Literatura

- ISO17799
- ITILv3
- ISO27000
- PFLEEGER, "*Security in Computing*", Prentice-Hall, 1989
- JACKSON, HRUSKA, "*Computer Security Reference Book*", Butterworth-Heineman, 1992
- RUSSELL, GANGEMI, "*Computer Security Basics*", O'Reilly&Associates, 1991
- SCHNEIER, "*Applied Cryptography*", John Wiley & Sons, 1994
- PŘIBYL, "*Ochrana dat v informatice*", scriptum ČVUT, 1993
- Frequently Asked Questions About Today's Cryptography, http://www.rsasecurity.com/rsalabs/faq/index.html

Zcela základní pojmy

Informačním systémem(IS) rozumíme soubor technických prostředků, softwaru a jeho konfigurací, záznamových medií, postupů, dat a personálu, který daná organizace používá ke správě svých informací.

Korektní stav IS odpovídá situaci, kdy systém je schopen v definovaném rozsahu poskytovat zajišťovat všechny požadované vlastnosti zpracovávaných informaci, či poskytovaných služeb, například:

- utajení
- dostupnost
- integrita
- nepopiratelnost
- včasnost
- současnost
- autenticita
- anonymita
- pseudonymita
- …

Uvedený výčet není v žádném případě vyčerpávající, nebo reprezentativní. Výběr služeb vždy individuální

Bezpečnostní incident je stav, kdy došlo k (potenciálnímu) porušení alespoň jedné z požadovaných vlastností.

Pravidla hry

Vlastník IS buduje spoustu mechanismů – tzv. *bezpečnostních protiopatření* pro zabránění vzniku incidentu

Základní princip ochrany výpočetních systémů.
O peníze jde až v první řadě.

→ Chráněné objekty mají svoji cenu, pro kterou jsou chráněny. Cena může být různá pro majitele a útočníka.
→ Ochrana není, ani přibližně, zadarmo.

Princip nejsnazšího průniku.
Je třeba očekávat, že útočník použije libovolný způsob průniku.

Fanatismus nepřináší dobré výsledky

Bezpečnost je souboj mezi zdroji (čtěte penězi, znalostmi, důvtipem, ..) útočníka a zdroji provozovatele systému. Kdo jich má víc, pravděpodobně zvítězí.

O co se hraje

Informační systém je tvořen souborem tzv. *aktiv*. Jejich společným cílem je poskytovat vám služby v požadované kvalitě. Mezi aktiva patří mimo jiné:

- záznamová media
- počítače
- tiskárny
- programy
- konfigurace
- vlastní informace
- sklad spisů
- napájení
- komunikační linky
- administrátoři
- uživatelé
- zálohy
- provozní prostory
- ...

Svůj soupis aktiv si každý musí provést sám.

Váš útočník hledá *expozici* tj. místo potenciálního poškození.

Zranitelností rozumíme nedostatek bezpečnostního systému, může být použit k poškození nebo zcizení informací.

Př: Data o novém výrobku jsou z pohledu útočníka expozicí, když si naplánuji, že je budu svým pobočkám posílat nešifrované majlem, je to zjevná zranitelnost.

Bezpečák by měl vidět samé *hrozby* tj. skutečnosti, které potenciálně mohou být původci bezpečnostního incidentu. Zdaleka nejstrašnější hrozbou jsou vlastní uživatelé. Kromě nich sem patří ještě:

- povodně a záplavy
- požáry
- zloději
- rozvědky
- konkurence
- hackeři
- vandalové
- nešikové s bagrem
- viry a červi
- závady techniky
- výpadky napájení
- teplota
- vlhkost
- vibrace
- ...

Přehled relevantních hrozeb si musí každý sestavit sám. Někdy se tomu učeně říká *model ohrožení*.

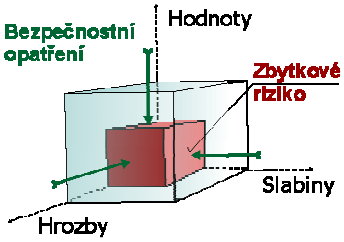
Cíl hry

Cílem překvapivě není zbavit se útočníka - prostě proto, že se to nevyplatí.

Naplněním hrozby vznikne bezpečnostní incident jehož finančnímu vyjádření se říká *dopad.*

Rozsah hrozeb spolu s pravděpodobností jejich realizace udává celkovou míru *rizika*.

Riziko vztažené k určitému období = *očekávaná ztráta*.



Cílem najít místo, kde se bezpečnostní opatření přestávají vyplácet.

Nevyloučili jsme zcela riziko incidentu – zbylo *zbytkové riziko*

Stav, kdy vám někdo nebo něco prostřelilo bezpečnostní opatření, je nutno brát jako další z provozních režimů IS.

Jak na to

Požadavky na bezpečnost

Je řada důvodů, proč vytvářet bezpečnostní opatření

- zákonné požadavky
- obecné standardy
- resortní normy
- ochrana obchodního tajemství
- dosažení provozní kontinuity
- požadavky protistrany
- zajištění konkurenčních výhod
- ...

Okruh možných řešení

Pomoci může celá řada technických norem a certifikátů

Plán

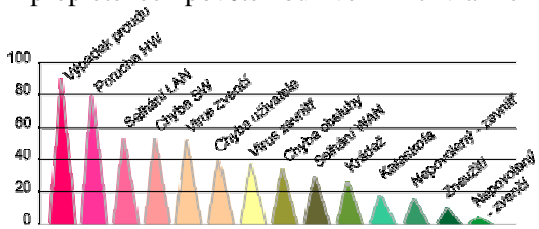
Potřebujete *bezpečnostní politiku*. – zde se naplánuje, jak budete řešit všechny oblasti bezpečnosti, kdo je za co zodpovědný a jak to budete implementovat a provozovat.

Realizace a provoz

Praktické realizace, následně provoz, monitorování, aplikace změn, verifikace, auditu atd. atp.

Krok stranou

Bezpečnost je naprosto netriviální propletenec povětšinou velmi triviálních záležitostí. Obrázek je namalován před zhruba čtyřmi lety podle průzkumu který činil Národní Bezpečnostní Úřad ve spolupráci s časopisem DSM a společnosti PriceWaterhouseCoopers.



Možné hrozby

1. *přerušení* - některá část systému je ztracena nebo nedosažitelná
2. *zachycení* - neautorizovaný subjekt získá přístup k nějakému objektu systému
3. *modifikace* - neautorizovaný subjekt získá možnost pozměňovat některé části systému
4. *fabrikace* - neautorizované vytvoření nového objektu
5. ...
6.

Zdroje ohrožení

1. vyšší moc (požár, povodeň, zemětřesení, blesk, …)
2. závady technického zařízení
3. neúmyslné lidské chyby
4. záměrné útoky

Klasifikace možných útočníků

klasifikovat lze dle mnoha kriterií, zejména dle

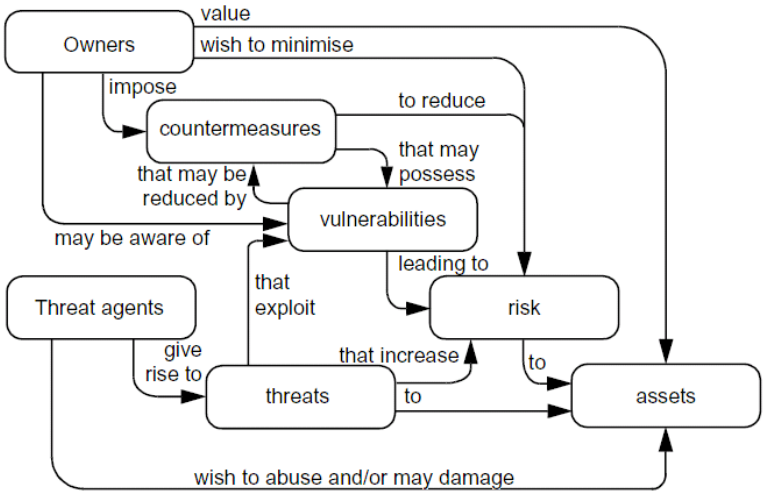
I. způsobu, jak se projeví způsobená škoda
   A. ztráta integrity
   B. ztráta dosažitelnosti
   C. ztráta autenticity …

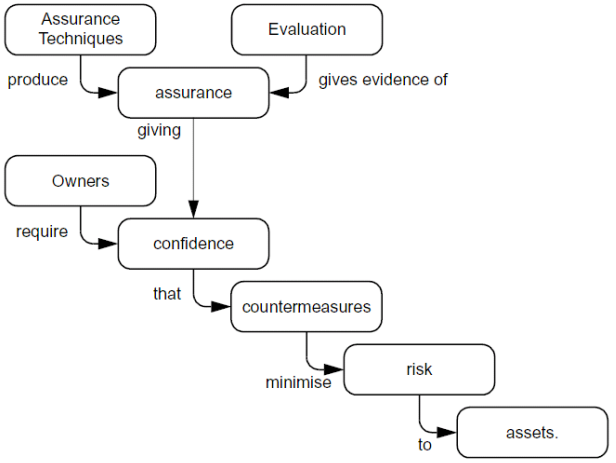
II. druhu způsobené ztráty
   A. neautorizované použití služeb
   B. přímá finanční ztráta
   C. fyzické poškození, vandalismus

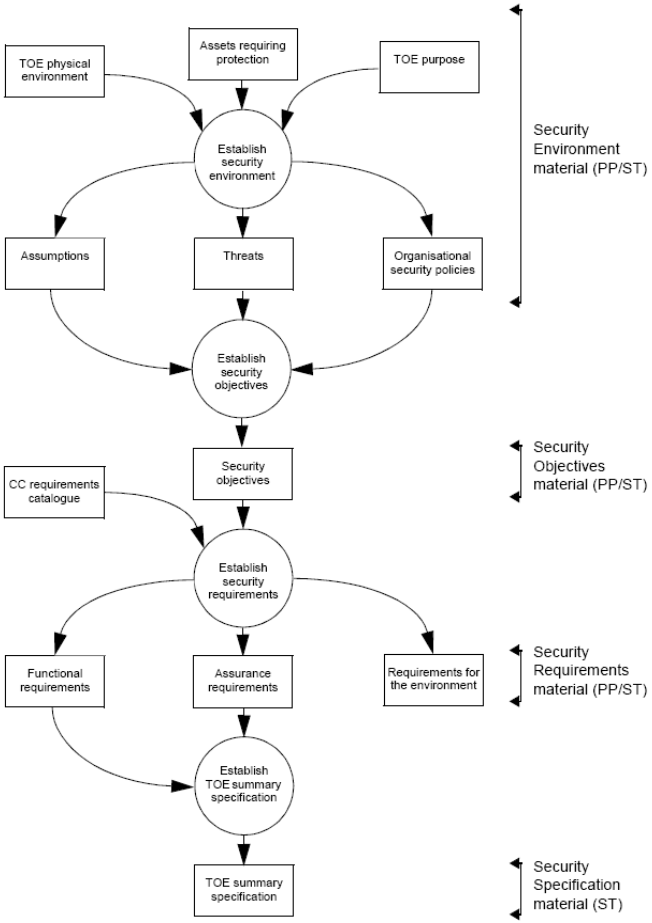
III. role, kterou výpočetní technika hraje v tomto konání
   A. objekt útoku
   B. nástroj
   C. prostředí
   D. symbol

IV. použitých prostředků
   A. opisování údajů
   B. špionáž
   C. vkládání falešných dat
   D. krádež
   E. odposlech
   F. scanování, prohledávání - kupříkladu hledání hesel zkoušením, hledání tfn. linek, které vedou k počítači, …
   G. piggybacking, tailgating - útočník se snaží projít vstupní kontrolou zároveň s autorizavanou osobou, nebo pokračovat v započaté session
   H. trojské koně - programy, vykonávající skrytou funkci
   I. viry
   J. trapdoors - skryté vstupy do systému, utajené příkazy umožňující přeskočit některé části procesu
   K. logické bomby - části kódu spouštěné výskytem určitých okolností - čas, dosažený obrat, stav systému
   L. salami attack - využívání zaokrouhlovacích chyb, drobné úpravy na hranici přesnosti zpracovávaných dat
   M. prosakování dat
   N. pirátství

Standardizace systémů spravujících senzitivní informace







Standardy

Proč normy a standardy?

