (*known plaintext*)
    - accès au texte en clair en plus du texte chiffré (p.ex. interception d'un document chiffré qui sera diffusé publiquement plus tard comme un discours). On vise à trouver la clé pour déchiffrer les futurs messages
  - Texte choisi (*chosen plaintext*)
    - capacité de choisir le texte en clair en plus du texte chiffré (p.ex. on a accès à un clavier qui permet d'écrire des messages qu'on intercepte sous forme chiffré)
  - Texte chiffré choisi (*chosen ciphertext*)
    - Accès à un texte chiffré, et l'algorithme de chiffrement (p.ex. hachage de mot de passe sur un PC). On vise à chiffrer du texte connu et à obtenir le texte chiffré
- Typiquement, texte chiffré seulement



# Attaques cryptographiques

- Plusieurs attaques possibles :
  - Attaque contre l'algorithme
  - Attaque contre l'implémentation
  - Attaque contre l'opération
- En pratique, on voit plus souvent les attaques contre les implémentations et les opérations
- Une mauvaise utilisation de la cryptographie peut introduire la possibilité d'un attaque contre l'algorithme



<http://xkcd.com/538/>



# Attaques cryptographiques

- Force brute (*rappel*)
  - Attaque de type « texte chiffré seulement »
  - Procédure
    - On déchiffre le texte chiffré avec la clé  $n$
    - On effectue un test sur le message déchiffré pour voir s'il s'agit d'un vrai message (ex. on calcule l'entropie du texte déchiffré, si elle est très faible, sûrement un message en clair)
    - Si le test indique qu'on a déchiffré le message, la clé recherchée est la clé  $n$
    - Sinon, on recommence avec la clé  $n+1$
  - En bref, on essaie toutes les possibilités de clés
  - Temps pour réaliser l'attaque dépend de la taille de clé et de la capacité de traitement
  - Lorsqu'on calcule le temps pour casser la clé, on calcule le pire cas (on essaie toutes les clés), si la distribution des clés est uniforme sur l'espace des clés, le temps moyen sera  $t_{\max}/2$



# Attaques cryptographiques

- Attaque dictionnaire
  - Attaque de type « texte clair choisi »
  - On construit un dictionnaire de symboles qui sont susceptibles d'être émis par la source
  - On essaie chacun des mots du dictionnaire jusqu'à ce qu'on trouve (ou non) une correspondance avec le mot chiffré
  - Selon Turing, on peut échanger de l'espace mémoire pour du temps et bâtir des tables de correspondance
  - Le calcul des tailles de tables de correspondance similaires au calcul de force brute
  - Efficace contre les chiffrement par bloc
  - Divers mécanismes pour diminuer la vulnérabilité aux dictionnaires
    - Grand espace de possibilité de message « uniformément couvert »
    - Codage avec compression et bourrage aléatoire
    - Utilisation de sel (salt)



# Complexité de calcul des attaques

- Chiffrement symétrique
  - Chiffrement/déchiffrement:  $O(n)$  ou  $O(n^2)$
  - Force brute :  $O(2^n)$
  - Attaque par dictionnaire :
    - $O(M) = O(2^{H(S)})$  où  $M$  est la taille du dictionnaire
  - Attaque quantique (algorithme de Grover) :  $O(2^{n/2})$ 
    - Il faut « juste » doubler la taille de clé pour obtenir le même niveau de sécurité « post quantique »
- Chiffrement à clé publique
  - RSA/EI Gamal/ECC
    - Chiffrement/déchiffrement :  $O(n^3)$
    - Force brute :  $O(2^n)$
    - Attaque par dictionnaire :  $O(2^{H(S)})$
    - Algorithmes factorisation/log discret « classique »:  $O(2^{n/3})$
    - Attaque quantique (algorithmes de Shor):  $O(n^3)$
- Normalement on doit intégrer la loi de Moore dans les calculs
  - Chaque 18 mois on double la puissance de calcul  
(mais on arrive peut-être à la fin...)

# Attaques cryptographiques

- Attaques sur l'implémentation souvent causées par la présence d'une des fautes mortelles en crypto :
  - Cryptographie propriétaire ou « maison »
  - Mauvais codage
  - Mauvaise gestion des clés en mémoire ou persistance des clés en mémoire (ex. attaque « cold boot »)
  - Mauvais génération de clé (ex. Debian OpenSSL)
  - Mauvaise source d'entropie dans le système
    - El-Gamal
    - Vecteur d'initialisation pour les algorithmes de flux
    - Challenge-response (à voir dans la section d'authentification)

# Principe de gestion de clés

- Générations de clés
  - Nécessité de source de bit parfaitement aléatoire
  - Méthode matériel vs. logiciel vs. "manuel"
  - "Souveraineté" et contrôle sur la génération des clés
  - Difficulté technique pour certains algorithmes
    - RSA :  $p$  et  $q$  premier, etc.
    - El-Gamal :  $p$  t.q.  $p-1$  a un grand facteur, etc.
- Gestion des clés et réduction de risque
  - Possibilité de révocation
    - Distribution au préalable
    - Contrôle positif (déttection de perte ou vol)
- Mécanisme de protection
  - Contrôle d'accès
  - Chiffrement des clés par mot de passe ou phrase de passe
- Principe de segmentation
  - Clés de réseaux vs. clés point-à-point
  - Durée de vie limitée des clés
- Distribution de clés
  - Nécessite de canaux privés dédiés
    - Distribution physique
    - Utilisation de KEK (key-encryption keys) ou équivalent



# Réductions de risque et principes de bases (« cheatsheet »)

- Souveraineté de clé
  - Entropie
  - Contrôle d'accès
- Principe de Kerchoff
  - Pas de « sécurité par obscurité »
- Professionalisme
  - Algorithmes
  - Protocoles
  - Implémentations
- Gestion de clés
  - Segmentation
    - Temps,
    - Systèmes/réseaux,
    - Niveau de classification
    - compartiment “verticaux”
  - Mécanismes de confiance (PKI)
    - Hiérarchique
    - Décentralisé
  - Révocation



POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE

# INF4420: Éléments de Sécurité Informatique

Autorisation, contrôle d'accès



# Contenu du cours

- Introduction au contrôle d'accès
- Contrôle d'accès sous LINUX
- Modèles DAC et MAC
- Modèle RBAC
- Modèle ABAC
- Introduction à l'IAM



# Contrôle d'accès

- Contrôle d'accès
  - Définition : Fonction permettant de limiter l'accès à des ressources aux individus/machines/entités qui ont le droit d'accéder à ces ressources
  - S'applique autant à des objets physiques qu'à des ressources informatiques

# Introduction au contrôle d'accès

- 4 Aspects
  - Identification : Déterminer l'identité du demandeur
  - Authentification : Validation de l'identité du demandeur
  - Autorisation : Validation du droit d'accès aux ressources
  - Audit/(« Accounting ») : Attribution d'actions à une identité  
→ AAA (Authentication, Authorization and Accounting) ou IAAA
- Dans un système d'exploitation
  - Identification : nom d'utilisateur, identificateur de processus (PID)
  - Authentification : commande d'authentification
  - Autorisation: matrice d'accès, contrôleur de référence
  - Audit : journaux (« logs »)

