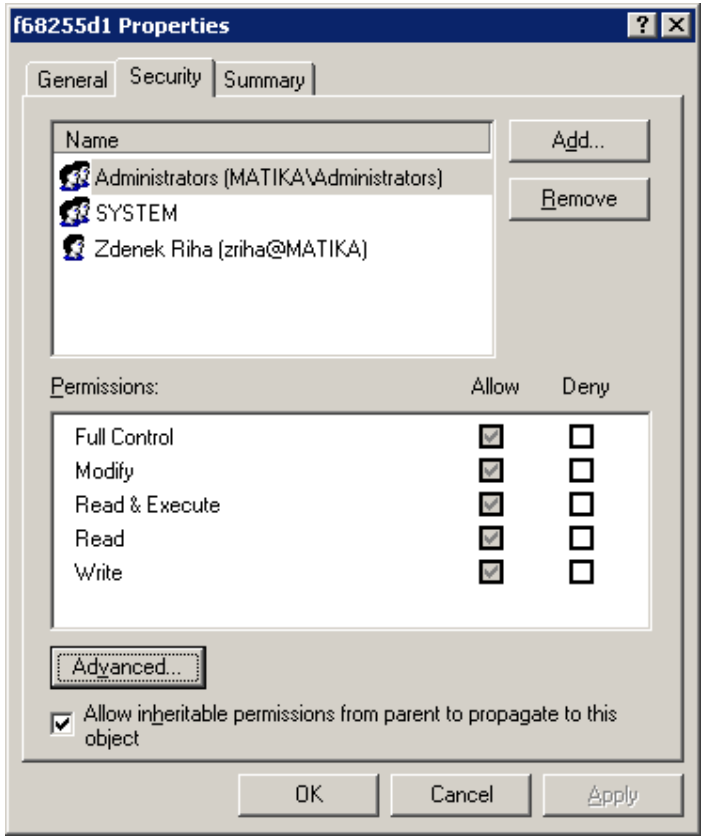
PV157 – Autentizace a řízení přístupu

# Řízení přístupu



# Řízení přístupu

- Funkce pro řízení, který subjekt (uživatel, proces …) má jaký přístup k určitému objektu (souboru, databázi, tiskárně …).
- Implementovaná bezpečnostním mechanismem řízení přístupu.
  - Hierarchie
    - ✓ Hardware
    - ✓ Operační systém
    - ✓ Middleware (např. databázový systém)
    - ✓ Aplikace (např. informační systém)
  - Předpoklad implementace
    - ✓ je bezpečně implementovaná funkce „Identifikace & autentizace“
- Kategorie mechanismů řízení přístupu
  - fyzické typicky ochrana off-line uložených archivních kopií
  - logické typicky ochrana on-line uchovávaných dat

# Pojmy

- **vlastník dat** (owner)
  - subjekt, statutární autorita odpovědná za daný typ dat,
  - za konkrétní data daného typu, za datový objekt

- **správce** (dat, objektu) (custodian)
  - subjekt, autorita pověřená odpovědností za bezpečnost konkrétních dat, za bezpečnost konkrétního objektu

- **uživatel** (user)
  také autorizovaný uživatel, oprávněný uživatel
  - subjekt, mající právo přístupu ke konkrétním datům, ke konkrétnímu objektu

# Typy omezení přístupu k objektu

- R, Read Only
  - ano: kopírování, prohlížení, tisk, ...
  - ne: rušení, vytváření, modifikace, …
- RW, Read/Write
  - ano: kopírování, prohlížení, tisk, … , rušení, vytváření, modifikace
- X, Execute
  - ano: řízení běhu procesu
  - ne: kopírování, prohlížení, tisk, … , rušení, vytváření, modifikace
- daný omezením
  - časem pouze v danou dobu
  - místem pouze z …
  - obsahem interval hodnot (bankovní automat)
  - typem služby e-mail ano, telnet ne

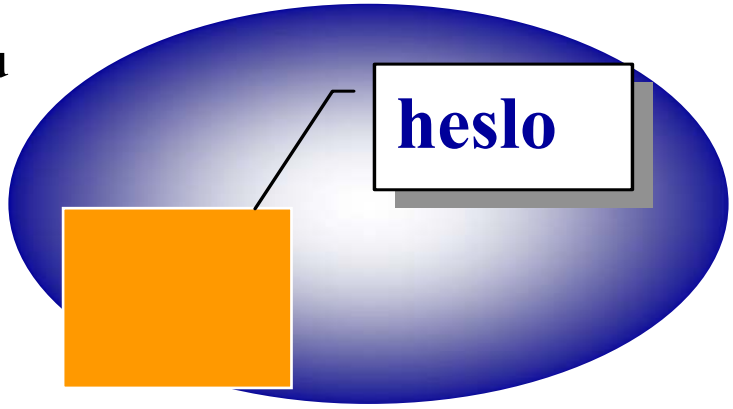
# Správa řízení přístupu – modely

- centralizovaná správa řízení přístupu
  - 1 správce všech objektů (IS, Informačního systému)
  - mnoho vlastníků, mnoho uživatelů
  - klad: přísné řízení, konzistentnost
  - zápor: vysoká (časová) režie v (distribuovaných) IS

- decentralizovaná správa řízení přístupu
  - objekt spravuje jeho vlastník – správce,
  - mnoho vlastníků – správců, mnoho uživatelů
  - klad: snadno dosažitelná vysoká odpovědnost
  - zápory:
    - ✓ obtížnost udržení konzistence komunikace mezi správci
    - ✓ není dostupný okamžitý celkový přehled stavu
    - ✓ hůře se prosazuje bezpečnostní politika IS

# Heslo

- heslo, šifrovací klíč
  - všeobecné
  - samostatné pro
    - čtení, modifikaci, rušení, …
- každý **objekt** má přiděleno(-a) svým vlastníkem heslo (hesla)
- právo přístupu má subjekt znající heslo
- použito například ke sdílení disků/adresářů v některých verzích MS Windows
- negativa:
  - nelze zjistit kdo všechno má právo přístupu
  - heslo musí být dostupné (uloženo v otevřeném tvaru nebo se slabým maskováním v programu, pom. souboru...)



# Matice přístupových práv

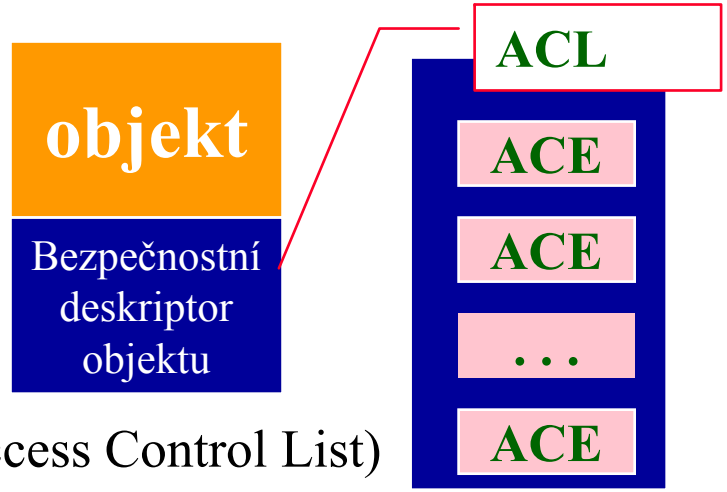
- Matice udávající přístupová práva subjektů k objektům

| | objektA | objektB | objektC |
|---|---|---|---|
| subjekt1 | rw | rx | – |
| subjekt2 | rwx | rwx | rwx |