- napoví, co máte chtít a jak to má vypadat
- certifikáty o shodě s normou zajistí, že nemusíte být experty, abyste si mohli vybrat správně
- zavádějí jednotnou kulturu a stanovují srovnatelná kriteria
- normu lze použít jako vodítko, abyste na nic nezapomněli

- usnadňují audit, kontroly, jednání s partnery

Podle míry utajení a spolehlivosti, které by měl systém poskytovat je podrobován různě rigorózním testům, v kterých musí obstát. Tento proces se nazývá *validace*.
lze použít několik způsobů validace:

- formální verifikace - celý systém je popsán soustavou logických formulí, tato soustava je redukována na tvrzení o bezpečnosti systému, v rámci verifikace je třeba ověřit správnost převodu
- validace - je obecnější metoda, zahrnuje verifikaci a další metody
  - ♦ testování požadavků - testuje se, zda je splněn každý z požadavků na funkčnost systému
  - ♦ kontroly návrhu a kódu - kontroly prováděné v průběhu tvorby systému
  - ♦ testování modulů a celého systému - ověřování funkčnosti na zkušebních datech
- Tiger Team Penetration Testing - dnes opět používaná metoda, nezávislý tým odborníků pověřen úkolem provést průlom bezpečnostními mechanismy

Pokud systém obstojí při validaci, může mu být vystaven v rámci následné *certifikace* certifikát, který je formálním vyjádřením shody s požadavky příslušné normy.

Zdroje požadavků na bezpečnost:

- zákony (např. 227/2000 Sb., 148/1998 Sb., 101/2001 Sb, 440/2005 Sb.)
- oborové normy
- technické standardy
- vnitrofiremní směrnice
- požadavky obchodních partnerů
- ...

Orange Book

Trusted **C**omputer **S**ystem **E**valuation **C**riteria
tvůrcem Ministerstvo obrany Spojených států, první ucelená technická norma
systémy rozděleny do čtyř základních tříd, dále dělení na podtřídy

D, C1, C2, B1, B2, B3, A1

třída D - žádná ochrana

třída C1 - volná ochrana

Oddělení uživatelů od dat, musí existovat metody umožňující uživatelům chránit vlastní data před ostatními, uživatel zvolí, zda tyto mechanismy bude používat
IBM MVS+RACF

třída C2 - Kontrolovaný přístup

Systém stále provádí volnou ochranu zdrojů, granularita však musí být až na úroveň jednotlivých uživatelů, musí být veden access log. Navíc ochrana proti *residuím* - obsahy paměti, registrů, … poté, co proces přestane tyto používat. Residua nesmí být zpřístupněna někomu jinému.
VMS, IBM MVS+ACF2

celá třída C je označována jako *optional protection* – musí být k dispozici příslušný mechanismus, který uživatel může použít

třída B1 - značkovaná ochrana

Každý kontrolovaný subjekt a objekt musí mít přiřazen stupeň utajení a musí být tímto stupněm označen, každý přístup musí být ověřován dle Bell-LaPadula modelu, musí existovat popis implementovaného formálního modelu, systém je podrobován testování

třída B2 - Strukturovaná ochrana

musí být k dispozici verifikovatelný globální návrh systému, systém musí být rozdělen do dobře definovaných nezávislých modulů, návrh musí zohledňovat princip nejmenších možných oprávnění, bezpečnostní mechanismy musí být uplatňovány vůči všem subjektům a objektům včetně všech zařízení, musí existovat analýza možných skrytých kanálů
vlastní systém musí běžet v rámci své bezpečnostní domény a provádět kontroly své integrity
Multics

třída B3 - Bezpečnostní domény

Systém musí být podrobitelný extenzivnímu testování, musí existovat úplný popis celkové struktury návrhu systému, musí být konceptuálně jednoduchý

musí existovat ochranné mechanismy na úrovni jednotlivých objektů, každý přístup musí být testován, kontrola na úrovni provádění jednotlivých typů přístupu daného subjektu
Systém musí být vysoce odolný vůči průnikům. Zařízení provádějící audit log musí umět odhadnout hrozící nebezpečí.

celá třída B je zonačována jako *mandatory protection* – musí být k dispozici odpovídající mechanismus, který uživatel nemůže obejít ani deaktivovat

tříba A1 - Verifikovaný návrh

Návrh systému musí být formálně verifikován, existuje formální model bezpečnostního mechanismu s důkazem konzistentnosti, formální specifikace systému s ověřením, že odpovídá formálnímu modelu, ověření, že implementace není odchylná od formální specifikace, formální analýza skrytých kanálů
SCOMP, patrně KVM/370, PSOS, KSOS

ITSEC

The **I**nformation **T**echnology **S**ecurity **E**valuation **C**riteria
mezinárodní sada kriterií, nadmnožina TCSEC
kriteria rozdělena na třídy funkčnosti (F) a korektnosti (E)
třídy funkčnosti F-D, F-C1, F-C2, F-B1, F-B2 a F-B3 zhruba co do funkčnosti odpovídají třídám C1 až B3 hodnocení TCSEC
kriteria hodnocení funkčnosti rozdělena na hodnocení integrity systému (F-IN), dostupnosti systémových zdrojů (F-AV), integrity dat při komunikaci (F-DI), utajení komunikace (F-DC) a bezpečnosti v rámci celé sítě (F-DX)
každé z těchto kriterií může být vyhodnocováno nezávisle, vyhodnocování prováděno pro požadovanou třídu funkčnosti

kriteria pro hodnocení korektnosti přidána pro zvýšení důvěryhodnosti systému
požadavky vyšší třídy korektnosti vždy nadmnožinou předchozích
E1 - testování
E2 - kontrola konfigurace a distribuce
E3 - ověření detailního návrhu a zdrojového kódu
E4 - zevrubná analýza slabin systému
E5 - důkaz, že implementace odpovídá detailnímu návrhu
E6 - formální modely, formální popisy a jejich vzájemná korespondence
tyto třídy odpovídají požadavkům na důvěryhodnost kladeným třídami C2 až A1 hodnocení TCSEC

kromě zmíněných kriterií hodnocení zabezpečených systémů existuje celá řada norem a doporučení upravující prakticky všechny podstatné rysy chování a architektury těchto systémů

Common criteria

metanorma stanovující principy a postupy, jak odvozovat konkrétní technické normy pro vývoj, testování, výsledné vlastnosti a provoz technických bezpečnostních protiopatření v různých prostředích
úzce souvisí s materiálem Common Evaluatin Methodology – pravidla pro vyhodnocování konkrétních systémů (target of evaluation) vůči daným požadavkům
formalizuje proces vyhodnocování:
evaluační kriteria → evaluační metodologie → evaluační schéma → evaluace → výsledky evaluace → certifikace → registr certifikátů
vyžaduje síť zkušebních laboratoří
odděluje funkcionalitu (sec. functional requirements) od „jistoty“ (sec. assurance req.)
sada konkrétních funkčních a „jistotních“ požadavků tvoří *profil zabezpečení (protectin profile)*
Funkční (functional) třídy:

- FAU – bezpečnostní audit
- FCO – komunikace
- FCS – kryptografická podpora
- FDP – ochrana uživ. dat
- FIA – identifikace a autentizace
- FMT – bezpečnostní management
- FPR – soukromí
- FTP – ochrana bezp. mechanismu
- FRU – využívání prostředků
- FTA – přístup
- FTP – důveryhodná cesta/kanál

Jistotní (assurance) třídy:

- ACM – správa konfigurací
- ADO – dodávka a provoz
- ADV – vývoj
- AGD – dokumentace, návody
- ALC – podpora životního cyklu
- ATE – testování
- AVA – vyhodnocení slabin

Vyhodnocení kvality bezpečnostního mechanismu v rámci evaluačních kriterií potom podléhá následující klasifikaci:

- APE – vyhodnocení profilu bezpečnosti
- ASE – vyhodnocení cíle hodnocení

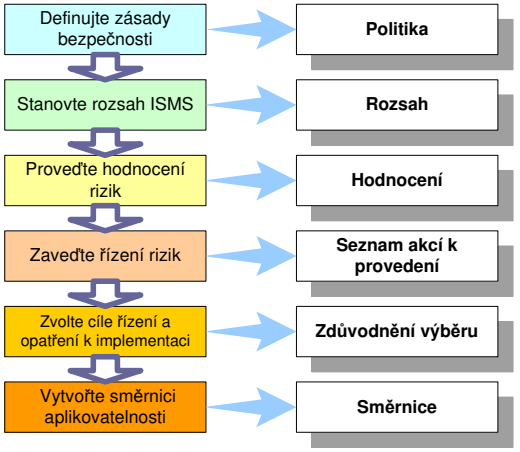
Úrovně vyhodnocení (eval. assurance level) dle kriterií:

- EAL1 – funkční testování

- EAL2 – strukturální testování
- EAL3 – metodické testování a kontroly
- EAL4 – metodický návrh, testování a ověření
- EAL5 – semiformální návrh a testování
- EAL6 – semiformálně verifikovaný návrh a testování
- EAL7 – formální návrh a testování

BS7799

(ISO IEC TR 17799, ISO 27000) organizační norma, která popisuje obecně, jaké činnosti musí organizace vykonávat pro zajištění bezpečnosti IS, nestanoví kvalitativní kriteria
založena na myšlence budování bezpečnosti shora dolů tj. od bezpečnostní politiky po implementaci protiopatření
předpokládaný proces tvorby bezpečnosti a z něho odvozené bezpečnostní dokumentace ukazuje obrázek



pokrývá tyto oblasti:

- bezpečnostní politika
- klasifikace a řízení aktiv
- personální bezpečnost
- fyzická bezpečnost a bezpečnost prostředí
- řízení provozu a komunikací
- řízení přístupu
- vývoj a údržba systémů
- řízení kontinuity operací
- soulad s požadavky (právní, technické, audit)

Pro podporu budování bezpečnosti a návrhu implementace bezpečnostních mechanismů podle BS7799 existuje standardní metodika a automatizovaný nástroj CRAMM.