# Introduction au contrôle d'accès

- Contrôleur de référence (« Reference Monitor »)
  - Composant qui s'interpose entre **tous les accès** de sujets à objets
  - Vérifie chaque demande d'entrée selon une procédure stricte
  - Maintient la sécurité au niveau voulu
  - S'implémente dans la noyau du système d'exploitation (OS)
  - Sujet = Utilisateur ou processus
  - Objet = Processus ou ressource (fichier)
  - Modes d'accès = { R-Read, W-Write, X-Execute }
  - Input: requête d'accès (sujet, objet, mode d'accès)
  - Output: Réponse (oui ou non) selon que l'accès est permis ou pas



# Introduction au contrôle d'accès

- Matrice d'accès

- Matrice qui liste les sujets (lignes) et objets (colonnes) dans un système, et les modes d'accès pour chaque (sujet, objet) (case de la matrice)

Sujet\Objet	File 1	File 2	Process 1	Process 2
Process 1	-	R	R,W,X	-
Process 2	-	X	R	R,W,X
User 1	R,X	W	-	R,X

- Pour un ordinateur avec A sujets et B objets, la matrice d'accès aura une taille de  $A \times B$ . La majorité de cellules seront vides !



- Listes de contrôle d'accès (ACL)

- Prendre chaque colonne de la matrice d'accès pour chaque sujet non-vide
- Stocker la liste d'accès avec l'objet
- Pour chaque accès à l'objet, le contrôleur de référence vérifie si le sujet a les droit requis

Sujet\Objet	File 1	File 2	Process 1	Process 2
Process 1	-	R	R,W,X	-
Process 2	-	X	R	R,W,X
User 1	R,X	W	-	R,X

# Contrôle d'accès Linux

- Modèle de sécurité Linux

- Toutes les ressources sont des objets (fichier, répertoire, mémoire, IO)
- Chaque objet a un propriétaire
- L'administrateur peut ajouter de nouveaux utilisateurs, lire et changer tous les objets, et changer les droit d'accès de tous les objets.
- Les utilisateurs peuvent seulement accéder aux objets pour lesquels ils ont la permission, et peuvent seulement changer les droits d'accès des objets dont ils sont propriétaires
- Les logiciels s'exécutent avec les droits de l'utilisateur qui a lancé le programme



# Contrôle d'accès Linux

- Utilisateurs
  - User ID (UID) pour chaque utilisateur
  - UID 0 est réservé pour l'administrateur (root)
  - Les fichiers ont l'ID de l'utilisateur qui a créé le fichier
- Groupes
  - Group ID (GID)
  - Les utilisateurs ont un groupe principal
  - Les utilisateurs peuvent joindre d'autres groupes
  - Les fichiers ont le groupe principal de l'utilisateur qui a créé le fichier
- Utilisateurs et groupes ne sont pas des objets



# Contrôle d'accès Linux

- Chaque fichier a un UID et GID assigné
- Chaque programme a un UID et GID assigné
- Avant d'exécuter un appel de fonction du système (« system call »), le contrôleur de référence vérifie :
  - Si  $UID = 0$ , permettre l'accès
  - Sinon, lire la liste de contrôle d'accès de l'objet et vérifier si l'accès est permis



# Contrôle d'accès Linux

- Permissions de fichiers
  - R (lire)
  - W (écrire/changer)
  - X (Exécuter)
- Pour
  - U (Propriétaire)
  - G (Groupe)
  - O (Autres utilisateurs)

```
-rw-r----- 1 Emilie profs 7627 Oct 1 12:50 exam
-rw-rw-rw- 1 root      root 12987 Sep 7 19:34 /etc/passwd
```



# Contrôle d'accès Linux

- Changement du propriétaire d'un objet
  - Commande chown
  - Exemple : chown patrick exam
    - Patrick devient le nouveau propriétaire du fichier exam
  - Seul l'administrateur root pour exécuter un chown



# Contrôle d'accès Linux

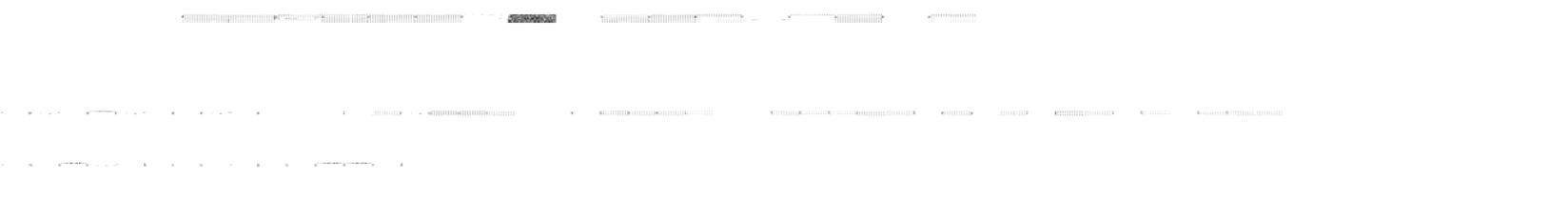
- Changement des droits sur un objet
  - Commande chmod
  - Seul le propriétaire et l'administrateur peuvent changer les droits
  - Exemple : chmod u=rwx, g=rx, o=r myfile
  - Commande équivalente à : chmod 754 myfile
    - read = 4
    - write = 2
    - execute = 1
    - pas de permission = 0
  - Ajout de droit : chmod g+w myfile
  - Retrait de droit : chmod o-r myfile

# Contrôle d'accès Linux

- Sticky bit
  - Pas d'effaçage du fichier
- setuid
  - Le programme s'exécute avec les permissions du propriétaire
  -
- setgid
  - Le programme s'exécute avec les permissions d'un utilisateur dans le groupe du propriétaire