# Matice přístupových práv

- Může být i trojrozměrná
- 3. rozměrem je program
- Máme trojice (subjekt, program, objekt) a jim odpovídající přístupová práva
  - (Alice, účetní program, účetní data) má práva {r,w}
  - (Alice, editor, účetní data) má práva {}
  - (Alice, editor, /etc/passwd) má práva {}
  - (Alice, passwd, /etc/passwd) má práva {r,w}
- Dvourozměrná i třírozměrná matice je v praxi tak velká, že je problém s ní efektivně pracovat (ukládat ji, vyhledávat v ní)
  - Tisíce až miliony objektů (souborů)
  - Tisíce až miliony subjektů (uživatelů)
- Matici můžeme ukládat po řádcích či po sloupcích

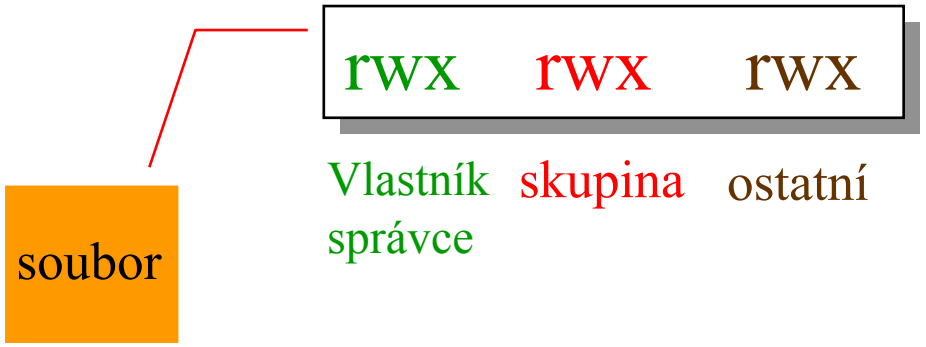
# Seznamy přístupových práv




- Ukládáme matici přístupových práv po objektech
- Matice je často řídká, proto ukládáme jen neprázdné prvky
- Seznam přístupových práv objektu – ACL (Access Control List)
- Element přístupových práv – ACE (Access Control Element)
  - prvek ACL
  - přidělení práv přístupu jednotlivému subjektu, skupině subjektů, …
- Výhoda seznamu přístupových práv
  - modifikaci/zpřístupnění lze omezit na jednotlivce ve skupině
- Nevýhoda seznamu přístupových práv
  - obtížná správa, neefektivní kontrola přístupových práv při přístupu k objektu
  - nesnadné zjistit k jakým objektům má určitý uživatel přístup (například při změně pracovního zařazení)

# Implementace ACL – UNIX

- Nelze zamítnout přístup jednotlivci ze skupiny vlastníka
- Ve starších verzích UNIXu mohl uživatel patřit do pouze jedné skupiny
- Nutnost vytvářet nové skupiny uživatelů (může jen root)
- Nelze zajistit důvěrnost:




110 100 000 originál
110 100 000 kopie člena skupiny

110 100 100 zrádný člen skupiny upraví

# SUID/SGID programy

- Matice přístupových práv jen dvourozměrná.
- Povolení práv pro určité programy je třeba řešit pomocí SUID/SGID bitů.
- Vlastník programu může nastavit, že program po spuštění bude běžet s právy uživatele/skupiny vlastníka.
- Například program pro změnu hesla potřebuje přístup zápisu do souboru /etc/passwd resp. /etc/shadow. Běžný uživatel však nemá k těmto souborům přístup pro zápis. Proto je program passwd nastaven jako SUID na uživatele root, který do těchto souborů zapisovat může.
- Přístup SUID/SGID není příliš intuitivní. Leniví programátoři píší hromadu programů, které musí běžet jako SUID root. SUID programy musí být napsány bezpečně, jejich vstupy (parametry, stdin, proměnné prostředí) nejsou důvěryhodné.

# Implementace ACL – moderní UNIX

- Administrátor (root, resp. uživatel s UID 0) má neomezený přístup ke všem objektům, může měnit libovolné soubory, upravovat logy apod.
- Nejen skutečný administrátor, ale i hacker apod.
- Snaha o omezení možností administrátora
  - Nové vlastnosti souborového sytému UNIXových verzí odvozených od Berkeley větve
    - ✓ Možnost nastavit dodatečné „flagy“ pro soubory
      - Append-only: lze pouze přidávat data – vhodné pro logy
      - Immutable: soubor není možné modifikovat – vhodné pro systémové soubory
      - Undeleteable – nesmazatelné
    - ✓ Možnost nastavit pro uživatele i skupiny
    - ✓ Nastavuje se při bootu, potom ani root nemůže provádět změny
    - ✓ Je i v Linuxu: chattr pro ext2/ext3 (lze však měnit kdykoliv)

# Atributy souborů v ext2/ext3

- Prohlížíme příkazem **`lsattr`**
- Nastavujeme příkazem **`chattr`**
- Atributy
  - A – čas posledního přístupu není aktualizován (vyšší výkon)
  - a – do souboru lze pouze přidávat data (nelze smazat nevhodný záznam např. v logovacím souboru)
  - c – soubor bude na disku komprimován
  - d – soubor nebude zálohován programem dump
  - i – soubor nemůže být nijak modifikován (bezpečnost)
  - j – určuje jakým způsobem se ukládají žurnálová data
  - s – při mazání souboru jsou uvolňované bloky vynulovány
  - S – při každém zápisu je provádí sync
  - t – zakazuje částečné fragmenty
  - u – i při smazání souboru se obsah ponechává (pro undelete)
- Atributy může nastavovat jen administrátor (root), může je kdykoliv jakkoliv změnit, proto je jejich význam pro bezpečnost jen omezený

# Implemetace ACL – moderní UNIX

- Jen trojice rwx pro vlastníka, skupinu a ostatní není dostatečně jemné
- Moderní UNIXové systémy se snaží tyto trojice doplnit skutečnými ACE
  - Tyto ACE obsahují výjimky ke standardním přístupovým právům (výše uvedené trojici)
  - Můžeme tak odebrat právo jednotlivci ve skupině, případně přidat právo jinému uživateli
- Tento přístup implementují mnohé komerční UNIXové systémy (např. VAX, …) a dnes i většina open source systémů
- ACL pro souborový systém extX se stalo standardní součástí jádra Linuxu >2.5

# POSIX ACL

- Klasická práva pro vlastníka (ACL_USER_OBJ), skupinu (ACL_GROUP_OBJ) a ostatní (ACL_OTHER) zůstávají
- Nově je možné přidávat výjimky pro
  - uživatele (ACL_USER)
  - skupiny (ACL_GROUP)
- a nastavit masku (ACL_MASK), která dále omezuje práva udělená uživatelům (ACL_USER) a skupinám (ACL_GROUP_OBJ a ACL_GROUP) [vlastníka se netýká]
- a u adresářů lze nastavit defaultní ACL pro nově vytvářené objekty
- ACL zapisujeme ve formě typ_subjektu:subjekt:práva
  - např. user:zriha:rw-

# POSIX ACL - příklad

- ACL zjistíme příkazem **`getfacl`**
- ACL nastavíme příkazem **`setfacl`**
- Příklad:
  - **`getfacl soubor`**