Federal Information Processing Standard

FIPS 140-1: Security Requirements for Cryptographic Modules, January 4, 1994.
FIPS 140-2: Security Requirements for Cryptographic Modules, May 25, 2001. Change Notices 2, 3 and 4: 12/03/2002

Standardy PKCS

vytvářeny v laboratořích firmy RSA Security (dříve RSA)
představují ucelený soubor technických norem popisujících implementaci různých nástrojů asymetrikcé kryptografie
původně proprietární normy, dnes široce používaný de-facto standard
PKCS #1:RSA Cryptography Standard
PKCS #3:Diffie-Hellman Key Agreement Standard
PKCS #5:Password-Based Cryptography Standard
PKCS #6:Extended-Certificate Syntax Standard
PKCS #7:Cryptographic Message Syntax Standard
PKCS #8:Private-Key Information Syntax Standard
PKCS #9:Selected Attribute Types
PKCS #10:Certification Request Syntax Standard
PKCS #11:Cryptographic Token Interface Standard
PKCS #12:Personal Information Exchange Syntax Standard
PKCS #13: Elliptic Curve Cryptography Standard
PKCS #15: Cryptographic Token Information Format Standard

Zdroje standardů

Mezinárodní standardizační instituty

ISO - International Organization for Standardization http://www.iso.ch/
IEC - International Electrotechnical Commission http://www.iec.ch/
ITU - International Telecommunication Union http://www.itu.ch/
WSSN - World Standards Services Network http://www.wssn.net/

Regionální standardizační instituty

CEN - European Committee for Standardization http://www.cenorm.be/

CENELEC - European Committee for Electrotechnical Standardization http://www.cenelec.org/
COPANT - Pan American Standards Commission http://www.copant.org/
ETSI - European Telecommunications Standards Institute http://www.etsi.org/

Formální modely bezpečnosti

první fází tvorby bezpečného IS je volba vhodného bezpečnostního modelu
připomeňme dodržení základních požadavků bezpečnosti:

utajení, integrita, dostupnost, anonymita, …

dále budeme předpokládat, že umíme rozhodnout, zda danému subjektu poskytnout přístup k požadovanému objektu, modely poskytují pouze mechanismus pro rozhodování

Jednoúrovňové modely

jsou vhodné případy, kdy stačí jednoduché ano/ne rozhodování, zda danému subjektu poskytnout přístup k požadovanému objektu a není nutné pracovat s klasifikací dat !všechna data stejná!

Monitor model

též reference monitor

- subjekt při přístupu k objektu vyvolá tzv. *monitor* a předá mu žádost jakou akci s kterým objektem chce provést
- monitor žádost vyhodnotí a na základě informací o přístupových právech vyhoví či nikoliv

výhodou jednoduchost a snadná implementovatelnost
nevýhodou je, že proces poskytující služby monitoruje volán při každém přístupu k libovolnému objektu, což systém velmi zatěžuje
další nevýhodou je, že tento model je schopen kontrolovat pouze přímé přístupy k datům, ale není schopen zachytit např. následující případ

```
if profit <= 0
   then delete file F
   else
      write file F, "_zpráva_"
endif
```

subjekt mající legitimní přístup k souboru *F* může získávat informace o proměnné *profit*, k níž by přístup mít neměl

Information flow model

odstraňuje posledně jmenovanou nevýhodu předchozího modelu
autoři si všimli, že uživatel může získávat i jiné informace, než na které se explicitně ptá
již ve fázi vývoje je prováděno testování všech modulů, zda jejich výstupy závisí na interakcích se senzitivními daty a případně jakým způsobem
z těchto dílčích výsledků je sestavován celkový graf závilostí
veškeré požadavky na systém procházejí inteligentním filtrem, který zjišťuje, zda nedochází k nežádoucí kompromitaci informací

Víceúrovňové modely

v předchozích modelech jsme měli jednoduché vztahy objekt je/není senzitivní, subjekt má/nemá přístup k danému objektu
obecně však může být několik stupňů senzitivity a "oprávněnosti"
tyto stupně senzitivity se dají použít k algoritmickému rozhodování o přístupu daného subjektu k cílovému objektu, ale také k řízení zacházení s objekty
víceúrovňový systém „rozumí" senzitivitě dat a chápe, že s nimi musí zacházet v souladu s požadavky kladenými na daný stupeň senzitivity
(např. tajná data ukládáat pouze na konkrétní diskové pole, přísně tajná data posílat mimo systém výhradně zašifrovaná HW šifrátorem, ...)
rozhodnutí o přístupu pak nezahrnuje pouze prověření žadatele, ale též klasifikaci prostředí, ze kterého je přístup požadován (tj. uživatel je prověřen na vyhrazená data, ale sedí u stanice, která nemá klasifikaci „na vyhrazená data" a tudíž přístup není povolen).

Military security model

u zelených mozků je každá informace zařazena do některé z kategorií utajení (např. *unclassified, confidental, secret, top secret*), které jsou disjunktní
silné uplatnění zde má *princip nejmenších privilegií* - každý subjekt má mít pouze taková oprávnění, aby mohl konat svoji práci
všechny chráněné informace jsou rozděleny podle obsahu do *oblastí* (compartments), informace může být i několika oblastech zároveň
klasifikací informace potom rozumíme dvojici <*stupeň_utajení*, *oblasti*>
aby subjekt mohl používat požadovanou informaci, musí mít dostatečné *oprávnění*.
oprávnění má stejný tvar jako klasifikace - <*stupeň_utajení*, *oblasti*>, tedy daný subjekt smí používat informace až do *stupeň_utajení* v těchto *oblastech.*

$$O \le S \Leftrightarrow st\_utaj_O \le st\_utaj_S \wedge oblast_O \subseteq oblast_S$$

Relace $\le$ odpovídá *oprávnění* subjektu *S* k danému objektu *O*.
Požadavky na stupeň utajení bývají označovány jako hierarchické, rozdělení na oblasti jako nehierarchické omezení.

Svazový model (Lattice model)

předchozí military model je specielním případem tohoto modelu
relace $\le$ je částečným uspořádáním, množina klasifikací všech informací v systému tvoří svaz, stejně tak množina oprávnění všech subjektů
v růných oblastech se používá různých svazů, např. v komerční oblasti jsou obvyklé stupně utajení *public, company confidental, high security*, rovněž rozdělení do oblastí se liší případ od případu …
svazový model je často používaným modelem v mnoha prostředích

dále popíšeme dva modely, zabývající se tokem informací uvnitř systému

Bell-LaPadula model

model popisuje povolené přesuny informací, takové, aby bylo zajištěno jejich utajení
pro každý subjekt *S* resp. objekt *O* v systému nechť je definována bezpečnostní třída $C(S)$ resp. $C(O)$
bezpečné přesuny informací mají následující vlastnoti:
Vlastnost jednoduché bezpečnosti (Simple Security Property):

Subjekt *S* může číst objekt *O* právě když

$$C(O) \le C(S)$$

**-vlastnost* (*-Property):

Subjekt *S* mající právo čtení k objektu *O* může zapisovat do objektu *P* právě když

$$C(O) \le C(P)$$

Obyčejně nepotřebujeme tak silná omezení, která klade **-vlastnost*. Často je tato vlastnost poněkud oslabena v tom smyslu, že systém povolí zápis do objektu nižší bezpečnostní třídy, pokud zapisovaná data nezávisí na čtených údajích.
Model byl je používán v systémech, které paralelně zpracovávají informace různého stupně utajení.

Biba model

předchozí model se však vůbec nezabývá integritou dat, Biba model je duálním modelem k Bell-LaPadula modelu
Nechť pro každý subjekt *S* resp. objekt *O* v systému je definována integritní bezpečnostní třída *I*(*S*) resp. *I*(*O*). Obdobně jako v předchozím případě definujeme:
Vlastnost jednoduché integrity (Simple Integrity Property):

Subjekt *S* může modifikovat objekt *O* právě když

$$I(O) \leq I(S)$$

*Integritní *-vlastnost* (Integrity *-Property):

Subjekt *S* mající právo čtení k objektu *O* může zapisovat do objektu *P* právě když

$$I(O) \geq I(P)$$

Biba model se zabývá zajištěním integrity a tedy i důvěryhodnosti dat. Bepečnostní třída entity vpodstatě popisuje míru její důvěryhodnosti pro ostatní.
Tento model vůbec neřeší utajení dat.

Přestože byla učiněna řada pokusů o nalezení kompromisu mezi zajištěním integrity a utajení, dosud neexistuje obecně přijatý model, který by řešil oba problémy.

Modely pro různé účely

Clark-Wilson model

dobře odpovídá potřebám komerčních organizací, přejímá postupy běžné v účetnictví
základní principy:

1. dobře formované transakce (konzistentní data → konzistentní data)
2. separace operací – žádnou operaci nesmí být schopen korektně provést jediný subjekt

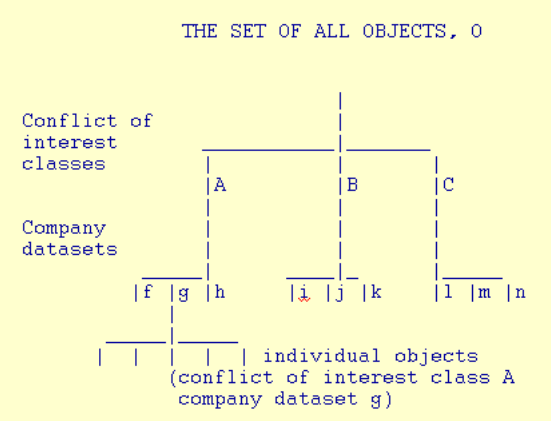
pravidla modelu rozdělujeme obvykle na požadavky na korektnost „C“ a na vynucení „E“
C1 – Všechny procedury testující validitu dat musí zajistit, že pokud doběhnou, všechna chráněná data jsou korektní.

C2 – Všechny používané transformační procedury musí být certifikovány, že po zpracování korektních chráněných dat zanechají chráněná data opět v korektním stavu.
E1 – Systém musí zajistit, že pouze procedury vyhovující požadavku C2 mohou pracovat s chráněnými objekty.
E2 – Systém musí udržovat seznam relací popisující, který subjekt smí spouštět které transformační procedury a musí zajistit dodržování těchto relací.
C3 – Seznam popsaný v E2 musí splňovat pravidlo separace operací.
E3 – Systém musí autentizovat každý subjekt pokoušející se spustit transformační proceduru.
C4 – Všechny transformační procedury musí zapisovat do append-only objektu (log) veškeré informace nezbytné pro rekonstrukci povahy provedené operace.
C5 – Každá transformační procedura zpracovávající nechráněná data musí buď skončit s tím, že chráněná data jsou v korektním stavu, nebo nesmí provést žádnou změnu.
E4 – Pouze administrátor provádějící certifikaci entit může provádět změny relací. V žádném případě nesmí mít právo spustit žádnou z procedur, které administruje.

Chinese wall model (Brewer-Nash)

Představuje příklad dynamického modelu – pravidla jsou generována až v okamžiku používání řízených objektů
„Konzultant musí zachovávat diskrétnost informací získaných v různých firmách, tj. nesmí radit konkurenční firmě na základě vnitřních znalostí jiné korporace. Může ale radit nekonkurenčním firmám, případně může dávat rady na základě obecných informací“.

objekty jedné organizace tvoří dataset, datasety rozčleněny do skupin (conflict of interest classes)
sanitizovaná informace – odstraněny ty části, které umožňují identifikovat konkrétního vlastníka



subjekt na počátku univerzální práva (ke všem objektům)

Vlastnost jednoduché bezpečnosti
Přístup je povolen pokud požadovaný objekt:

1. je ve stejném datasetu jako objekt, ke kterému subjekt již přistupoval, nebo
2. náleží do jiné skupiny

**-vlastnost*
Zápis je povolen pouze v případě že:

1. přístup je možný podle vlastnosti jednoduché bezpečnosti, a zároveň
2. není čten žádný objekt obsahující nesanitizované informace náležející do jiného datasetu než toho, do kterého se zapisuje

Závěrem dva teoretické modely pro modelování správy oprávnění

Graham-Denning model

model pracuje s množinou subjektů *S,* množinou objektů *O*, množinou práv *R* a přístupovou maticí *A*.
Každý objekt má přiřazen jeden subjekt nazývaný *vlastník,* každý subjekt má přiřazen jiný subjekt nazývaný *kontroler*.
Model definuje následující práva:

- *vytvořit objekt* - povoluje subjektu vytvořit v systému nový objekt
- *vytvořit subjekt*, *zrušit objekt*, *rušit subjekt* - obdobně jako předchozí

- *číst přístupová práva* - povoluje subjektu zjistit aktuální přístupová práva jistého subjektu k určitému subjektu
- *přidělit přístupová práva* - dovoluje vlatníku objektu přidělit jistá práva k objektu určitému subjektu
- *zrušit přístupová práva* - dovoluje vlastníku objektu resp. kontroleru subjektu odebrat danému subjektu jistá práva k objektu resp. subjektu
- *předat přístupová práva* - dovoluje subjektu předat některé ze svých práv jinému subjektu (každé oprávnění může být předatelné či nikoliv, obdrží-li subjekt předatelné právo, může jej dále předat jako předatelné či nepředatelné).