# Contrôle d'accès Linux

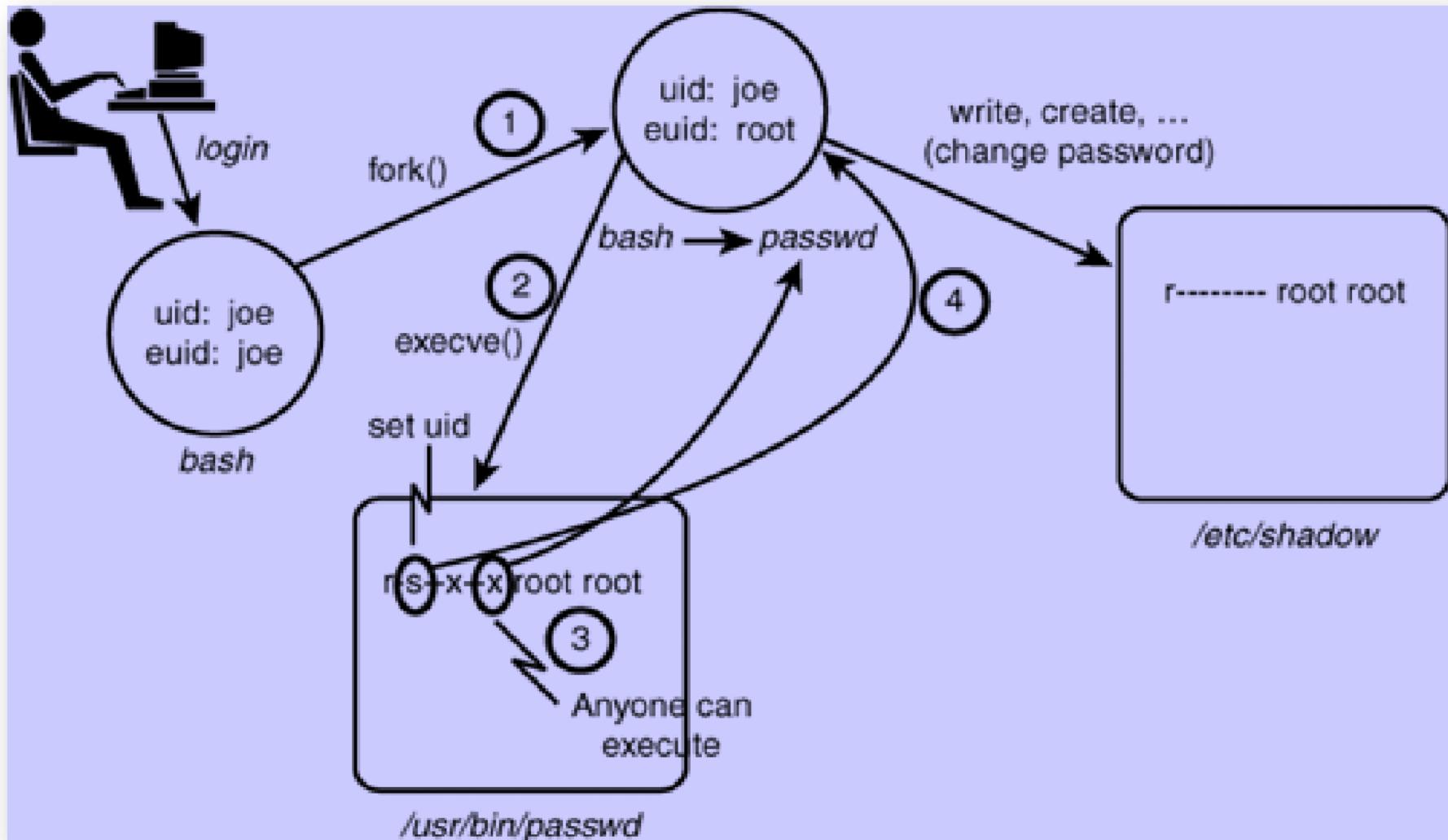
- Les mots de passe sur Linux se changent en utilisant la commande /usr/bin/passwd
- Les utilisateurs peuvent changer leur mot de passe
- Root peut changer le mot de passe de tous les utilisateurs



- Comment est-ce qu'un utilisateur peut changer son mot de passe sans permission d'écriture à /etc/shadow ?



# Exemple setuid





# Limites de DAC

- Linux utilise le Contrôle d'accès discrétionnaire (DAC)
  - Les utilisateurs peuvent changer les permissions de leur fichiers  
`chmod 655 /home/david/declaration_impot_16`
- DAC représente les droits sous forme de matrice
- DAC fonctionne correctement sous 2 conditions
  - **Si les usagers ne font pas d'erreurs**
  - **Si on peut faire confiance à tous les programmes**  
**➔ Impossible !!!**

	Jean	Paul	Marie
File1	<b>rw</b>	<b>r</b>	<b>w</b>
File2	<b>r</b>	-	<b>rw</b>
File3	<b>r</b>	<b>w</b>	-



# Limites du modèle DAC

- Exemple de matrice

	Dossier médical	Ordonnance
Médecin	RW	RW
Patient (Attaquant)	-	R

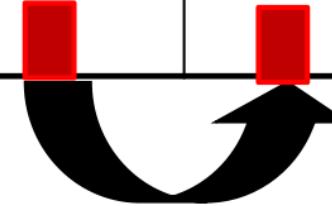


# Limites du modèle DAC

	Dossier médical	Ordonnance
Médecin	RW	RW
Patient (Attaquant)	-	R
Application piégée	RW	RW

Application  
s'exécutant pour le  
compte du médecin  
(hérite des droits du  
médecin dans DAC)

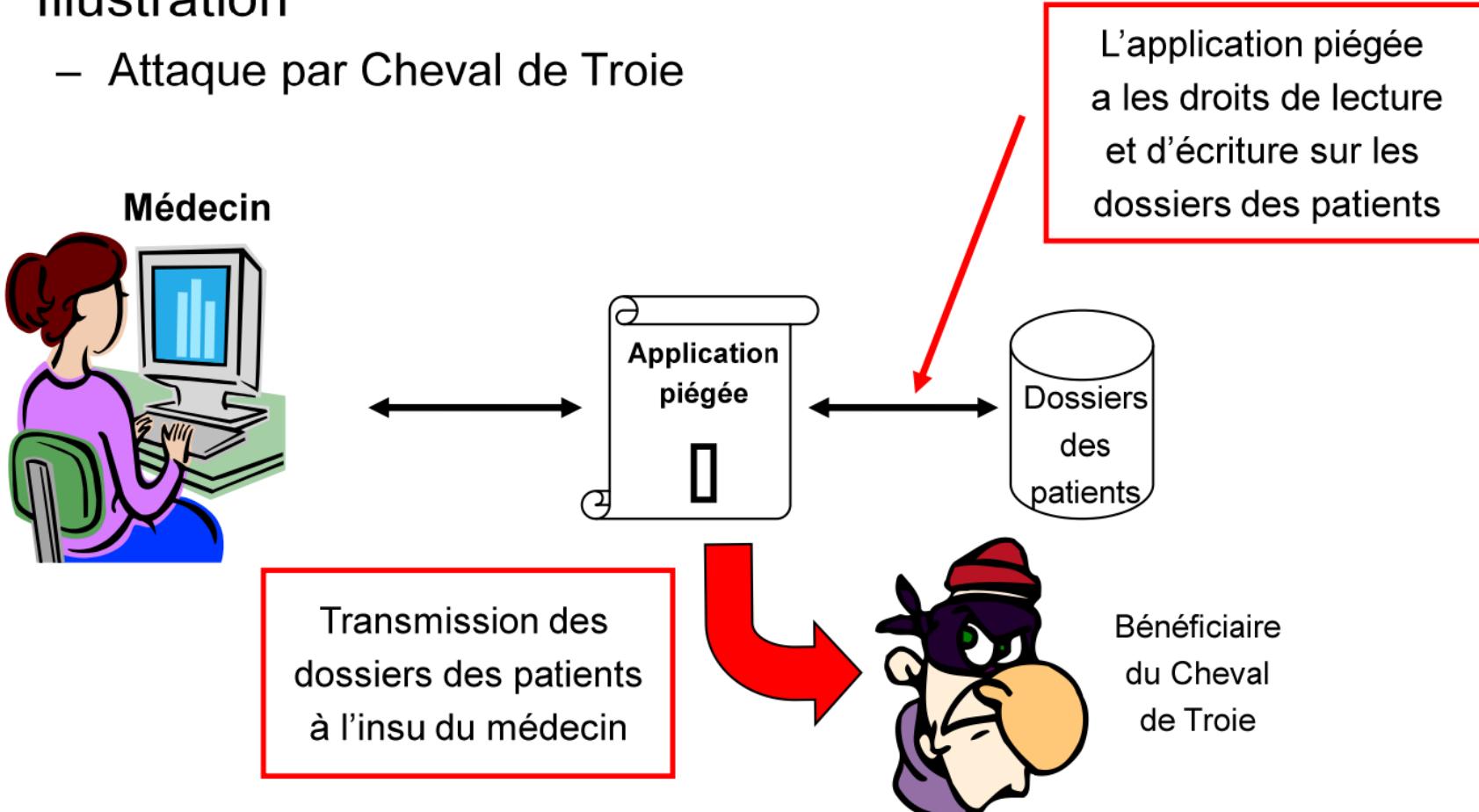
Transfert illégal non  
contrôlé par DAC