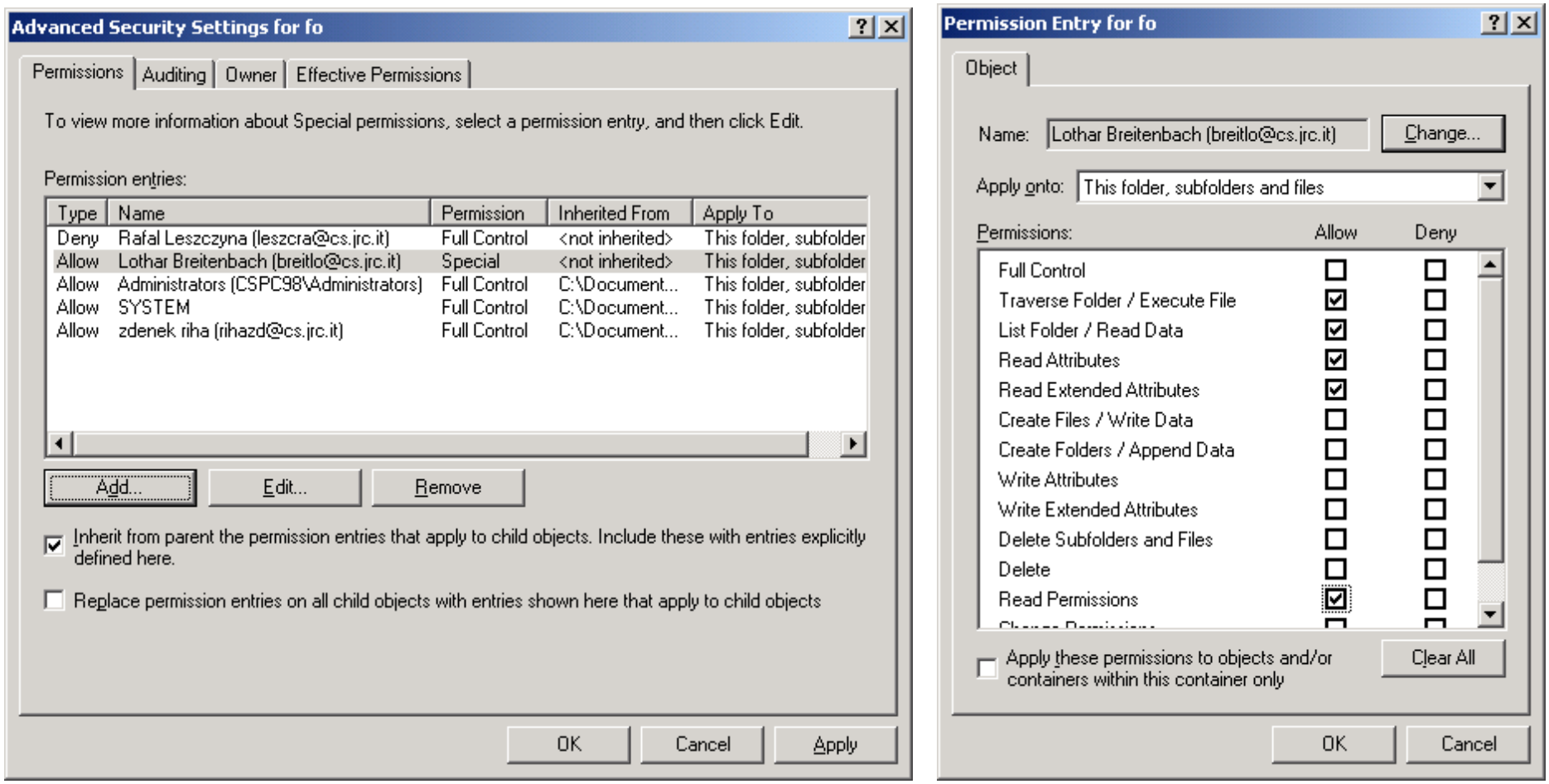
```
user::rw-                                   (vlastník)
user:lisa:rw-        #effective:r--         (uživatelka lisa)
group::r--                                  (skupina)
group:soft:rw-       #effective:r--         (skupina soft)
mask::r--                                   (maska)
other::r--                                  (ostatní)
```

  - Zkráceně lze psát tato práva jako:
    - g:soft:rw,u:lisa:rw,u::rw,g::r,o::r,m::r
  - Zde maska omezuje w právo uživatelky lisa a skupiny soft

# Implementace ACL – Windows

- Nejen číst data/zobrazovat obsah složky, zapisovat data/vytvářet soubory, spouštet soubory/procházet složku, ale také
  - Číst atributy,
  - Číst rozšířené atributy,
  - Připojovat data/vytvářet složky
  - Zapisovat atributy,
  - Zapisovat rozšířené atributy,
  - Přebírat vlastnictví,
  - Číst oprávnění,
  - Měnit oprávanění,
  - ostraňovat
- a to pro uživatele nebo skupiny uživatelů.
- Atributy nejen Povolit/Odepřít, ale také možnost auditování úspěšného či neúspěšného pokusu o přístup.
- Větší možnosti nastavování přístupových práv znamenají možnost přesněji postihnout/implementovat bezpečnostní politiku.
- V praxi však často uživatel přijde, naloguje se jako administrátor (aby mohl instalovat aplikace apod.) a jako administrátor pracuje do ukončení své práce.

# Windows ACL – příklad



# Řízení přístupu na čipových kartách

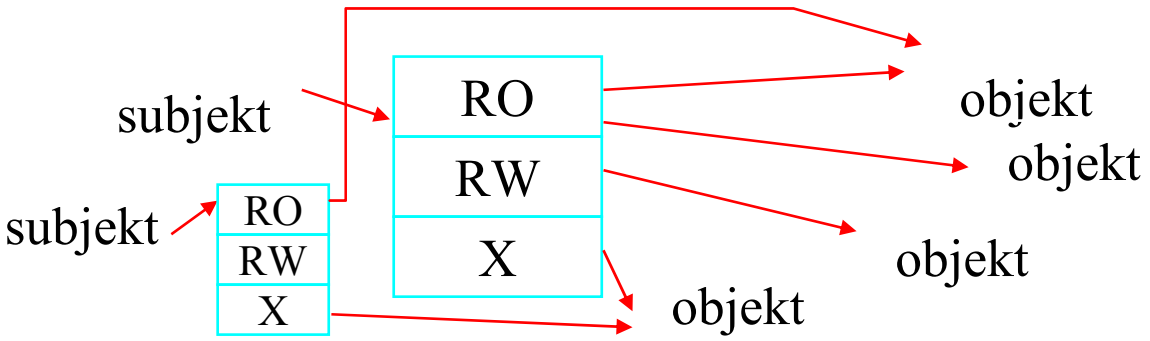
- Řízení přístupu k datům na kartě je tvořeno především řízením přístupu k souborům.
- S každým souborem je svázána hlavička souboru, která určuje přístupová práva k souboru.
- Základním principem řízení přístupu je zadávání PINů a jejich management.
- Přístup k souboru může být například vázán na splnění některé z těchto podmínek:
  - ALW (vždy povolen přístup)
  - CHV1 (nutné zadat PIN uživatele 1)
  - CHV2 (nutné zadat PIN uživatele 2)
  - NEV (přístup nepovolen)

# Čipové karty – PIN management

- PINy jsou ukládány v samostatných souborech (EF). Přístupová práva k těmto souborům určují možnost změny těchto PINů.
- Při změně PINu je požadavek provázen starým a novým PINem.
- Počet neúspěšných pokusů bývá omezen. Po překročení limitu (3 – 5) je PIN blokován.
- Pro odblokování je třeba zadat PIN a odblokovací PIN (u SIM karet nazýván PUK).
- I počet neúspěšných odblokování je omezen.