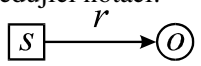
Následující tabulka uvádí podmínky nutné pro vykonání operací s přístupovými právy.

vytvořit objekt *o*	-
vytvořit subjekt *s*	-
zrušit objekt *o*	vlastník je v $A[x,o]$
zrušit subjekt *s*	vlastník je v $A[x,s]$
číst přístupová práva *s* k *o*	kontroler je v $A[x,s]$, nebo vlastník v $A[x,o]$
zrušit přístupové právo *r* subjektu *s* k *o*	kontroler je v $A[x,s]$, nebo vlastník v $A[x,o]$
přidělit *s* právo *r* k objektu *o*	vlastník je v $A[x,o]$
předat přístupové právo *r* nebo *r** k objektu *o* subjektu *s*	*r** je v $A[x,o]$

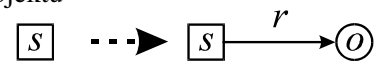
*r** označuje předatelné právo

Take-Grant system

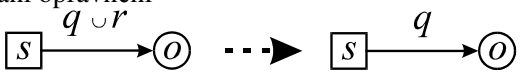
model pracuje s čtyřmi základními primitivami: *create, revoke, take, grant.*
předpokládáme, že systému obsahuje množinu subjektů *S,* množinu objektů *O*, objekty dělíme na aktivní (zároveň i subjekty) a pasivní (nejsou subjekty) a množinu práv *R*
Pro popis operací použijeme následující notaci:



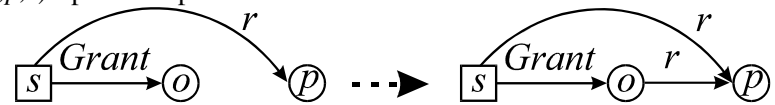
Subjekt *s* má k objektu *o* oprávnění *r*.
create(o,r) - vytvoření objektu



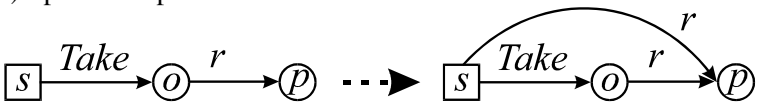
revoke(o,r) - odebrání oprávnění



grant(o,p,r) - předání oprávnění



take(o,p,r) - převzetí oprávnění



Výhodou popsaného systému je, že umožňuje v subpolynomiálním čase řešit dotazy na dostupnost jistého objektu pro daný subjekt.

Bezpečnostní politika

není pravda

- že bezpečnostní politika je jen „pro ty velké“
- že bezpečnostní politika je obrovské množství práce na celé měsíce
- že by bezpečnostní politika nic neřešila
- že vaše bezpečnostní politika musí být zcela jedinečná a šitá od počátku na vás.

Všechno je otázka míry

Bezpečnostní politika vám říká, jak zvládnout problém zajištění IS proti incidetnům.

Důležitější než rozsah je, aby pokrývala všechny důležité okruhy problémů formou, která je srozumitelná všem, kterých se týká.

umožní rozmyslet si, kde vás bota tlačí
materiál by neměl popisovat neexistující systém (zkoumaný IS se v průběhu zkoumání vyvíjí)

Organizace pohybující se ve stejné branži budou mít podobné nároky na bezpečnost

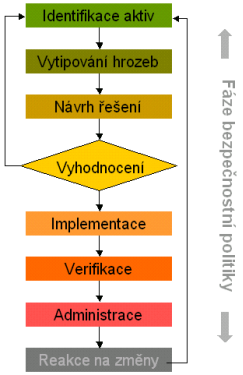
Výhody existence BP

stejně jako každá činnost, i provozování systému pro správu informací je spojeno s jistým rizikem (chyba zařízení, obsluhy, programu, vandalismus, krádež, …)
provedení kvalifikovaného odhadu rizik přináší:

- zlepšení obecného povědomí - pracovníci si problém uvědomí a mají šanci jej pochopit
- identifikace hodnot, slabin a možných kontrol celého systému - ne vždy je jasné, které části systému mají největší hodnotu, odkud pramení největší nebezpečí
- zlepšení východiska pro strategická rozhodnutí - některé ochranné a kontrolní mechanismy velmi snižují produktivitu systému přičemž jejich přínos není zřejmý, různé druhy nebezpečí jsou různě reálné a představují mnohdy daleko větší hrozbu, než by se dalo očekávat
- lepší rozložení výdajů na bezpečnost - některé velmi drahé ochranné mechanismy poskytují pouze malé zvýšení bezpečnosti a popřípadě i naopak

Jak na to

oblíbený obrázek ukazující životní cyklus bezpečnostní politiky
bezpečnost je proces - bez soustavného přizpůsobování se změnám vnějšího prostředí a vývoji vlastního IS je to celé k ničemu



Identifikace a odhad aktiv

Základem je zjistit, co vlastně ve svém informačním systému mám a k čemu je to dobré

... nikdo neví, kde fyzicky server leží, co dělá ta modrá krabice v rohu ... co se stane, když tenhle kus přestane fungovat ...

Přesnější výsledek docílíme sčítáním po jednotlivých kategoriích, např

- hardware - počítače, monitory, pásky, tiskárny, disky, komunikační media, …
- software - operační systém, koupené programy, vlastní zdrojové kódy, knihovny
- data - vlastní uložená data, logy, archivní kopie, listingy, …
- lidé - pracovníci potřební k správnému chodu systému, správci, programátoři

- dokumentace - programů, technického vybavení, systému, administrativní postupy
- spotřební materiál - papír, diskety, tonery, pásky do tiskáren, …
- zákazníci
- image společnosti
- ... atd. Zjevně každý si musí vymyslet vlastní seznam.

V podstatě v tomto kroku provedeme zevrubnou inventarizaci celého systému. Cena některých částí může být pouze velmi přibližně odhadnuta a i takový odhad může být velmi obtížný.

Vytipování hrozeb

je potřeba určit, co nás bude stát realizovaný bezpečnostní incident

zkoumáte:

- kolik vás bude stát náprava (nové pořízení)
- kolik přijdeme (tj. kolik nevyděláme)

... za kolik si pořídíte novou dobrou pověst seriózní firmy s dlouholetou tradicí ... kolik bude stát, když konkurence získá váš tajný návod na výrobu té nejlepší slivovice...

Příklady hrozeb:

- dopad přírodních katastrof - požár, vichřice, záplavy, výpadky napájení, selhání techniky
- poškození třetími osobami - přístupy po síti, vytáčená spojení, hackeři, kolem-jdoucí, lidé zkoumající odpad firmy

Dotazníček

Hodnota	*Utajení*	*Integrita*	*Dostupnost*
Hardware		přetížení, zničení, poškození	
Software	odcizen, kopírován	modifikován	smazán, přesunut
Data	zpřístupněna vně firmy	zničena chybou SW ; HW ; lidí	smazána
Lidé			únava, nemoc
Dokumentace			ztracena, odcizena

- následky zlomyslných pracovníků - zklamaní pracovníci, úplatkářství, zvědavci
- důsledky neúmyslných chyb - zadání špatných příkazů, vadných dat, skartace špatných dokumentů, kompromitace tajných materiálů
- ... a asi tisíc dalších

Zjišťování těchto faktů lze provádět formou dotazníku, který vyplní zainteresovaní pracovníci (viz. obrázek).

Odhad pravděpodobnosti zneužití

zjistíme, jak často dojde ke zneužití některé z expozic systému

pomáhá přemýšlet o věci nikoliv jako o pravděpodobnosti, ale jako o četnosti

když ani toto nepomáhá (jak často přijde do Prahy tisíciletá voda), lze použít některou z následujících metod:

- Odhad na základě obecných dat - např. pojišťovny mají rozsáhlé záznamy o počtu katastrof a o průměrných způsobených škodách, výrobci mají přehled o životnosti a počtu selhání zařízení, …
- Odhad na základě vlastních dat - za dobu činnosti firmy vzniklé záznamy o závadách zařízení, počtech vadných loginů, …
- Bodovací systém počtu výskytů události – např. dle tabulky
- *Delfská metoda* - okruh hodnotitelů provede hodnocení dané veličiny. Poté je každý seznámen s výsledky ostatních a upraví své hodnocení. Pokud jsou upravená hodnocení podobná, máme výsledek, v opačném případě výsledek vznikne dohodou hodnotitelů.

Frekvence	*Hodnocení*	*Frekvence*	*Hodnocení*
více než 1 x za den	10	1 x za měsíc	5
1 x za den	9	1 x za 4 měsíce	4
1 x za 3 dny	8	1 x za rok	3
1 x za týden	7	1 x za 3 roky	2

Výpočet očekávaných ročních ztrát

Stačí prostě vynásobit odpovídající dopady a pravděpodobnosti a vše sečíst

nadhodnocení dopadů a četností může vést ke zcela nesmyslným odhadům ztrát

kvalifikovaný odhad ztrát bývá často vyšší, než se obvykle předpokládá

Postprocessing

Poté, co jsme zjistili, jaké jsou naše problémy, je třeba najít zodpovědět:

- Jaké právní normy chrání utajení a integritu dat?
- Jaké další normy je nezbytné dodržet?
- Co nás bude stát, pokud se na shora uvedené skutečnosti nebudeme brát ohled.

když jde o peníze, musí zhusta city stranou.

Návrh řešení / Přehled použitelných ochranných mechanismů

Když je třeba zavést nové ochranné mechanismy:

Můžete

- probrat jednotlivé expozice systému a zkoumat možnosti jejich pokrytí
- mezi všemi ochrannými mechanismy hledat nějaký, který by řešil náš problém.

uvažte, že stejně nevyrazíte ze svého IS nic, co by sám od sebe neuměl

Výsledkem je seznam navrhovaných opatření.