# Limites du modèle DAC

- Illustration
  - Attaque par Cheval de Troie





- MAC = Mandatory Access Control
  - Contrôle d'accès obligatoire
- MAC ajoute des étiquettes à chaque sujet et objet
- Une politique d'accès contient les règles d'accès permises pour chaque sujet et objet
- Politique par défaut
  - REFUSÉ (« Deny ») : l'accès n'est pas permis, à moins qu'il y ait une règle indiquant le contraire dans la politique d'accès
- Les règles et étiquettes peuvent seulement être changées par un administrateur avec un logiciel de confiance (« trusted »)

# Politique de sécurité multiniveau

- Exemple classique de contrôle d'accès MAC
- Etiquette = niveau de sécurité
- Exemple

Public  $\leq$  Confidentiel  $\leq$  Secret

# Politique de sécurité multiniveau

- Les utilisateurs reçoivent un niveau d'habilitation
  - Les utilisateurs s'engagent à ne pas diffuser n'importe comment les informations qu'ils détiennent
- Les informations reçoivent un niveau de classification
  - Mesure la confidentialité de l'information



# Conditions de sécurité (Modèle de Bell & LaPadula)

- No Read Up
  - Un sujet s peut lire un objet o si :
    - $\text{niveau\_classification}(o) \leq \text{niveau\_habilitation}(s)$
- No Write Down
  - Un sujet s peut modifier un objet o si :
    - $\text{niveau\_habilitation}(s) \leq \text{niveau\_classification}(o)$



# Conditions de sécurité (suite)

- Objectif du « No write down »
  - Soit un programme s'exécutant au niveau « Secret »
  - Ce programme peut lire des données classées « Secret »
  - Mais le « No write down » empêche un éventuel piège contenu dans ce programme de transmettre les données lues vers un utilisateur qui ne serait pas habilité au niveau « Secret »

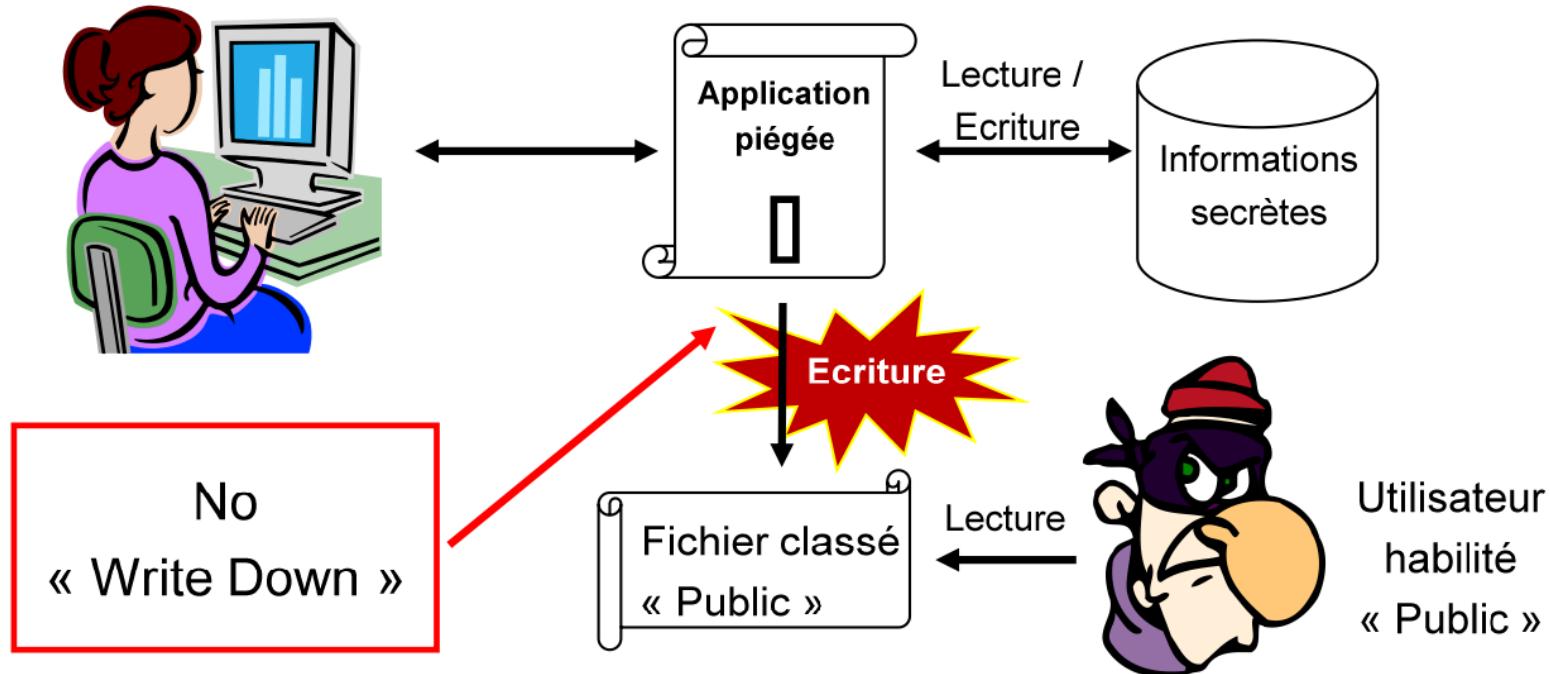


# Conditions de sécurité (suite)

Utilisateur habilité

« Secret »

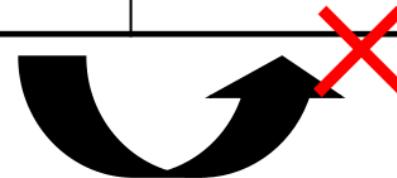
(Médecin)