# Seznam přístupových oprávnění

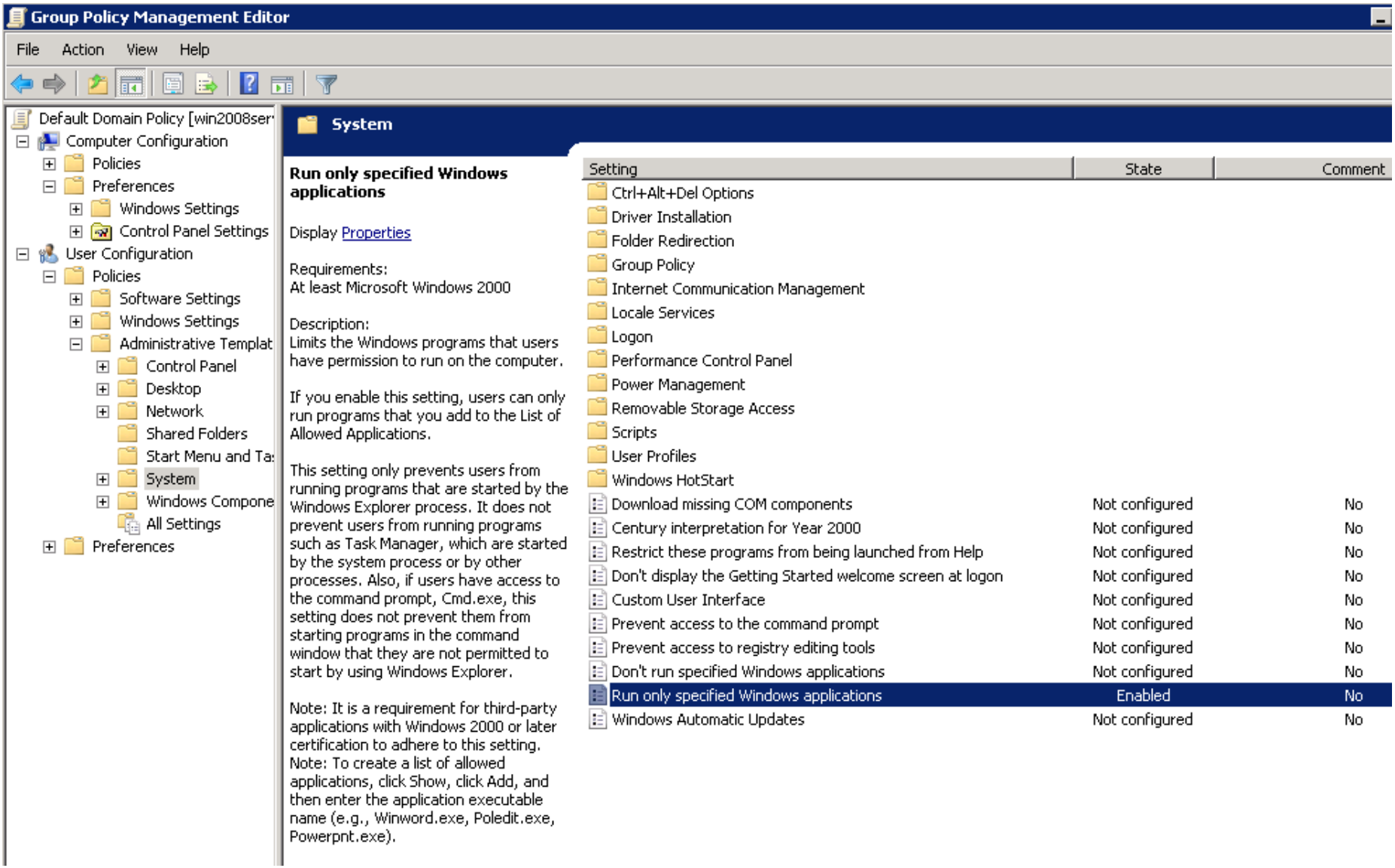
- Seznam přístupových oprávnění (capabilities)
- Ukládáme matici přístupových práv po **subjektech**
- Není žádnou novinkou
- Například: model Multics, IBM AS/400
- Dnes často ve formě certifikátů
- Výhoda seznamu přístupových oprávnění
  - Efektivní kontrola přístupových práv při přístupu k objektu
- Nevýhoda seznamu přístupových práv
  - Nesnadné zjistit kdo má k určitému objektu přístup



# Skupinové politiky

- Windows (2000 a výše; omezeně i dříve) implementují nejen ACL, ale i přístupová práva ve formě přístupových oprávnění.
- Tato bezpečnostní oprávnění mohou převážit nebo doplnit přístupová práva ve formě ACL.
- Bezpečnostní politika je svázána se **skupinami** uživatelů [skupiny (groups) jsou definovány v aktivním adresáři (active directory)].
- „Skupinové politiky“ (“Group policy”) [dříve „system policy“] se vztahují na domény, počítače, celé organizace.
- Příklad: Skupinová politika obsahuje seznam aplikací, které skupina uživatelů nemá oprávnění spouštět (např. Internet Explorer, Outlook Express, …). Přístupová práva programu na disku (Internet Explorer) jsou nastavena na [everybody: rx]. Skupinová politika převáží přístupové práva souboru (aplikace) → spuštění aplikace je zakázáno.

# Skupinové politiky



# Spouštění nedůvěryhodného kódu

- Jak se chovat k nedůvěryhodnému kódu, získanému např. z internetu?
  - Autentizace kódu, viz např. Authenticode
  - Speciální jazyk neumožňující škodlivou činnost (JavaScript)
  - Kontrolní programy, které před spuštěním kontrolují kód na škodlivou činnost
    - Obecně nerozhodnutelné
    - V praxi heuristiky antivirových programů
  - Spuštění kódu s minimálními uživatelskými právy (např. unixový nobody)
  - Omezené prostředí, ve kterém nemá kód přístup k určitým prostředkům („Sandbox“)
    - Např. interpret Javy – Java Virtual Machine – applety stažené z webu (bez přístupu na disk, komunikace možná jen s původním serverem)

# Podpora řízení přístupu v HW

- Přístup do paměti, ke strukturám OS:
  - Úkolem je zamezit komunikaci/ovlivňování procesů jinak než explicitně povoleným způsobem
  - Např. „fence address“ – limit paměti, do nižších adres má přístup jen operační systém
  - Např. „segmentové adresování“ – Adresování formou segment+offset. Segment může měnit jen operační systém (referenční monitor)
  - Např. dva režimy procesoru – autorizovaný a neautorizovaný. V neautorizovaném režimu není možné měnit segmentové registry.
  - Např. „Rings of protection“ – několik režimů činnosti s různými právy. Měnit ring možné jen v režimu ringu 0 (operační systém). Volání rutin operačního systému – změna ringu: GATE
  - Např. vojenské systémy chrání nejen data procesů, ale i metadata (jaké procesy s jakými parametry spuštěné kterými uživateli běží v systému)

# Objektové programování

- Přístup k datům může být omezen pouze na explicitně uvedené metody
- Příklad v C++

```
class example {
private:
int counter;
protected:
void add_substract(int);
public:
void decrease(void);
void increase(void);
};
```

# Řízení přístupu

- Jedna z nejdůležitějších komponent bezpečnosti jakéhokoliv systému
- Bohužel ne vždy je kód řízení přístupu (referenční monitor) bezchybný. (typicky nedostatečná kontrola nedůvěryhodného vstupu a následný „buffer overflow“)
- Příliš mnoho kódu operačního systému je označeno za důvěryhodný (jádro obsahuje ovladače nejrůznějších zařízení napsaných programátory řady firem/organizací)
- „Race condition“ – operace ověření přístupových práv a použití práv není atomická (zneužitelné u programů s většími právy – SUID/SGID programy, webové CGI aplikace)
- Trojští koně

# Řízení přístupu

- Separace oprávnění – pro vykonání určité akce je potřebný souhlas několika osob (např. velkou bankovní transakci musí podepsat dva bankovní úředníci)
  - Řízení přístupu na úrovni OS (často ani middleware) toto nepodporuje
  - Nutná podpora přímo v aplikačním SW

- Princip nejmenších privilegií – uživatel má právo přístupu jen k takovým objektům, ke kterým z titulu svého pracovního zařazení přístup potřebuje. Počátečně je množina oprávnění malá a postupně se rozrůstá. Žádný uživatel nemá přístup k objektům, ke kterým přístup nepotřebuje.