Verifikace / Nástin ročních úspor ze zavedení ochranných mechanismů

S bezpečností je jeden problém – nic užitečného to nedělá

spočítat odhad očekávaných ztrát v aktuálním stavu

známe cenu zavedení nových ochranných mechanismů
znovu vyčíslíme očekávanou ztrátu po zavedení těchto opatření

rozdíl těchto hodnot je **odhad celkových úspor**

Struktura bezpečnostního plánu

bezpečnostní plán popisuje, jak daná organizace přistupuje k otázkám bezpečnosti
plán musí být dostatečně často revidován a musí být zkoumáno jeho dodržování
vypracováním plánu bývá pověřena skupina odborníků pokud možno ze všech důležitých organizačních struktur firmy, velikost a struktura tohoto týmu závisí na velikosti firmy
součásti bezpečnostního plánu:

Pokrytí

přesný popis, jakými oblastmi IS se zabývá, jaké hrozby uvažuje

Bezpečnostní politika

vyjadřuje vůli pracovat na dosažení jistého stupně bezpečnosti
bývá rozdělena do více dokumentů

1. Statement (záměr bezpečnosti) – cca 1 strana základního záměru, podepsaná top managementem
2. Politika a principy bezpečnosti – podle potřeby i desítky stran
3. Navazující dokumenty a směrnice (viz níže) – dokumenty různé povahy, srozumitelné pro uživatele, nebo vysoce technické pro administrátory, dodavatele apod., v případě rozsáhlé organizace i tisíce stran

- popis celkových cílů bezpečnostních aktivit - např ochrana dat před katastrofami, před úniky mimo organizaci, apod.
- kdo má zodpovědnost za udržení bezpečnosti - pověřený pracovník, vedení, všichni
- závazky organizace na udržení bezpečnosti - počet vyčleněných pracovníků, minimální výdaje do této oblasti

Klasifikace hodnot

popis obsahuje seznam hodnot systému, soupis hrozeb pro tyto hodnoty a používané ochranné mechanismy
dále je popsán způsob získávání a vstupní validace dat, případně předpoklady o jejich vlastnostech
měly by být popsány metody odhalování slabin systému, popisy akcí, které je třeba podniknout v případě odhalení nové slabiny
odhady ztrát a dopadů
vlastníci

Analýza rizik

obecný pohled na situaci
detailní popis relevantních nezanedbatelných rizik

Doporučení

seznam dalších bezpečnostních opatření, které je třeba přijmout k doplnění, nebo nahrazení sočasných mechanismů
součástí by měl být rozbor nákladů a ztrát
seznam by měl být seřazen podle naléhavosti navrhovaných opatření, navrhována by měla být pouze opatření, jejichž celkový efekt není záporný

Odpovědnost za implementaci

je třeba určit konkrétní osoby zodpovědné za zavedení a provozování konkrétních bezpečnostních mechanismů, těmto lidem důkladně vysvětlit jejich úkol a důvody

též je nutné navrhnout způsob hodnocení splnění těchto úkolů
možné rozdělení zodpovědnosti:

- uživatelé osobních počítačů - každý ručí za svůj počítač
- administrátor databázového systému - zodpovídá za přístup k datům a jejich integritu
- firma může pověřit zvláštního pracovníka zodpovědného za vytvoření obecných pravidel práce s daty a jejich uvolňování či rušení
- pracovníci osobních oddělení zodpovídají za přijetí důvěryhodných a spolehlivých pracovníků

Časový rozvrh

některá opatření mohou být příliš nákladná, nebo složitá, než aby mohla být zavedena naráz
musí existovat plán, do kdy budou která opatření zavedena, případně nejzajší termíny splnění jednotlivých fází bezpečnostního plánu
též pořadí zavádění opatření může být důležité

Soustavná pozornost

je třeba již v plánu stanovit termín, kdy musí být provedeno nové zhodnocení bezpečnostní situace a ověření funkčnosti bezpečnostních aktivit
získaná ocenění hodnot a bezpečnostních rizik musí být průběžně aktualizována

Závazek dodržování bezpečnostního plánu

všichni pracovníci by měli být s bezpečnostním plánem seznámeni a měla by jim být vysvětlena jeho důležitost i jejich role v rámci plánu
podstatné je, aby vedení organizace přijalo závazek, že bude poskytovat dostatečnou podporu provádění bezpečnostního plánu

Bezpečnost

Hlavní komponenty bezpečnosti lze rozdělit takto:

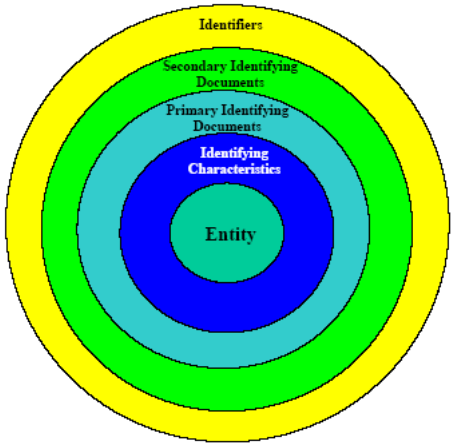
- kontrola prostředí
- autentizace / identita
- autorizace
- separace
  - fyzická
  - **časová**
  - logická
  - kryptografická
- integrita
- dostupnost
- auditabilita

Autentizace

jde o proces (mechanismus) zjištění/ověření identity subjektu
zásadní význam pro možnost aplikace bezpečnostních mechanismů
asociace subjektu/identity s příslušnou sadou autorizací

Správa identity

Identifikátory: jméno, userID, rodné číslo
Sekundární identifikující dokumenty: směnka, výplatní páska, permanentka, ...
Primární identifikující dokumentu: občanský průkaz, pas, dokumenty svázané přímo s identifikující charakteristikou (např fotografií, otiskem prstu)
Identifikující charakteristika: biometrika, fotografie, další prostředky rozpoznání jednotlivce
Entita: bytost, místo, věc



Identita uživatele

Rostou nároky na informace udržované o uživateli, prostou identifikaci nahrazuje komplikovaná struktura označovaná jako ***profil***

- userID, heslo
- jméno, příjmení, tituly, ...
- kontaktní informace
- příslušenství ke skupinám, organizačním jednotkám, ...
- certifikáty, klíče
- personalizace
- oprávnění
-
- ...

Další příbuzné pojmy:

- alias
- anonymita
- pseudonymita

Autentizace protistrany

systémy pro správu informací musí zajistit dodání těchto informací autorizovaným uživatelům
navíc autentizace je nutná i při zajišťování např. fyzické bezpečnosti
Mechanismus autentizace může být založen na některém z následujících faktů:

- Co ví (pouze) dotyčná osoba - heslo, pass-phrase, šifrovací klíč
- Co vlastní - token, schopnost
- Schopnost provést operaci
- Cosi charakteristického - biometriky

Hesla

Charakteristika dobrého hesla:

- Obsahuje kromě velkých a malých písmen též číslice a další na klávesnici se vyskytující značky
- Dostatečná délka
- Nejde o obvyklé slovo nebo známou frázi
- Nepravděpodobné - nelze jej odvodit ze znalosti osoby vlastníka
- Často obměňované

- Není nikde po okolí poznamenáno

Passphrase

velmi dlouhá "hesla" – třeba citát z knihy, říkanka, ...
lze i na biometriky

Skupinová hesla

z různých důvodů občas systémy připouštějí hesla společná skupinám uživatelů - tato hesla jsou málo bezpečná, bývají často vyzrazena

Piny

(personal identification number)
jsou číselné řetězce standardní délky, sloužící k podobným účelům jako hesla
v souvislosti s platebními a kreditními kartami často používány 4-místné piny

Challenge-Response systémy

heslo může být zachyceno v průběhu vkládání, nebo při přenosu cílovému uzlu
časté změny hesla jsou pro uživatele zatěžující
vhodnější je, pokud systém zašle výzvu v podobě náhodné zprávy a uživatel jako heslo vrátí správnou reakci na tuto zprávu - např. její zašifrování tajným klíčem apod.

Jednorázová hesla

implementováno pomocí tokenů

Vícefaktorová autentizace

kombinace několika autentizačních postupů, např. pin + smart karta
vyšší úroveň bezpečnosti

- několik nezávislých bezpečnostních mechanismů aplikovaných paralelně, nebo
- aktivace silnějšího mechanismu a následná autentizace za použití tohoto mechanismu

Výměna tajností

protokol pro případ, že komunikující strany příliš nedůvěřují svému okolí a nechtějí vyzradit svoji identitu
pokud sdílejí tajný klíč *e*:

1. *A* zašle *B* zprávu **E**(*m, e*)
2. *B* vrátí *A* zprávu **E**(*m* + <*heslo*>, *e*)

pokud tajný klíč nesdílejí, neobejdou se bez centrální autority *C*:

1. *A* zašle *C* zprávu {*B, m*} e_A
2. *C* vytvoří transakční klíč *k*
3. *C* zašle **E**(<*B, m, k,* **E**(<*A, m, k*>, e_B)>, e_A) zpět *A*
4. Dešifrováním zprávy *A* získá *m, k* a **E**(<*A, m, k*>, e_B)
5. *A* zašle **E**(<*A, m, k*>, e_B) uzlu *B*

pro zajištění ochrany proti znovupoužití starých zpráv *m* musí obsahovat timestamp

Asymetrické klíče

Schopnost provádět operace tajným klíčem jednoznačně identifikuje držitele (dokazovatel) tohoto klíče:

1. ověřovatel zašle dokazovateli náhodně volený řetězec
2. dokazovatel jej transformuje za použití tajného klíče
3. ověřovatel pomocí odpovídajícího veřejného klíče ověří správnost

Symetrické klíče

protokol běží stejným způsobem jako v případě asymetrických klíčů, pouze v tomto případě může ověřovatel napodobovat (impersonation) dokazovatele

Passphrases

jde vlastně o dlouhá hesla, mohou to být části písní, básniček, části citátů …
pokud použijeme vhodný kompresní algoritmus, lze passphrázi transformovat ve velmi kvalitní heslo
navíc je možné aplikovat různé další měření - např. rytmus stisku jednotlivých kláves, jež bývá pro každého charakteristický

Tokeny, smart cards

token je obecné označení pro předmět, který autentizuje svého vlastníka
musí být jedinečný a nepadělatelný
obvyklá implementace jsou nejrůznější magnetické nebo čipové karty
pokud karta umí reagovat na vnější podněty, má např. vlastní výpočetní kapacitu, paměť, hovoříme o tzv. *smart card*
předložení tokenu bývá často kombinováno s nutností zadat odpovídající heslo

Tokeny pouze s pamětí

jsou obdobou mechanických klíčů, paměť může obsahovat jednoznačný identifikační řetězec

Tokeny udržující hesla

token po zadání jednoduchého uživatelského hesla vydá určený kvalitní klíč, který udržuje

Tokeny s logikou

umí zpracovávat jednoduché podněty typu vydej následující klíč, vydej cyklickou sekvenci klíčů, může mít omezen počet použití
pomocí těchto tokenů lze realizovat systém s one-time hesly
každý klíč cyklické sekvence může zpřístupňovat jistou část výpočetního systému
tyto tokeny lze používat též k ochraně programů, přístupům k nejrůznějším placeným službám apod.

Inteligentní tokeny (smart cards)

jsou ideálním doplňkem chalenge-response systémů, mohou mít vlastní vstupní zařízení pro komunikaci s uživatelem, vlastní časovou základnu, mohou zajišťovat např. šifrování, generovat náhodná čísla apod.

Biometriky

jde o techniky identifikace lidí na základě jejich osobních charakteristik
navzájem se odlišují různou mírou spolehlivosti, ceny a v neposlední řadě i společenské přijatelnosti
hledáme charakteristiky mající dostatečnou mezi-osobní variabilitu při zachování vnitro-osobní reprodukcibility
kvalitu biometrik lze charakterizovat:

- četnost nesprávných odmítnutí - autorizovaného subjektu
- četnost nesprávných přijetí - útočníka
- kvalitou senzorů (!)

Verifikace hlasu

testovaný subjekt přečte systémem náhodně zvolenou frázi, sejmutá zvuková stopa je kmitočtově omezena (nejčastěji na 3kHz) a je proveden rozbor zvuku na základě původu jednotlivých složek zvuku v činnosti hlasového aparátu - fonace, frikace.

Výsledek je vhodným způsobem komprimován na vzorek velikosti 1 až 2 kB a porovnán se srovnávacím vzorkem
Výhodou je přirozenost a možnost provádět verifikaci např. prostřednictvím telefonu.

Verifikace dynamiky podpisu

Sledují se změny tlaku, zrychlení v jednotlivých částech, celkový průběh zrychlení, zarovnání jednotlivých částí podpisu, celková rychlost, celková dráha a doba pohybu pera na a nad papírem apod.
Ze získaných hodnot je opět vytvořen vzorek, který je porovnán se srovnávacím vzorkem.
Výhodou opět přirozenost a sociální akceptovatelnost, nevýhodou malá mechanická odolnost snímačů, a značná variabilita podpisu u některých lidí.