# Conditions de sécurité (suite)

	Dossier médical (classé S)	Ordonnance (classé P)
Médecin (habileté S)	RW	RW
Attaquant (habileté P)	-	R
Application piégée (niveau S)	RW	R W



Transfert illégal bloqué  
par Bell et LaPadula



# Conditions de sécurité (suite)

	Dossier médical (classé S)	Ordonnance (classé P)
Médecin (habileté S)	RW	RW
Attaquant (habileté P)	-	R
Application piégée (niveau courant S)	RW	<del>RW</del>
Application piégée (niveau courant P)	<del>RW</del>	RW

Intérêt du niveau courant : le médecin doit travailler au niveau P pour pouvoir écrire l'ordonnance



- Conditions de Bell & LaPadula trop rigides
- Aujourd'hui utilisation d'un autre modèle MAC
  - DTE (Domain Type Enforcement)
  - Implémenté dans SELinux (Security Enhanced Linux)



# Pourquoi RBAC ?

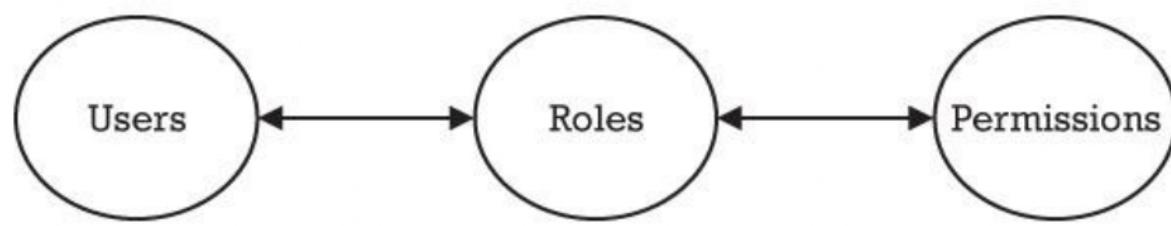
- DAC est de gestion difficile car chaque usager est un cas individuel
  - Considérez des compagnies de milliers d'employés
- DAC suppose que les usagers sont propriétaires des ressources et peuvent transférer les droits sur elles,
  - Tandis que normalement c'est l'organisation qui est propriétaire des ressources, et veut en retenir le contrôle



- RBAC = Role Based Access Control
- RBAC est basé sur deux points
  - Le fait que dans les organisations les employés sont affectés à des rôles
    - Comptable, programmeur, docteur, infirmière, technicien ...
    - Les rôles sont organisés en **hiérarchies**
  - Le fait que chaque employé, pour exécuter son rôle, a besoin de certaines permissions



- RBAC s'appuie sur la notion organisationnelle de rôle pour associer des permissions de sécurité aux différents rôles
- Le rôle devient un mécanisme pour associer des permissions aux usagers





## Usagers



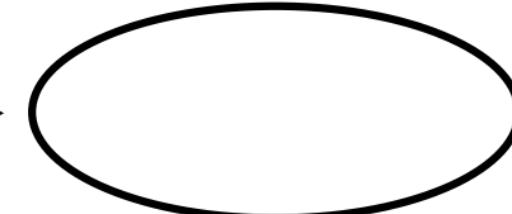
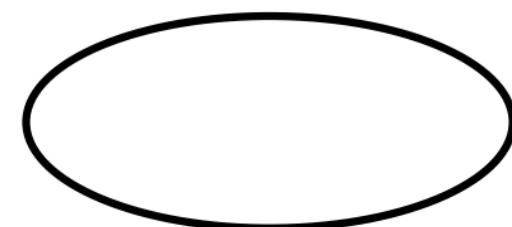
Cette affectation  
peut changer souvent