# Politiky řízení přístupu

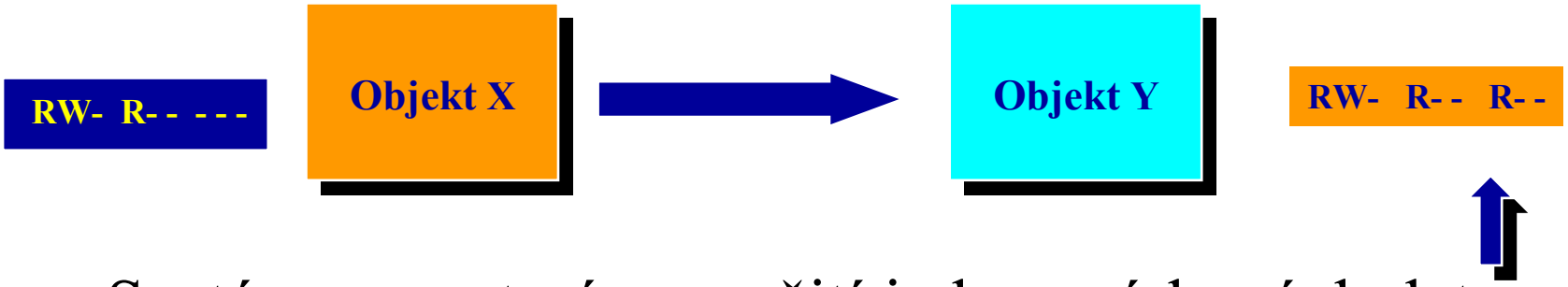
- **volitelný přístup** (*discretionary*)
  - subjekt (vlastník objektu) rozhoduje o tom, kdo má k objektu přístup
  - volitelná = určuje subjekt–vlastník objektu
  - typicky politika podporovaná operačním systémem
    - ✓ podporuje i operace změny vlastníka objektu
- **povinný přístup** (*mandatory*)
  - systémová politika nezávislá na vůli subjektů rozhoduje o tom, kdo má k objektu přístup

# Volitelné řízení přístupu – výhody

- Jednoduchost = malá režie
- Velká vyjadřovací schopnost
- Lze relativně jednoduše vázat udělení přístupových práv na splnění dodatečných časových, místních aj. podmínek
- Flexibilita

# Volitelné řízení přístupu – nevýhody

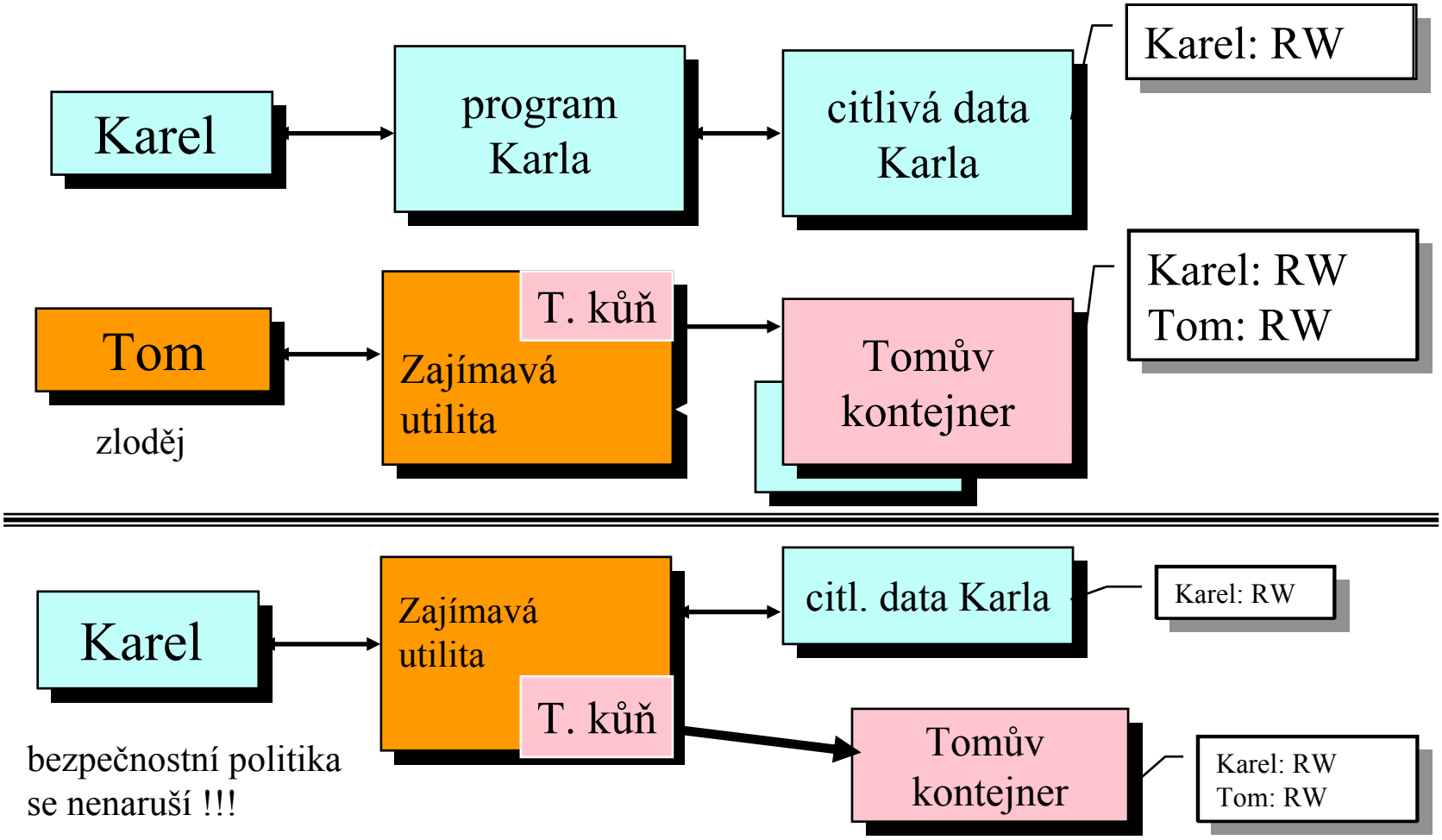
- Nedostatečná bezpečnost
  - ➢ použití pouze přístupových práv není dostatečné v situacích, kdy klademe důraz na bezpečnost
- Není odolné vůči "Trojským koním"




- Systém se nestará o využití jednou získaných dat
  - ➢ např. dám skupině právo čtení na důvěrný soubor, nějaký člen si ho zkopíruje a nesprávně nastaví přístupová práva

# Volitelné řízení přístupu

- Příklad útoku Trojským koněm na správu souborů s ACL



# Víceúrovňové systémy

- MLS (Multilevel systems)
- Koncept podporovaný mnohaletým výzkumem sponzorovaným vládami (zvláště USA a UK)
- Původně podpora důvěrnosti, později i integrita (komerční systémy)
- Primární model bezp. politiky – Bell-LaPadula
  - 1973, pro US AirForce
  - ochranné schéma klasifikací a oprávnění po 2. světové válce