Verifikace otisků prstů

Systém provádí statistický rozbor výskytu tzv. markant - hrbolků, smyček a spirál v otisku prstu a jejich vzájemné polohy
často se provádí testování uživetelem zvoleného výběru několika prstů
Výhodou je vynikající mezi/vnitro-osobní variabilita, a dobrá zpracovatelnost vstupních dat, nevýhodou jsou možné negativní asociace uživatelů, a mnohdy sporná spolehlivost snímačů

Geometrie ruky

Metoda zkoumá délku a šířku dlaně a jednotlivých prstů, boční profil ruky apod.
Výsledkem je velmi malý vzorek - cca 18 bytů. Metoda je poměrně spolehlivá avšak poněkud dražší. Možnost podstrčení odlitku ruky.

Obrazy sítnice

Zařízení pořídí obraz struktury sítnice v okolí slepé skvrny, tento obraz je digitalizován a převeden na vzorek délky přibližne 40 bytů (!)
Obrázky sítnice mají stejné charakterizační vlastnosti jako otisky prstů
Výhodnou metody je značná spolehlivost a velmi obtížná napodobitelnost. Proto jde o metodu vhodnou k nasazení v prostředí nejvyššího utajení. Nevýhodou jistá subjektivní nepříjemnost, opět jde o velmi drahou technologii.

Další biometriky

rysy obličeje, Bertillonovy míry, rytmus psaní na klávesnici, EEG, EKG, otisky dlaní a chodidel, otisky chrupu, genetické rozbory, …

Autorizace

informační systém může poskytovat různé **úrovně ochrany objektů**
1. *žádná ochrana* - postačující pokud dochází k samovolné časové separaci
2. *isolace* - (semi)paralelně běžící procesy jsou zcela odděleny a vůbec o sobě vzájemně neví, systém zajišťuje úplné ukrytí objektů ostatních procesů
3. *sdílení všeho nebo ničeho* - vlastník objektu deklaruje, zda je objekt přístupný všem (public), nebo soukromý (private) a tedy viditelný jen pro něho
4. *sdílení s omezenými přístupy* – testuje se oprávněnost každého pokusu o přístup k danému objektu, k danému objektu a subjektu existuje záznam zda subjekt má právo – konkrétním způsobem - přistupovat k příslušnému subjektu
5. *sdílení podle způsobilosti* (share by capability) - nadstavba předchozího způsobu sdílení, rozsah oprávnění může dynamicky záviset na aktuálním kontextu
6. *limitované použití objektů* - nespecifikuje pouze zda subjekt smí přistupovat k danému objektu, ale i operace, které subjekt smí s objektem provádět

seznam je seřazen podle (implementační) složitosti, které tentokrát přímo úměrně odpovídá kvalita poskytované ochrany

Granularita - kontrola přístupu může být implementována na různých úrovních (byte, věta, soubor, adresář, …), je potřebné volit mezi režií kontroly a dostatečně jemným rozlišením

- paměť
- soubory a data na záznamových zařízeních
- běžící programy
- adresáře souborů
- hardwarová zařízení
- různé datové struktury (stack,…)
- interní tabulky OS
- různé instrukce
- hesla a autentizační mechanismy
- vlastní ochranný mechanismus

cíle ochrany objektů
Kontrolovat každý přístup - subjekt může pozbýt přístupová práva a tedy je nutno mu zabránit v dalším používání objektu
Povolení co nejmenších práv - subjekt by měl mít pouze nejmenší možná oprávnění nutná ke korektnímu plnění jeho úkolu a to i v případě, že případná další práva by

pro něj byla bezcenná - toto uspořádání snižuje možnost průniku v případě selhání části ochranného mechanismu
Ověření přijatelného používání - někdy je daleko podstatnější než přidělení či odepření přístupu moci kontrolovat, co subjekt s daným objektem provádí

Mechanismy ochrany obecných objektů

Adresář (directory)

metodu popíšeme pro případ uživatelů systému v roli subjektů a souborů coby objektů, lze ji však snadno rozšířit na libovolné objekty a subjekty
každý soubor má svého vlastníka, který k němu vlastní veškerá práva včetně práva určovat rozsah oprávnění ostatních uživatelů k tomuto souboru
s každým uživatelem je spojena specielní struktura - *adresář* - obsahující odkazy na všechny soubory, k nimž má daný uživatel nějaké oprávnění, včetně popisu tohoto oprávnění
žádný uživatel nesmí zapisovat do svého adresáře

Nevýhodou může být velký rozsah adresářů a velmi obtížná správa a úpravy takto přidělovaných oprávnění. Rovněž udržení přehledu o tom, kdo k danému souboru má jaká práva může být problematické.

Seznam oprávnění (Access Control List)

opačný přístup k problému
tentokrát je s každým *objektem* udržován seznam informací, které subjekty k němu mají jaká oprávnění
metoda umožňuje snadno přidělovat implicitní práva subjektům případně skupinám subjektů
při vhodném označení subjektů a použití expanzních znaků může být tato metoda dostatečně pružná

Př: Pepek_Group1_Troja
*_Group1_*

seznamy zpravidla bývají udržovány setříděné tak, že záznamy s expanzními znaky jsou na konci - tak stačí hledat první shodu s identifikací subjektu a použít tímto záznamem specifikované oprávnění

Přístupová matice (Access Control Matrix)

řádky matice odpovídají jednotlivým subjektům, sloupce objektům
v políčku daném řádkem a sloupcem je záznam o úrovni oprávnění odpovídajícího subjektu k příslušnému objektu
přístupová matice je zpravidla velmi velká záležitost, zhusta řídká

Způsobilost (Capability)

Způsobilost budeme chápat jako nefalšovatelný token, jehož vlastnictví dává vlastníkovi specifická práva k danému objektu. Lze chápat jako lístek do kina.

jednou z metod zajištění nefalšovatelnosti je, že tokeny se nepředávají přímo subjektům, ale jsou udržovány v chráněné oblasti paměti, přístupné pouze systému

při přístupu k objektu tak systém zkontroluje existenci příslušného tokenu, tento postup lze urychlit tím, že zvlášť udržujeme seznam *Způsobilostí* právě běžícího procesu
výhodou metody je, že dovoluje definovat nové dosud neznámé způsoby používání objektů a přidělovat odpovídající oprávnění
nevýhodou opět poněkud obtížná správa těchto tokenů, zejména odebrání Způsobilosti je netriviální operace

Security Labeling

s každým subjektem a objektem asociujete bezpečnostní label popisující pověření/klasifikaci entity
podpora víceúrovňových modelů

Procedurálně orientovaný přístup

namísto přidělování obecného přístupu k subjektu (čtení, zápis, …) můžeme přidělovat právo používat některých funkcí z rozhraní, prostřednictvím kterého je objekt zpřístupňován
metoda podporuje koncept skrývání a zapouzdřování informací popsaný v minulé lekci
nevýhodou je jistá ztráta efektivity a rychlosti přístupu

Zvládání granularity autorizace

Ochrana po skupinách

uživatelé jsou podle svého zaměření, pracovního zařazení, …, vhodně rozděleny do skupin
pro účely ochrany objektů je svět rozdělen na vlastníka souboru, skupinu, do ktere vlastník patří a ostatní uživatele
předpokládá se, že uživatelé v rámci skupiny potřebují sdílet data
při vytvoření objektu vlastník specifikuje, jaká práva přiděluje sobě, uživatelům ve stejné skupině, ostatním
metoda je jednoduchá, snadno implementovatelná leč neposkytující dostatečně jemné rozlišení, navíc je většinou nutné, aby každý uživatel byl právě v jedné skupině, jinak nastávají problémy s přidělováním práv skupinám

Hesla nebo jiné tokeny

při vytvoření objektu vlastník specifikuje hesla, potřebná pro jisté módy přístupu k objektu, heslo zašle uživatelům, kteří mají mít přístup
systém splní žádost o přístup k objektu pouze tomu, kdo se prokáže odpovídajícím heslem
nevýhodou je, že v případě zapomenutí není možno zjistit, jak heslo vypadalo, v případě, že dojde k vyzrazení hesla je složité nastavit nové, stejně obtížné je odejmout právo přístupu

Dočasné propůjčení oprávnění

mechanismus známý ze systému UNIX.
stejné přidělování práv jako v případě ochrany po skupinách, navíc je možno stanovit, že (spustitelný) soubor smí být prováděn s oprávněním vlastníka
prostřednictvím rutin běžících s oprávněním vlastníka lze řízeně přistupovat k objektům, ke kterým uživatel přímý přístup nemá

problémem popsaných schémat je jistá těžkopádnost, uživatel nemůže selektivně přidělovat práva jistým uživatelům k jistým skupinám objektů
kontrolní matice a podobné metody jsou zase příliš rozsáhlé a obtížně spravovatelné

VAX VMS/SE

ke každému souboru může uživatel vytvořit *Seznam oprávnění* udávající kdo má jaká práva

každý uživatel je členem jedné skupiny, navíc administrátor může vytvořit skupinu typu *obecný identifikátor* , a tuto skupinu mohou uživatelé uvádět v *Seznamech oprávnění*
Seznamy oprávnění mohou být též použity pro přidělování přístupu k ostatním systémovým zdrojům

Systém rolí a skupin

oprávnění jsou sdružována do ucelených souhrnů – tzv. rolí – které odpovídají svým obsahem okruhu práce, kterou vykonává pracovník na určitém zařazení (správce uživatelů, finanční účetní, skladník, ..)
uživatel nezískává oprávnění „po jednom“, ale přidělením role
pro zjednodušení práce bývá k dispozici systém kompozitních rolí, odvozených rolí atd.
namísto práce s jediným uživatelem může být možné definovat a hromadně spravovat celé supinu uživatelů, majících stejná oprávnění

Referenční uživatelé

předpřipravené vzory častých typů uživatelů obsahující např. přiřazené role oprávnění, personalizaci, nastavení ...
- ulehčují správu oprávnění

Metody fyzické ochrany

fyzická ochrana se snaží eliminovat případnou hrozbu ještě dříve, než přijde do přímého kontaktu s vlastním výpočetním systémem
možná nebezpečí můžeme rozdělit:

- přírodní katastrofy
- vandalové
- odposlech
- neautorizované používání

Přírodní katastrofy

není jim možné předcházet, je třeba se soustředit na omezení možného odpadu a na odstranění případných následků

Záplavy

v podstatě dvou druhů
stoupající voda - většinou bývá dost času odstavit systém a přinejmenším datové nosiče přesunout do bezpečí. Pro tyto účely by každá komponenta systému měla být jasně vyznačen stupeň důležitosti, aby s odsunem mohli efektivně pomáhat i nekvalifikovaní pracovníci.
padající voda - např. poruchy potrubí, izolací apod. Tyto záplavy bývají velmi rychlé, v první fázi stačí vhodný nepromokavý kryt a následný odsun zařízení. Opět vhodné vyznačení stupňů důležitosti komponent.

Požáry

představuje často nebezpečí nejen pro techniku, ale i pro obsluhující personál
je vhodné mít vyzkoušen postup zahrnující bezodkladné odstavení systému a evakuaci personálu a životně důležitých komponent systému
je třeba vhodně volit automatické protipožární systémy chránící prostory výpočetního systému, vhodné je umístit nejdůležitější části systému v prostorách s vysokou pasivní požární bezpečností

Ztráty napájení

pro nejdůležitější části systému je nutné zajistit náhradní zdroje energie, které jsou v případě výpadku hlavního napájení schopny dostatečně rychle zajistit dodávky elektřiny
na kratší dobu jsou to různé akumulátory a UPS zdroje, v případě potřeby překonávat delší výpadky pak agregáty na výrobu el.e.
důležité jsou rovněž filtry a přepěťové ochrany chránící zařízení před výkyvy napětí, blesky apod.