## Rôles



Cette affectation  
ne change pas souvent

## Permissions





# RBAC

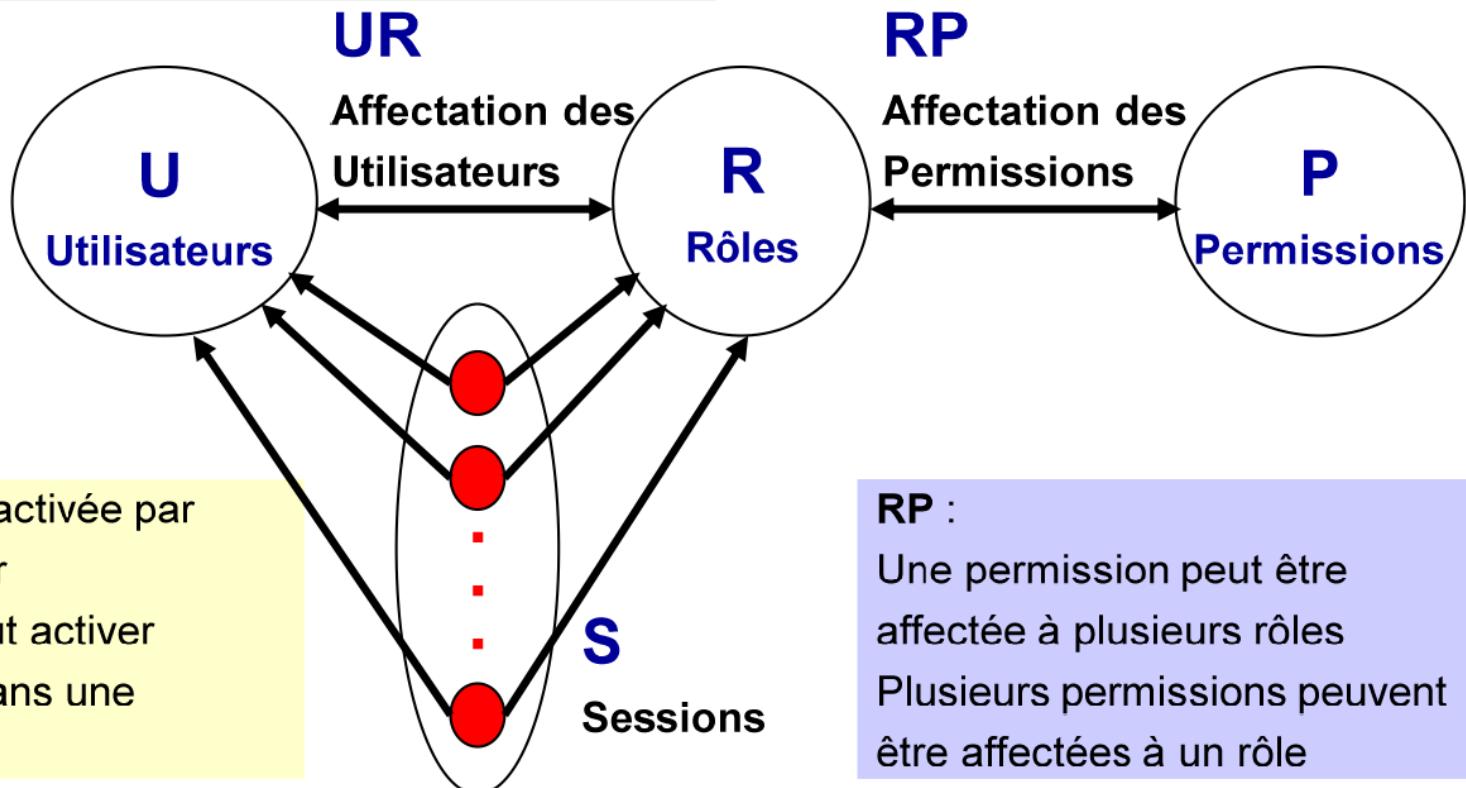
## Concept de session

- Pour utiliser ses rôles, un usager doit activer des sessions
- Une session est un processus qui agit pour un usager
  - En changeant de session, un usager peut activer de nouveaux rôle(s)
  - P.ex. un employé de banque Paul peut activer un rôle quand il est aux prêts et un autre rôle quand il est aux investissements
- Pour accéder à une session, un sujet doit s'authentifier
- Un sujet peut se trouver dans plusieurs sessions simultanément



## UR :

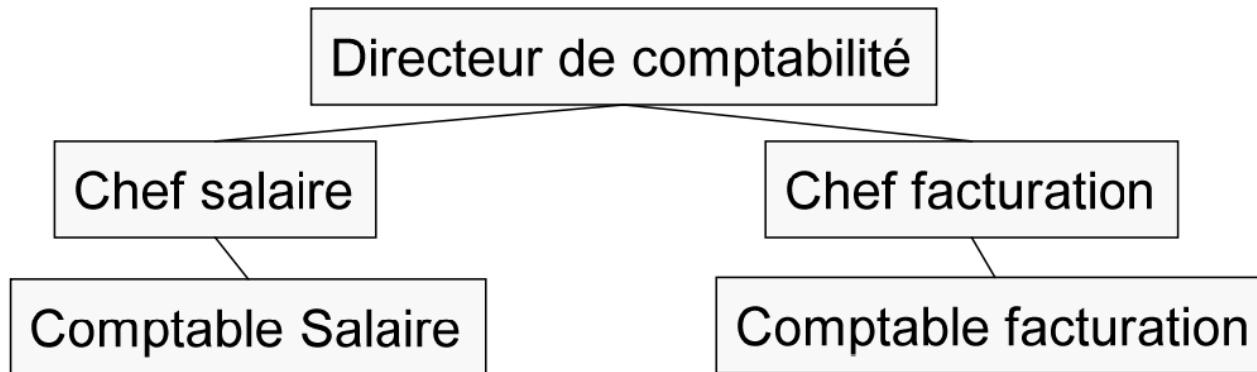
Un rôle peut être affecté à plusieurs utilisateurs  
Plusieurs rôles peuvent être affectés à un utilisateur



Une session est activée par un seul utilisateur  
Un utilisateur peut activer plusieurs rôles dans une session

# RBAC Hiérarchique

- On peut introduire une hiérarchie de rôles
- Propriété de cette hiérarchie
  - Héritage des permissions



- Si Comptable Salaire a une permission, alors le Chef Salaire et le Directeur de Comptabilité l'ont aussi

# RBAC avec contraintes

- Les contraintes sont un élément extrêmement important de RBAC
  - Les contraintes servent à empêcher certaines situations indésirables
- Exemple : contrainte de cardinalité
  - Il ne doit y avoir qu'un seul utilisateur affecté au rôle de directeur

# RBAC avec contraintes

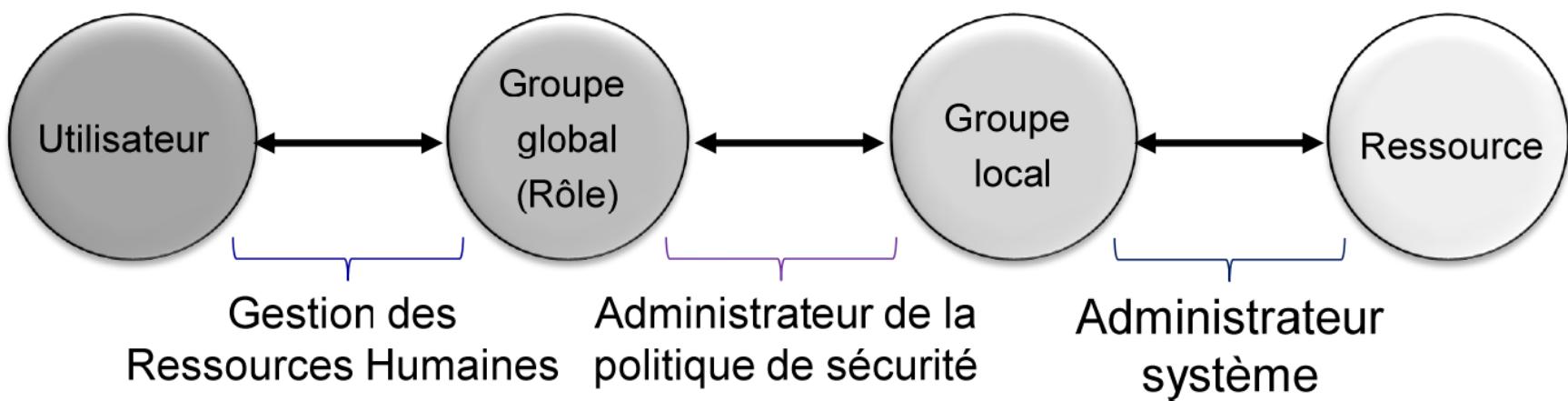
- Contraintes de « séparation des pouvoirs »
  - En anglais : « Separation of duty » (SOD)
  - Ce sont les contraintes les plus importantes
  - Exemple : Celui qui approuve un chèque (rôle R1) ne peut pas être celui qui le signe (rôle R2)
  - Dans RBAC, cela se traduit par une contrainte qui rend impossible qu'un utilisateur soit affecté à R1 et R2
- SOD statique (SSOD) ou dynamique (DSOD)
  - SSOD : la séparation s'applique en toute situation
  - DSOD : la séparation s'applique uniquement dans une même session



- AGLP
  - Access – Global – Local – Permissions
  - Implémentation de RBAC sous Windows
  - Repose sur les Active Directory(AD)
- Éléments
  - Groupe « Globaux »
    - regroupement des utilisateurs, typiquement selon leur rôles
    - généralement définis sur des serveurs de domaine globaux (d'où le nom)
  - Groupes « Locaux »
    - auxquels sont attribués des Permissions sur des ressources
    - généralement définis et résidants sur les serveurs où ces ressources se trouvent (d'où le nom)
    - les membres sont exclusivement des groupes globaux

- Principes de base

- On n'attribue pas de permissions aux groupes globaux ni aux utilisateurs
- On n'ajoute pas d'utilisateurs dans les groupes locaux
- Séparation des responsabilités
  - Gestion des usagers : U et G
  - Politique de sécurité : G et L
  - Administrateur de système : L et R