# Povinné řízení přístupu

- systémová politika nezávislá na vůli subjektů rozhoduje o tom, kdo má k objektu přístup
- zavedeme
  - kategorie subjektů (proces, uživatel) = důvěryhodnost
  - klasifikace objektů (data) = důvěrnost
- definujeme uspořádání klasifikací objektů
- definujeme množinu kategorií subjektů
- **referenční monitor** (monitor odkazů)
  - implementace funkce prosazující bezpečnost *řízení přístupu*
  - při každém přístupu subjektu k objektu kontroluje, zda tento přístup odpovídá zásadám bezpečnostní politiky
  - bezpečnostní politika: pravidla toku dat mezi objekty a subjekty

# Politiky nejsou exkluzivní!

- Při povinném přístupu (prosazovaném systémem) lze (a často je to žádoucí) obvykle podporovat i volitelný přístup!!!
  - Bezpečnost (daná politikou povinného přístupu) s určitou flexibilitou (podporující bezpečnost) vyjadřovacími prostředky volitelného přístupu

# Model Bell-LaPadula (1)

- paradigma objekt – subjekt
- uživatel
  - Má počáteční bezpečnostní úroveň uživatele, resp. bezpečnostní **oprávnění** (*clearance*)
  - Přihlašuje se na aktuální bezpečnostní úrovni uživatele, s právy přístupu k objektům nepřevyšujícími práva daná bezpečnostním oprávněním
- subjekt
  - aktivní element – proces činný na pokyn uživatele
  - provádí akce:
    - ✓ read-only, append (zápis bez čtení), read-write, execute
  - bezpečnostní úroveň procesu = bezp. úroveň jeho uživatele
    - ✓ Je daná „důvěryhodností“ subjektů vlastnících proces a „důvěrností“ (citlivostí) zpracovávatelných objektů (klasifikací)

# Model Bell-LaPadula (2)

- objekt
  - pasivní, chráněný element
  - obsahuje informace
  - soubor dat, prostor paměti, program
  - **klasifikace** objektu = bezpečnostní úroveň objektu
    - ✓ daná důvěrností (citlivostí) informace obsažené v objektu
    - ✓ definuje/mění ji vlastník objektu, vlastnictví objektu je nepřenositelné

# Bell-LaPadula – Klasifikace / kategorie

- Bezpečnostní úroveň L= (C, $\underline{S}$)
  - C – klasifikace objektů

| | | |
|---|---|---|
| TS | top secret | přísně tajné |
| S | secret | tajné |
| C | classified | pouze pro vnitřní potřebu (důvěrné.) |
| U | unclassified | neklasifikováno |

    Definice uspořádání: TS > S > C > U
  - $\underline{S}$ – podmnožina množiny kategorií subjektů
    - ✓ množina kategorií subjektů je dána aplikací
      - {odbor obrany, ekonomický odbor, vnitřní odbor}
      - {ekonomický odbor, vnitřní odbor}
      - {odbor obrany, vnitřní odbor}
      - {vnitřní odbor}
- Upořádání bezpečnostních úrovní – dominance

  L1=(C1, $\underline{S1}$), L2=(C2, $\underline{S2}$), $L1 \geq L2 \Leftrightarrow C1 \geq C2 \wedge \underline{S1} \supseteq \underline{S2}$

# Bell-LaPadula – příklad

- bezpečnostní úrovně

  L1 = (S, {ekonom.}) tajné, {ekonom. odbor}

  L2 = (C, {ekonom.}) pro vnitřní potřebu, {ekonom. odbor}

  L3 = (TS, {obrana}) přísně tajné, {odbor obrany}

  L4 = (TS, {ekonom., obrana}) přísně tajné, {odbor obrany, ekonomický odbor}

- uspořádání (dominance) bezpečnostních úrovní

  $L1 \geq L2$ S > C, {ekonom.} ≡ {ekonom.}, L1 dominuje L2

  L1, L3 L1 neporovnatelné s L3 : {ekonom.} {obrana}

  $L1 \leq L4$ S < TS, {ekonom.} $\subseteq$ {ekonom, obrana}

  L2, L3 L2 neporovnatelné s L3 : {ekonom.} {obrana}

  $L2 \leq L4$ C < TS, {ekonom.} $\subseteq$ {ekonom., obrana}

  $L3 \leq L4$ TS = TS, {vnější vztahy} $\subseteq$ {ekonom., obrana}

# Klasifikace v ČR a NATO

| NATO | ČR | Německo |
|---|---|---|
| cosmic top secret | přísně tajné | streng geheim |
| NATO secret | tajné | geheim |
| NATO confidential | důvěrné | VS-vertraulich |
| NATO restricted | vyhrazené | VS-nur-für den Dienstbegrauch |

# Bell-LaPadula – stav systému

- Stav systému, $\Sigma = (b, \underline{M}, f)$

  b – množina aktivních (právě realizovaných) přístupů
  - ✓ trojice (subjekt, objekt, právo)

  $\underline{M}$ – matice přístupových práv
  - ✓ M[s, o] přístupová práva subjektů s k objektům o

  f – úrovňová funkce: $\underline{O} \cup \underline{S} \rightarrow L$,
  - ✓ množiny: O – objektů, S – subjektů, L – bezpečnostních úrovní
  - ✓ udává bezpečnostní úroveň každého subjektu a objektu
    - o objekty, každý má jedinou klasifikaci (bezp. úr. obj.): $f_o$
    - o subjekty, každý vlastní dvě „bezpečnostní úrovně subjektu“:
      - bezpečnostní oprávnění, clearance $f_p$
      - aktuální bezpečnostní úroveň subjektu $f_a$, $f_a(s) \leq f_p(s)$
- bezpečnost systému je chápána jako vlastnost <u>stavů systému</u>

# Bell-LaPadula – bezpečnost stavu

- Stav systému se mění operacemi změny stavu systému
  - uplatnění přístupových práv, změny přístupových práv
- Stav systému je bezpečný pouze tehdy, když jsou splněny všechny bezpečnostní vlastnosti
  - omezení daná vztahy bezpečnostních úrovní subjektů a objektů
- Operace změny stavu systému se povolí pouze tehdy, když výsledný stav systému po jejím provedení bude bezpečný – kontroluje referenční monitor
- Důvěryhodnost subjektu
  - důvěryhodný subjekt – smí porušovat bezpečnostní politiku povinnou pro nedůvěryhodné subjekty
    - Ví, co smí a nesmí, kdy komunikuje s jinými důvěryhodnými subjekty, ...
  - nedůvěryhodný subjekt – jeho chování je třeba hlídat doplňkovými omezeními podle zavedené bezpečnostní politiky

# Bell-LaPadula – operace změny stavu

- právo přístupu – read-only, append, execute, read-write
- operace změny stavu
  - získat/vrátit právo přístupu, zahájit/ukončit operaci s objektem
    - mění množinu aktivních přístupů **b**, doplňuje / ruší trojici (s, o, p)
  - dát / odebrat právo přístupu subjektu k objektu, modifikace **M**
    - musí být v souladu s politikou definovanou axiomy povinné bezpečnostní politiky
  - změnit aktuální bezpečnostní úroveň subjektu
    - musí se zachovat dominance bezpečnostního oprávnění subjektu, mění se **f**
  - změnit bezpečnostní úroveň objektu, jeho klasifikaci
    - pouze pro „neaktivní“ (se kterým nikdo nepracuje) objekt, mění se **f**, lze ji pouze
      - použít oprávněně – nová bezpečnostní úroveň *objektu* musí dominována bezpečnostní úrovní *subjektu* provádějícího změnu
      - a zesilovat – nová úroveň objektu musí dominovat předchozí úrovni