Chlazení

některé komponenty jsou citlivé na teplo, při ztrátě chlazení může dojít k jejich zničení
větší systémy nelze bez odpovídajícího chlazení provozovat ani krátkodobě, příslušná technologie se tak může stát klíčovou komponentou

Hmotnost

serverová technologie může vyžadovat speciální podlahy se zvýšenou nosností, může se stát problémem zejména při zařizování záložního centra po havárii
jindy lze hmotnost využít jako bezpečnostního mechanismu

Prašnost, vibrace, další vlivy

... mohou i výrazným způsobem omezovat životnost komponent, případně zvyšovat poruchovost

Prostorová ochrana

prostředky umožňující zabránit potenciálním útočníkům ve vstupu do prostor, kde jsou instalovány důležité komponenty systému, nebo vnesení potenciálně nebezpečných předmětů či detekci takové skutečnosti, případně může být cílem zabránění opuštění prostoru, respektivě vynesení komponent
předmětem útoku můžou být:

- vlastní počítače
- výměnná záznamová media
- části počítačů (myši, světelná pera, …)
- vytištěné senzitivní materiály
- části síťové technologie,
- klíčoví pracovníci

metody ochrany:

Stráže

měly by být k dispozici nepřetržitě, musí osobně znát všechny pracovníky, nebo musí umět rozpoznat oprávněnou osobu jiným způsobem - např. dle tokenu
stráž musí provádět záznam o pohybu všech osob
problémem jsou zaměstnanci, se kterými byl nedávno rozvázán pracovní poměr a celková míra spolehlivosti stráží

náhradou nebo doplňkem stráží mohou být různé turnikety a přechodové komory vybavené zařízeními pro identifikaci osob nebo tokenů

Elektronická prostorová ochrana

- dveřní a okenní kontakty - detekují otevření
- otřesové hlásiče - detekují rozbití nebo proražení střežené plochy - skla, příčky, přepážky, …
- vodičové desky, drátěné sítě - slouží k detekci průrazů ve stěnách, podlahách apod.
- kontaktní matice - při instalaci pod podlahové krytiny slouží k detekci vstupu osob do chráněného prostoru
- Mikrovlnné, ultrazvukové, infračervené detektory - reagují na změnu rep. přerušení svazku příslušného záření
- zvukové hlásiče - reagují na specifické zvuky jako řezání, vrtání, šroubování, …
- kyvadlové hlásiče - reagují na otřesy a vychýlení z původní roviny

k ochraně jednotlivých předmětů lze použít obrazových vah, závěsů apod.
vhodným prostředkem v mnoha případech je průmyslová televize, zejména v kombinaci se záznamem snímaného obrazu

Detekce výstupu

vhodnou metodou je pokoušet se odhalit člověka odnášejícího část vybavení
prostředky jsou obdobné, jaké se používají v obchodních domech - různé nálepky či přívěsky, které lze snadno detekovat

Likvidace medií se senzitivními informacemi

je nutné mít k dispozici prostředky ke zničení nebo znehodnocení medií před jejich exportem z chráněného perimetru, nikdy není jasné, kam se dostanou

Zkartovače

existují v mnoha verzích pro nasazení v různých stupních zabezpečení, navzájem se liší jemností a způsobem provádění skartování
slouží především k ničení papírových dokumentů, dále disket pásek ze streamerů, kazet, barvicích pásek z úderových tiskáren

Přepisování magnetických medií

prosté smazání souboru většinou vede pouze k odstranění záznamu o jeho existenci, proto je nutné zajistit skutečné fyzické přepsání původních dat, pro větší stupeň bezpečnosti několikanásobné
metoda je zdlouhavá a ne úplně bezpečná

Degaussery

přístroj vygenerováním silného elektromagnetického pulsu dokáže zničit původní magnetické pole
ani v tomto případě nejde o zcela spolehlivou metodu vhodnou pro nasazení v nejvyšších stupních utajení

Odpovědnost za zabezpečení

celkovou odpovědnost má vedení organizace, lze ji rozdělit na odpovědnost za návrh bezpečnostní strategie a na odpovědnost za dodržování bezp. opatření
důležitou součástí bezpečnosti jsou opakované namátkové kontroly

Elektromagnetické vyzařování

… je způsobeno změnou proudu ve vodiči
problémem je vyzařování nejrůznějších částí počítačů, zejména monitorů a přenosových linek, důležité informace mohou unikat i naindukováním do napájecích obvodů zařízení
odposlech elmg. záření začasté není možné kriminalizovat a je nutno se vyrovnat s faktem, že jej útočník provádí
metody ochrany:

- Vzdálenost - intenzita vyzařování klesá se čtvercem vzdálenosti
- Zmatení - množství podobných signálů ztěžuje odposlech, je možno generovat záření podobných vlastností jako vyzařování výpočetního zařízení, nebo umístit více vyzařujících zařízení blízko sebe
- Speciální vybavení - je možné zakoupit součásti vyvinuté tak, aby jejich vyzařování nepřekračovalo jistou únosnou mez
- Vhodné umístění - alternativou nákupu nevyzařujících zařízení je umístění standardních zařízení v prostorách, které zamezují šíření záření - jde o různé schránky, skříně případně celé stíněné místnosti

Obnova provozu

při ztrátě či poškození celého, nebo části IS je nezbytné co nejdříve nahradit (je-li to možné) ztracené informace a služby a obnovit činnost v -pokud možno- původním rozsahu
není možné mít stále zálohy všech částí výpočetního systému, je třeba hledat kompromis mezi proveditelností záloh a jejich aktuálností a úplností
účinné zálohování musí být součástí globální bezpečnostní strategie, musí vycházet z celkového hodnocení bezpečnostní situace

kromě prvotní funkce zotavení z chyb poskytuje dobře navržený systém záloh též prostředky k vytvářená archivních kopií různých stadií zpracovávaného projektu (viz. zpráva konfigurací)
obecným řešením jsou „zálohy“

Uživatelé

v běžném případě nejsou zálohy pro chod systému potřebné, pouze zdržují, nejsou konstruktivní
většina uživatelů nechápe jejich důležitost, často jsou dlouhodobými zkušenostmi utvrzováni v přesvědčení, že havárie nenastávají

Mechanismus pro vytváření záloh

- musí být schopen provádět zálohy na co možná nejrůznější zařízení
- vlastní zálohy by měly být uloženy v nějakém standardním formátu
- je nezbytná verifikace vytvořené záložní kopie
- žádoucí je komprese ukládaných dat - přináší zrychlení a menší objem záložních dat
- důležitá je kryptografická ochrana záloh, aby mohly být ukládány mimo chráněné prostory
- zálohovací SW by měl používat mechanismus pro korekci chyb - vede ke zvýšení pravděpodobnosti, že zálohu se podaří přečíst a obnovit původní data
- mělo by být možno specifikovat, která data mají být (kdy) zálohována
- SW by měl vytvářet podrobný audit o své činnosti
- vlastní SW by měl být schopen běžet na co největším počtu HW platforem
- zálohovací SW by měl být před ostrým nasazením důkladně otestován, je vhodné znát názor uživatelů
- nejčastější použití záloh spočívá v přenesení a obnově dat na jiný funkční stroj
- úspěšný přístup k zálohám by měl být vázán na zadání hesla

Záložní media

výměnná media

- USB disky - snadno dostupné levné řešení, nevýhodnou mizivá kapacita
- pásky - dodávají se v kapacitách od stovek GB až jednotky TB, patrně nejrozšířenější záložní medium, nutno používat skutečně kvalitní materiál
- WORM disky – dnes diskutabilní pro dlouhodobé zálohy a pořizování archivních kopií, velmi dobrá cena za byte, relativně nejistá spolehlivost, malá kapacita

- hard copy - může být za jistých podmínek vhodná pojistka, jistě lepší, než nemít data vůbec ☺

Nevýměnná media

- další disk - důležitá data jsou kopírována na další disk, jednoduché, rychlé, nechrání před zničením celého počítače, nebo jeho odcizením atp.
- disk mirroring - zápisy na jednom disku jsou automaticky prováděny i na druhém disku - opět chrání pouze před chybami disku, výhodou je transparentnost a on-line zotavení z chyby
- duplexing - dva stroje si udržují přesně stejný obsah paměti a synchronně provádějí veškeré operace, při výpadku prvního stroje automaticky přebírá jeho funkci záložní počítač - velmi drahé řešení, opět plně transparentní
- síť - zálohování lze provádět kopírováním dat na jiný počítač zapojený v síti

Zálohy hardware

je nutné být schopen vyrovnat se dostatečně rychle s chybou technického vybavení - tedy buď mít k dispozici záložní systém, nebo alespoň kritické součástky, případně mít možnost vadnou součástku nahradit částí jiného stroje
řešením též smlouva o servisních zásazích s dodavatelem technologie
existují firmy, provádějící záchranu dat z havarovaných zařízení - tyto výkony však bývají drahé

Software

- ze všech medií s instalacemi nového sw. by měla být neprodleně pořízena alespoň jedna kopie
- originální instalace by měla být po celou dobu chráněna proti zápisu a ihned po pořízení kopií uložena na bezpečném místě
- vlastní instalace probíhá z pořízených kopií

tyto zásady ztrácejí naléhavost v souvislosti s rozšiřováním instalací na CDROM

Zásady pro pořizování záloh

pravidla pro pořizování záloh závisí na konkrétní situaci - jak často dochází k změnám dat, denní objem nových dat, důsledky případné ztráty dat, atd.

Data nebo programy

ztráta dat je vždy velmi problematické, znovupořízení dat může být obtížné, nebo dokonce nemožné
naproti tomu software je zpravidla možno znovu nainstalovat, což však zabere nějaký čas, navíc vytvoření dobré konfigurace může trvat velmi dlouho

konfigurační soubory by tedy měly být zálohovány společně s daty

Typy záloh

pokud máme k dispozici dostatek místa, je vhodné provádět zálohu komplet všech dat a programů
dříve častý model Grandfather-Father-Son - tři cyklicky používané archivní kopie, nejnovější kopie vždy vytvořena na mediích po nejstarší kopii, případně nejstarší kopie je uložena na bezpečném místě
alternativou je provádění kompletní zálohy vždy po určitém čase a v mezidobí pořizovat pouze zálohy změněných souborů
v současné době se zálohování soustřeďuje na objekty příslušné úrovně IS (OS, DB, aplikační server), systémy řízení zálohovacího mechanismu umožňují sofistikovaná schémata zálohování pracující s jednotlivými zálohovanými objekty namísto celých setů záloh
četnost těchto částečných záloh může být závislá na důležitosti zálohovaných dat

Uložení záložních kopií

význačné záložní kopie (např. záloha z konce týdne, záloha při ukončení fáze projektu, …) by měly být uloženy na bezpečném místě vzdáleném od místa, kde je instalován výpočetní systém
oddělené uložení kopií chrání proti následkům přírodních katastrof, zlodějům, teroristům apod.
přivezení kopií ze vzdáleného místa uložení poskytne personálu čas pro překonání prvotního stresu a provedení racionálního rozboru situace

místo uložení záložních kopií by mělo být zajištěno proti přírodním katastrofám, mělo by pro uložené materiály zajišťovat stabilní prostředí, je vhodné aby bylo chráněno proti vniknutí neoprávněných osob
každý vstup do tohoto zařízení musí být zaznamenám, stejně jako všechny zde prováděné operace

Plány obnovy (plány kontinuity – bus. continuity planning)

pro případ poruchy by měly být vypracovány podobné procedury, co je třeba učinit za účelem rychlého odstranění následků
tyto plány musí být důkladně prověřeny a otestovány, musí být stále k dispozici (nejlépe v tištěné podobě) pro případ poruchy

Obnova provozu

často může být životně důležité obnovit dostatečně rychle činnost výpočetního systému
většina výrobců počítačů je schopna v kritickém případě dodat během jediného dne náhradní technické vybavení stejné, nebo dostatečně podobné původnímu

- *cold site* je zařízení vybavené zdroji elektrické energie, klimatizací, komunikačními linkami atd., kde může být výpočetní systém velmi rychle nainstalován a uveden do provozu
- *hot site* je zařízení navíc vybavené též výpočetním systémem, který je připraven ke spuštění, stačí pouze přinést zálohu dat a programů, pomocí hot site lze obnovit provoz zcela zničeného výpočetního systému během několika hodin
- clustery – redundance na úrovni funkčních jednotek (serverů, systémů) zajišťující automatické přenesení výpočetních operací na zbylé kapacity
- mirroring – on-line redundance na úrovni datových úložišť
- zálohy

Programy ohrožující bezpečnost

v této části se budeme zabývat pouze aplikačními programy
programy mohou poškodit či zcela zničit data, způsobit kompromitaci utajovaných skutečností, omezit a popřípadě vyloučit funkčnost celého systému.