# ABAC

- ABAC = Attribute Based Access Control
- Dans ABAC, la décision d'accès dépend de politiques qui combinent entre eux des attributs
  - Attributs de l'utilisateur
  - Attributs de l'action
  - Attributs des ressources
  - Attributs liés à l'environnement
- Politique d'autorisation = ensembles de règles



# ABAC

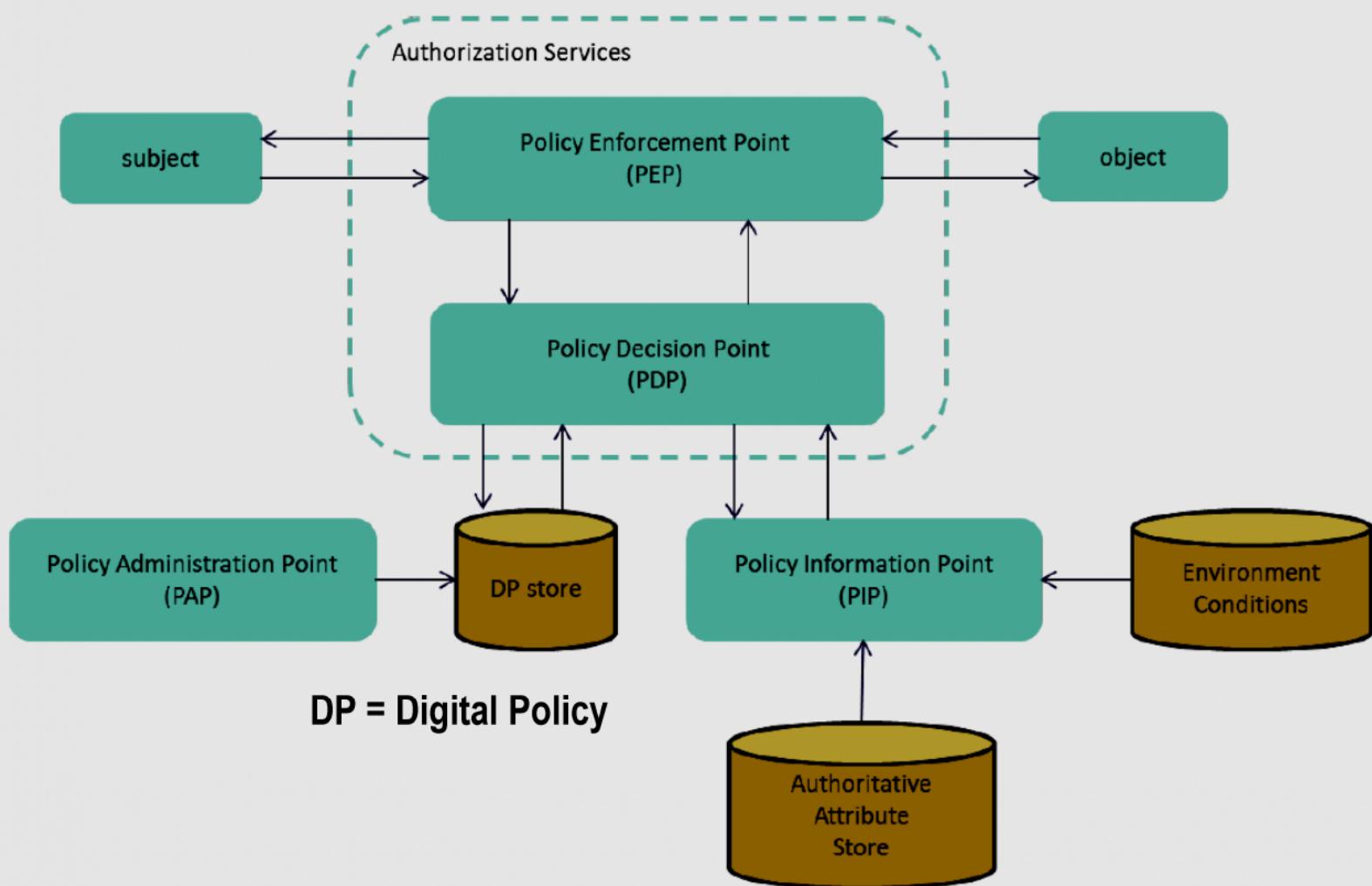
- L'intention de ABAC est d'être plus général et flexible que les modèles précédents

# Attribute Based Access Control

- Sujet, Ressource et Action sont des catégories qui regroupent des attributs
  - Attributs du Sujet : Nom, Département, Rôle, etc.
  - Attributs de l’Action : Ident, Type
  - Attributs de la ressource : Type, Ident, Auteur
- Les attributs ont des valeurs
  - Nom(Sujet)=Gervais, Département(Sujet)= GIGL, Role(Sujet)=Professeur,
  - Type(Ressource)=Reserve, Ident (Ressource)= QA.75.5.2005,
  - Ident(Action)=EmpruntLivre, Type(Action)=Bibliotheque.
- La requête de contrôle d'accès est un ensemble d'éléments (attribut(catégorie)=valeur) – les paramètres de la requête
  - Nom(Sujet)=Gervais et Ident(Action)=EmpruntLivre et Ident(Ressource)=QA.75.5.2005
- Les règles de contrôle d'accès sont basées sur des cibles exprimées par des expressions booléennes
  - Permettre si (Role(Sujet)=Professeur ou Role(Sujet)=Etudiant) et Ident(Action)=EmpruntLivre et Type(Ressource)=Reserve et  $7:00 \leq \text{Heure} \leq 20:00$



# ABAC Schéma architectural



Source: NIST Special Publication 800-162

# Éléments architecturels de ABAC

- PEP: Policy Enforcement Point
  - Donne ou refuse un accès
- PDP: Policy Decision Point
  - Prend la décision si l'accès doit être donné ou refusé
  - Utilise les politiques et règles qui sont enregistrées dans une base de données appelée Policy Store
- PIP: Policy Information Point - fournit les informations dont le PDP a besoin pour prendre ses décisions
  - Les valeurs des attributs
  - L'état de l'environnement:
  - L'environnement est aussi une catégorie avec ses attributs
    - L'heure et la localisation de l'usager ou de la ressource
- PAP: Policy Administration Point
  - Gère le Policy Store: ajout, suppression de règles



# ABAC : Exemple

- Le PEP reçoit la requête
  - (Marc) demande (d'emprunter) (le livre QA.75.5.2005) à (18:00)
- Le PEP informe le PDP qu'il a reçu cette requête
- Le PDP détermine que la règle applicable pourrait être :
  - Permettre (au Professeur) ou (à l'Etudiant) (d'emprunter) (un livre réservé) entre (7:00) et (20:00)
- Mais il ne sait pas si Marc est un professeur, qu'il est 18:00,....
- Le PDP interroge le PIP, le PIP consulte la base des attributs et informe le PDP que :
  - *Marc est un professeur titulaire*
  - *Un professeur titulaire est un professeur*
  - *Le livre QA.75.5.2005 a été réservé*
  - *il est 18:00*
- Le PDP conclut que la demande d'accès est *Permise*
- Le PDP en informe le PEP qui informe Marc