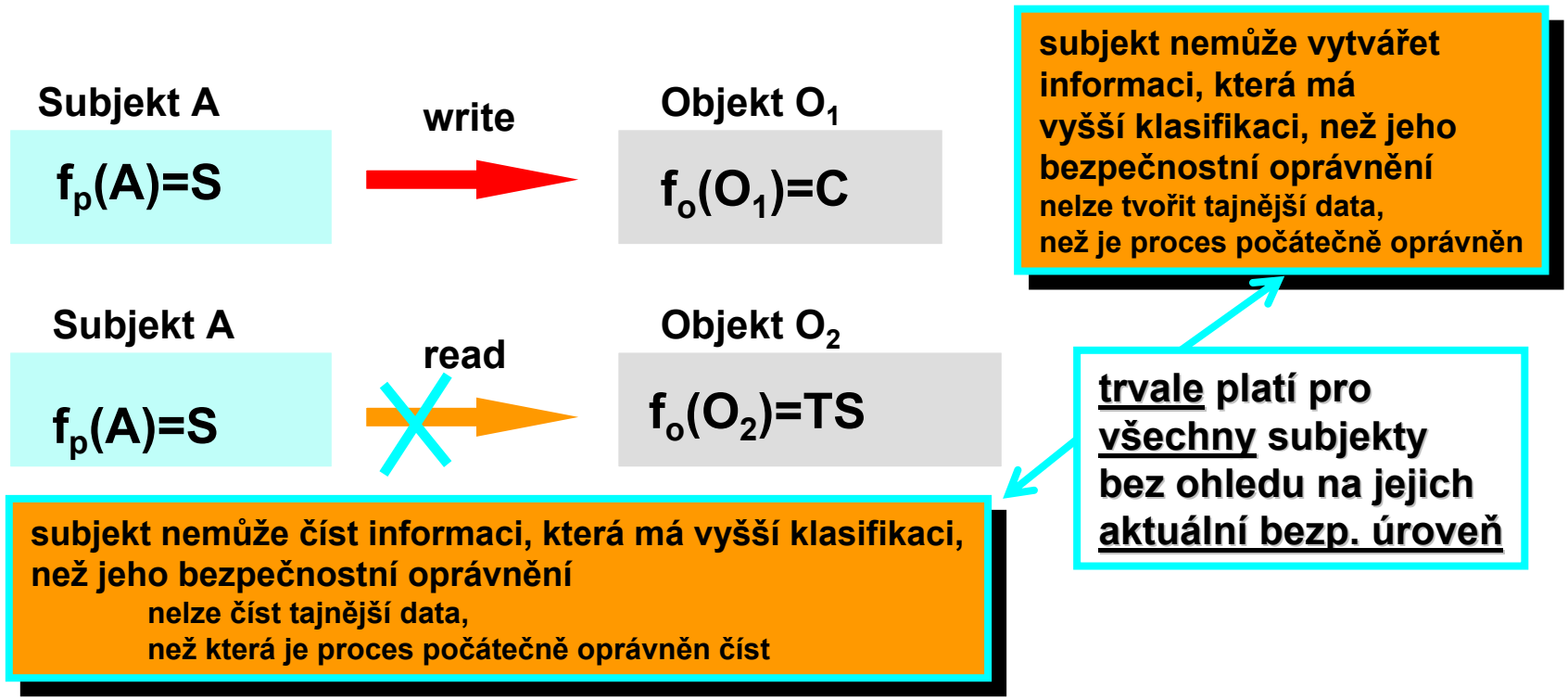
# Vlastnosti (axiomy) modelu Bell-LaPadula

- Procesy nesmějí číst data na vyšší úrovni (tzv. základní bezpečnostní vlastnost – *ss property*, též *NRU - no read up*).

- Procesy nesmějí zapisovat data do nižší úrovně (tzv. *-vlastnost, též *NWD - no write down*).

# ss-vlastnost

- subjekt může přistupovat (read/write) pouze k objektům s bezpečnostní úrovní (klasifikací) dominované jeho bezpečnostním oprávněním (clearance)
- Pak $\forall s \in S\ o \in O\ read \in M[s,o] \vee write \in M[s,o] \Rightarrow f_p(s) \geq f_o(o)$



# Nedostatečnost ss-vlastnosti

- subjekt A nižší bezpečnostní úrovně $l_0$ ,
  než je klasifikace objektu $o_1$, může vytvořit Trojského koně, který bude spuštěn s vyšší bezpečnostní úrovní $l_v > o_1$
- Trojský kůň může přečíst obsah objektu s klasifikací $o_1$
- Trojský kůň může vytvořit objekt s okopírovanou informací s klasifikací objektu $o_2 < o_1$
- subjekt A může číst objekt s klasifikací $o_2$

# *–vlastnost

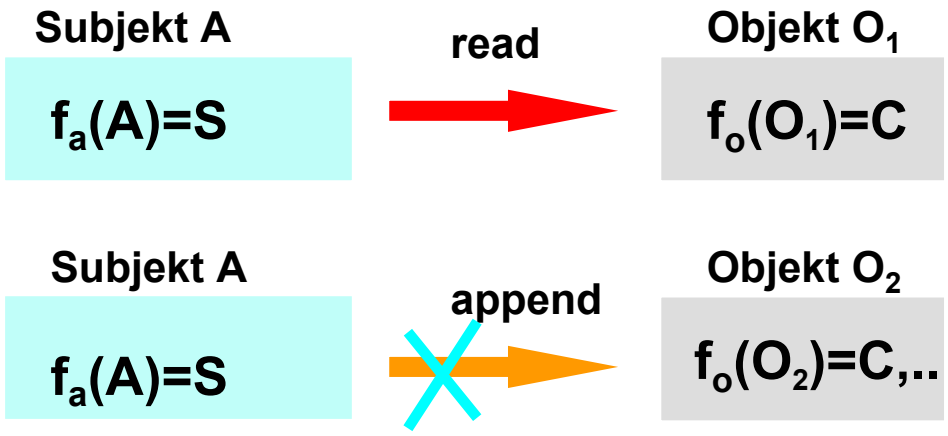
- pouze pro *nedůvěryhodné subjekty*
- Pak $\forall s \in S' \; \forall o \in O$
  - ✓ read $\in M(s,o) \Rightarrow f_a(s) \geq f_o(o)$
  - ✓ write $\in M(s,o) \Rightarrow f_a(s) = f_o(o)$
  - ✓ append $\in M(s,o) \Rightarrow f_a(s) \leq f_o(o)$
- splnění tohoto axiomu implikuje splnění předchozího axiomu; opak ale neplatí

**nedůvěryhodný subjekt může číst informaci, jestliže její klasifikace je dominovaná aktuální b. ú. subjektu**
- číst lze jen méně tajná data

**nedůvěryhodný subjekt může zapisovat informaci, jestliže její klasifikace je shodná s aktuální b. ú. subjektu**
- tvořit lze jen stejně tajná data

**nedůvěryhodný subjekt může doplňovat informaci, jestliže její klasifikace dominuje aktuální b. ú. subjektu**
- doplňovat lze stejně tajná data
- nebo tajnější data