# XACML

- XACML
  - eXtensible Access Control Markup Language
  - Langage qui implémente le modèle ABAC
- Langage basé sur la syntaxe XML
- Norme OASIS
  - Organization for the Advancement of Structured Information Standards (<https://www.oasis-open.org/>)
  - Première version disponible en 2003
  - XACML v3 depuis 2013



# XACML en bref

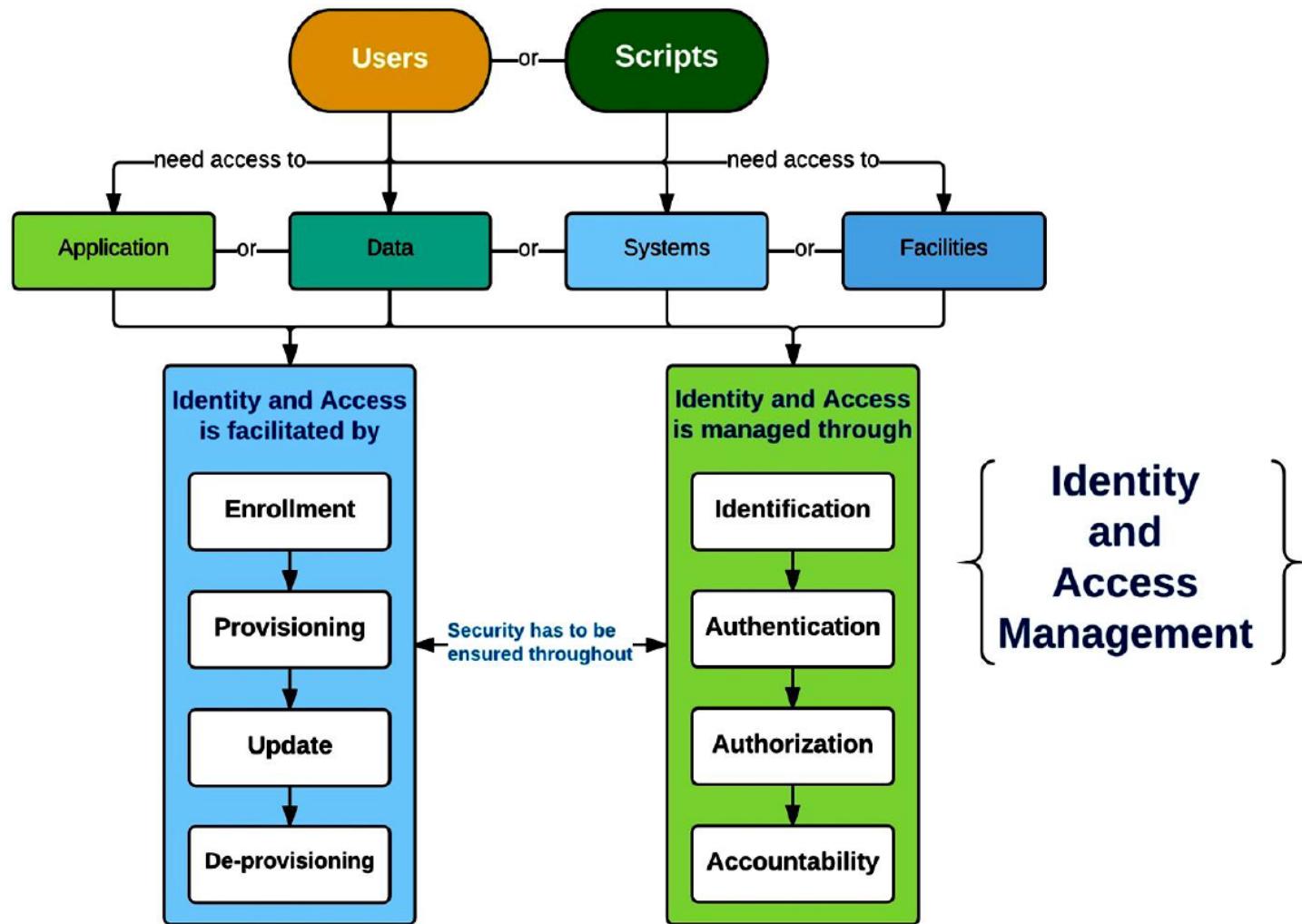
- Une architecture pour l'implémentation
- Des principes de communication entre les composants
- Un langage pour les règles et le politiques
- Un langage pour les requêtes et les réponses
- Types de données normalisés
- Fonctions et algorithmes de combinaison
- Extensibilité
- Différents profils
  - Pour RBAC, ...

# Gestion des identités et des accès

- IAM = Identity and Access Management
  - En français : GIA = Gestion des Identités et des Accès
- Principe de base de l'IAM
  - Ne pas mélanger les fonctions du système et le contrôle d'accès
  - Fonctions du systèmes = Services, Applications
- Séparer l'implémentation des applications et de la sécurité
  - Ne pas coder « en dur » l'authentification dans les applications
  - Ne pas coder « en dur » les autorisations dans les applications
- Bon principe pour exprimer, déployer et mettre à jour la politique de contrôle d'accès



# Fonctions principales de l'IAM





# Fonctions principales de l'IAM

- Le provisionnement
  - Déploiement statique de la politique de contrôle d'accès
  - En Anglais = User Provisioning
- Objectifs
  - S'assurer automatiquement que la politique de sécurité est effectivement appliquée dans les applications
  - En général, via l'exécution de script
- Fonctions principales du provisionnement
  - Créer, mettre à jour, supprimer automatiquement les comptes dans les applications et les systèmes cibles
  - Synchroniser les mots de passe entre les comptes applicatifs



# Fonctions principales de l'IAM

- La réconciliation
  - Compare l'état souhaité des comptes, décrit par la politique de sécurité, avec l'état réel existant dans les systèmes et les applications
  - De manière automatique périodique, ou suite à des modifications de la politique, ou à la demande
- Fournit des rapports de réconciliation indiquant les écarts
- Permet de traiter ces écarts manuellement ou automatiquement (avec précautions)
- Exemple
  - Le compte doit exister d'après la politique d'accès définie mais il n'existe pas dans le système cible
  - Le compte est activé dans le système cible, alors que l'autorisation est désactivée



# Autres fonctions de l'IAM

- Gestión des identités
  - Gestión de l'annuaire des utilisateurs
  - Gestión du SSO (Single Sign On)
- Gestión des autorisations
  - Expression de la politique d'autorisation
    - RBAC, ABAC, ...
  - Policy Mining
    - Extraction de la politique à partir des comportements observés
  - Supervision de la politique
    - Détection des comportements anormaux et des anomalies de déploiement
    - Exemple : Compte « dormant » qui n'a pas été utilisé depuis 6 mois



POLYTECHNIQUE  
MONTRÉAL

UNIVERSITÉ  
D'INGÉNIERIE

A la semaine prochaine