**nelze poslat zprávu procesu s nižším bezpečnostním oprávněním**



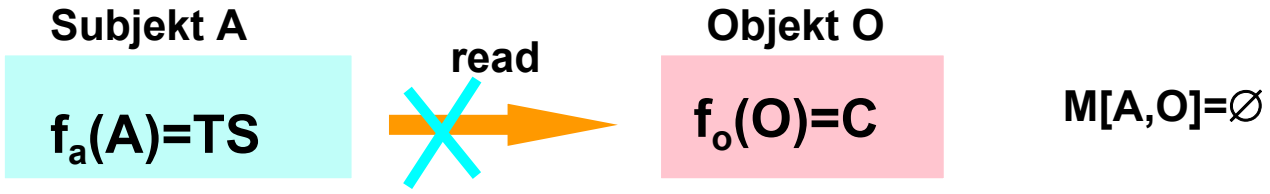

**důvěryhodné subjekty mohou porušovat *-vlastnost**

# Volitelný přístup v Bell-LaPadula

- Také volitelný přístup (discretionary access property)
  - ➢ pro povolení přístupu je nutné mít patřičná práva v matici přístupových práv **M**, subjekt musí být pro danou operaci autorizován

$$\forall s \in S\ \forall o \in O\ \forall a \in A\ \ <s,o,a> \in b \Rightarrow a \in M[s,o]$$

    - ✓ model B-P je rozšířením modelu s maticí přístupových práv
  - ➢ Příklad – subjekt s bezp. úrovní TS **nemusí** mít právo čtení k jistému objektu O, byť tento má klasifikací C




- Stav systému – je považován za bezpečný, jsou-li splněny všechny 3 vlastnosti

# Problémy víceúrovňových systémů

- Jak klasifikovat data?
- Tendence k příliš striktní klasifikaci!
- Jak propojit MLS jednoho typu MLS jiného typu (neekvivalentní klasifikací) – např. US vs. UK?
- Vývoj MLS bývá příliš komplikovaný a drahý
- Administrace je náročná
- Uživatel bývá při své práci příliš omezován
- Jak řešit snížení klasifikace?

# Skrytý kanál

- *Covert channel* - mechanismus, který není primárně určen pro komunikaci, ale může být využit (zneužit) pro komunikaci mezi jednotlivými úrovněmi
- Typické je využití nějakého sdíleného prostředku
  - zaplnění disku
  - pozice hlavičky na disku
  - zamykání souborů
  - čas posledního přístupu k souboru
  - aktuální zátěž procesoru
- Obrana – snížení šířky pásma komunikačního kanálu
  - diskové kvóty, nucené nastavení hlaviček disku
  - zavedení šumu

# Polyinstance

- Chráníme existenci informace na vyšší úrovni
- Uživatel na nižší úrovni chce vytvořit soubor, který již existuje na úrovni vyšší
  - Můžeme zakázat ⇒ prozradíme existenci souboru
- *Noninterference* = vlastnost, kdy akce uživatele na vyšší úrovni nijak neovlivní to, co vidí uživatel na nižší úrovni
- Souborový systém: zavedeme konvence pro pojmenování
- Databáze: netriviální problém (smyšlený příběh vs. zatajení)
- Př.: USA: UK:

| klasifikace | Účel skladu |
|---|---|
| C | Sklad atomových zbraní |
| U | Sklad uniforem |

| klasifikace | Účel skladu |
|---|---|
| C | Sklad atomových zbraní |
| U | klasifikováno |

# Model Biba

- K zajištění integrity
  - „převrácený“ model Bell-LaPadula
  - Integrita a důvěrnost jsou svým způsobem doplňující se koncepty (někdo musí zapsat = změnit integritu, aby šlo vůbec číst)
- Číst lze jen data vyšší úrovně (důležitější, přesnější, spolehlivější)
- Zapisovat lze jen „dolů“ (podřízeným)
- Např. systém pro informování cestujících bere data od signalizačního systému, ale nemůže jeho data měnit.

# Dopad MLS

- Velké množství bezp. projektů a výzkumu
- Koncepty pro ne-MLS systémy, jako např.
  - Důvěryhodná cesta (*Trusted Path*) – bezpečný kanál pro komunikaci komponent
  - Důvěryhodná distribuce (*Trusted Distribution*) – bezpečná distribuce systému
  - Důvěryhodná správa zařízení (*Trusted Facility Management*) – bezpečná administrace

# Skutečné MLS systémy

- Upravené verze běžných systémů
  - Trusted Solaris
  - HP Virtual Vault
  - TrustedBSD
  - SE Linux
  - AppArmor

# SE Linux

- vyvinuto za pomoci NSA
- součásti jádra Linuxu od verze 2.6.0
- je nadstavbou POSIX capabilities (tj. práva je možné nastavovat jemněji než jen běžný uživatel vs. root)
- od verze 2.6.12 obsahuje i MLS
- aktivní (enforcing) vs. pasivní (permissive) režim

# SE Linux (2)

- Při přístupu subjektu k objektu pomocí systémového volání se kromě běžných přístupových práv kontroluje splnění bezpečnostní politiky
- Tuto kontrolu provádí *bezpečnostní server* vůči sadě pravidel
- Rozhoduje se na základě trojice identita:role:typ a klasifikace v MLS (volitelné)
  - identita (user_u, system_u nebo speciální identita pro některé uživatele), odlišná od UID
  - role (sysadm_r, system_r, user_r)
  - typ – typ objektu (file_t, default_t, user_home_dir_t)

# SE Linux (3) – příklad

```
# ls -Z /
drwxr-xr-x  root     root     system_u:object_r:bin_t          bin
drwxr-xr-x  root     root     system_u:object_r:boot_t         boot
drwxr-x---  root     root     root:object_r:user_home_dir_t    root
drwxr-xr-x  root     root     system_u:object_r:sbin_t         sbin
drwxr-xr-x  root     root     system_u:object_r:file_t         selinux
-rw-r--r--  root     root     system_u:object_r:net_conf_t     yp.conf

# ls --scontext
system_u:object_r:etc_t          shadow

# chcon system_u:object_r:httpd_sys_content_t index.html

# id -Z
root:staff_r:staff_t
```

# Integritní úrovně Windows Vista

- Každý objekt má přiřazenou integritní úroveň znamenající jeho důvěryhodnost
- 6 hierarchických integritních úrovní
  - Untrusted – procesy přihlášené anonymně
  - Low – procesy pracující s Internetem
  - Medium – běžná úroveň
  - High – administrátor
  - System – systémové procesy, služby jádra
  - Installer – instalace a odinstalace

# Integritní úrovně Windows Vista

- Subjekty na nižší úrovni nemohou modifikovat objekty na vyšší úrovni
  - Na čtení a spouštění se to nezvtahuje (vs. Biba)
- Integritní úroveň není dynamická
  - Čtením méně důvěryhodných objektů se úroveň procesu nesnižuje (vs. Biba)
- Integritní politika zabraňuje přístupu k objektům, ale nesleduje informační toky

# Role based access control (RBAC)

- Není ani volitelné ani povinné řízení přístupu, ale samostatná kategorie
- Uživatelům jsou přiřazeny role (uživatel může mít více rolí)
- Role znamenají práva k provedení určitých akcí, tato práva mohou být specifikována velice jemně (např. přidat záznam, upravit nějakou položku apod.), jemněji než pomocí ACL
- Role jsou však specifické pro každý IS, v heterogenní organizaci není snadné vytvořit jasný systém rolí, jím odpovídajících práv a rozdělení uživatelů do rolí

# Role based access control (RBAC)

- Příklad RBAC – oracle
  - create role vyuka;
  - grant CREATE SESSION, ALTER SESSION, CREATE PROCEDURE, CREATE SEQUENCE, CREATE SYNONYM, CREATE TABLE ... to vyuka;
  - grant vyuka to zriha;

Příští přednáška 24. 4. 2012 v 10:00

zriha@fi.muni.cz
matyas@fi.muni.cz