Útoky proti integritě a utajení dat

Trapdoors

pojmem označujeme nedokumentovaný vstup do programového modulu, nejčastěji bývá vytvořen v rámci tvorby tohoto modulu za účelem snazšího ladění:

- doplnění příkazu do množiny příkazů, které modul normálně vykonává. Doplněný příkaz např. aktivuje ladicí tisky apod.
- špatně ošetřené vstupy modulu, modul nerozezná nepřípustná vstupní data, ale nechová se a nich korektně
- někdy jsou trapdoors nutné pro provedení auditu systému

trapdoors jsou obvykle odstraněny před dokončením modulu, ale mohou být opomenuty, ponechány za účelem ladění dalších modulů či snazší správy dokončeného programu, nebo pro získání neoprávněného přístupu k běžícímu programu

Trojské koně (trojan horses)

program, který kromě svých "řádných" funkcí vykonává ještě další skryté akce

objevit Trojského koně v programu o stovkách tisíc či milionech řádků je velmi obtížné, navíc v tomto smyslu může být program upraven až následně po otestování a zařazení do provozu

Salámový útok (salami attack)

jde o programy, které se snaží využívat ve svůj prospěch malých zaokrouhlovacích chyb na hranici přesnosti počítače

…ve velkém množství

program, převádějící na programátorovo konto zaokrouhlovací chyby při výpočtu úroků

- program zajišťující bezhotovostní platby, který čas od času zvýší klientovi poplatky o malou sumu, kterou pak převede na vhodný účet
- upravený scheduler, spouštějící vybrané programy v čase běhu jiných

salámový útok je opět velmi těžko detekovatelný, neboť bývá prováděn v rozsáhlých SW systémech, navíc nepůsobí viditelné problémy

objevení často pouze náhodné

Skryté kanály (covert channels)

v prostředích spravujících klasifikované informace zpravidla aplikační programátoři po ukončení vývoje nemají přístup k běžícím programům
chtějí-li získat přístup ke zpravovaným informacím, vytvoří skrytý kanál
implementace je naprosto obecná:

- zdánlivé chyby na výpisech
- existence či neexistence specifických souborů
- vznik jistých systémových událostí
- nepatrné změny front-endu
- …

zvláště vhodné pro únik malého množství informace, opět prakticky nedetekovatelné

(Objevené) slabiny programů (exploits)

nekorektní zpracování vstupních dat může vést k pádu systému, případně k totální ztrátě kontroly nad systémem
existuje obrovské množství automatických nástrojů, které i laikovi umožňují využívat známe slabiny obvyklých programů

Útoky proti dosažitelnosti služeb systému

Hladové programy (greedy programs), DOS útoky

- na mnoha počítačích běží s velmi nízkou prioritou programy, které vykonávají nejrůznější zdlouhavé výpočty, které "nepospíchají"
  nechtěné či úmyslné zvýšení priority může znamenat zahlcení celého systému
- dalším případem procesy generující velké množství synovských procesů, mnohdy chybou programu

- programy běžící v nekonečné smyčce
  operační systémy často obsahují obranné mechanimy, které násilně ukončí programy, jež běží příliš dlouho
- v době provádění I/O operace neběží virtuální čas procesu, tedy procesy generující velmi velké množství I/O operací mohou běžet takřka neomezeně dlouho
- různé druhy síťových operací a aktivit

Viry

(relativně malý) program s autoreprodukční schopností

- zpravidla se připojí nebo nahradí část kódu napadeného programu, při spuštění tohoto programu se nejprve provede kód viru, který se nainstaluje do paměti a převezme nebo pozmění některé funkce systému
- viry často obsahují obranné mechanismy proti detekci, jsou schopny instanci od instance podstatným způsobem měnit vlastní kód, po nainstalování do paměti provedou operace, jež ostatním programům učiní použitou oblast paměti nedostupnou
- velmi často po určitou dobu pouze provádějí reprodukci bez jakýchkoliv vedlejších projevů - v době odhalení tak může být napadena drtivá většina programů
- šíření virů částečně omezuje nástup systémů poskytujících ochranu paměti a dokonalejší správu prostředků, tyto systémy však bývají komplikované a dokáží tak poskytnout úkryt mnohem komplexnějším virům
- následky virové nákazy mají nejrůznější podobu - od převážně neškodných zvukových či obrazových efektů, které pouze rozptylují obsluhu a zatěžují počítač až po rozsáhlá poškození či zničení veškerých dat a programů
- podmínkou šířená virů je neopatrná manipulace s programovým vybavením, přenášení většinou nelegálního software atp.
- dobrou obranou je kromě proškolení personálu též rozdělení veškerých programů a souvisejících dat do oddílů, které jsou navzájem dostatečně odděleny, tak aby nemohlo docházet k šíření virů z jednoho oddílu do druhého.

Červy (worms)

síťová obdoba virů, mají schopnost prostřednictvím komunikačních linek se šířit z jednoho počítače na druhý

- obecně vzato mají stejné zhoubné účinky jako viry, avšak díky schopnostem samovolně se šířit v dnes již celosvětovém měřítku je jejich expanze daleko rychlejší (řádově hodiny!) a dopad tedy daleko větší
- obranou je rovněž kvalitní správa programového vybavení, používáni pouze dobře otestovaných programů a rozdělení sítě na domény, mezi kterými dochází k minimálnímu sdílení informací, které je navíc podrobeno důkladné kontrole

Metody vývoje bezpečného programového vybavení

programátoři mají k dispozici mnoho prostředků jak překonat obranné mechanismy systému
při vývoji software je tedy nutné používat metody eliminující tyto snahy

Modularita, zapouzdření, ukrytí informací

- program má být rozdělen na malé navzájem nezávislé moduly
  → výhodou snadná udržovatelnost, srozumitelnost, znovupoužitelnost, opravitelnost, testovatelnost
- moduly mezi sebou mají minimum vazeb, veškeré interakce se odehrávají přes precizně definované a *zdokumentované* rozhraní
- stačí, aby všechny moduly měli definovaný vstup, výstup a funkci, je nežádoucí šířit způsob, *jak* modul svoji funkci provádí - nemůže tak docházet k úmyslným pozměňováním ostatních modulů

Nezávislé testování

je obtížné prokazovat správnost programu - to že nebyly nalezeny chyby může pouze znamenat, že byla použita nevhodná metodika testování
testování by měl provádět nezávislý tým - snižuje se nebezpečí, že výsledný program obsahuje nežádoucí kód, či že nebyl dodržen původní cíl projektu.

Správa verzí a konfigurací (Configuration management)

hlavním cílem je zajištění dostupnosti a používání správných verzí software
občas je potřeba vrátit se k verzi před provedením jistých úprav, tento problém je závažnější v prostředí, kde na projektu pracuje celý tým
⇒Správa konfigurací zajišťuje udržování integrity programů a dokumentace, veškeré změny jsou vyhodnocovány a zaznamenávány

⇒Správa konfigurací zabraňuje úmyslným změnám již odzkoušených programů (vkládání trapdoors, logických bomb, …)

Hlavní důvody pro zavedení správy konfigurací

1. Zabraňuje nechtěným ztrátám předchozích verzí software
2. Odstraňuje komplikace při vývoji několika podobných verzí (např. pro různé platformy) zároveň
3. Poskytuje mechanismus pro kontrolované sdílení modulů, z nichž je skládán vytvářený systém

Za účelem udržení přehledu je nutné vést důkladnou dokumentaci všech kopií, správou konfigurací se zpravidla zabývá vyčleněný pracovník.

Programátor pracuje na vlastním exempláři modulu, který po ukončení etapy jej předá správě konfigurací včetně popisu provedených úprav a celkové charakteristiky a dále již v něm nemůže činit změny.

Je vhodné, aby správce konfigurací přijímal programy výhradně ve zdrojové formě se soupisem a popisem provedených změn. Nutné vést detailní log co, kdo, kdy dělal.

Spolehlivý software

Program je *funkčně korektní* pokud vykonává správně všechny očekávané funkce a nic víc.
Spolehlivým software (trusted software) rozumíme programy o kterých věříme, že jsou funkčně korektní a že tuto korektnost vynucují i modulů, které samy spouštějí.

operační systém je zpravidla spolehlivý sw.

Vlastnosti spolehlivých programů

- Funkční korektnost - viz. výše
- Zajištění integrity - pgm. zachová korektnost používaných dat i v případě nesprávných příkazů nebo příkazů zadaných neoprávněnými uživateli
- Omezená práva - program mající přístup k utajovaným datům minimalizuje přístup k těmto datům, přístup nepostupuje dalším ne-spolehlivým programům
- Vhodná úroveň oprávnění - program byl zkoušen v souladu s mírou utajení dat, která spravuje a s ohledem na prostředí, ve kterém běží

Spolehlivý sw. zajišťuje přístup k citlivým datům pro (obecně nespolehlivé) uživatele, kterým není možné dát přímý přístup k prvotní reprezentaci dat. Dále zajišťuje provádění sensitivních operací.

Vzájemné podezřívání (Mutual suspicion)

ne všechny programy jsou spolehlivé, ani s pomocí OS není vždy možné spolehlivost vynutit
programy vytvořené podle konceptu vzájemného podezřívání pracují jako kdyby ostatní moduly byly vadné

- nevěří volajícím, že předávají korektní vstupy
- ověřují, že volaná rutina předala správná data
- s ostatními komunikují pouze prostřednictvím dobře chráněného rozhraní

Omezení (Confinement)

používají operační systémy proti podezřelým programům
podezřelý program má přísně vymezeno, jaké systémové zdroje smí používat (sandbox)
např. runas v MS Windows, chroot v unixových systémech

Parcelizace informací (Information compartement)

veškerá data a programy v systému jsou rozdělena do několika oblastí, každá informace leží právě v jedné oblasti, každý program může pracovat s daty z nejvýše jedné oblasti, do které sám patří

Access Log

systém musí zaznamenávat co, kdo, kdy, s čím a jak dlouho dělal
podle stupně utajení se volí množina zaznamenávaných aktivit, zejména je nutné zaznamenávat chyby (nepovolené přístupy, špatná hesla, …)

Administrativní nástroje ochrany

někdy může být vhodné ovlivňovat přímo lidský faktor, a ne až výsledný produkt

Standardy vývoje programů

není vhodné povolovat programátorům, aby pracovali zcela dle svého, je třeba mít na paměti, že výsledný kód musí být verifikovatelný, udržovatelný apod.

nejčastější administrativní kontroly vývoje software

Standardní návrh - obsahuje popis povolených vývojových prostředků, jazyků, metodologií
Standardy pro tvorbu dokumentace, stylu kódování, jazyka - popis, jak má výsledný kód vypadat, volby názvů proměnných, styl komentářů, …
Standardy programování - závazné popisy, jakým způsobem se provádí programátorská práce v globálním měřítku, rozpisy peer reviews, auditů programů apod.
Standardy testování - soupis používaných verifikačních metod, archivování výsledků testů …
Standardy konfiguračního managementu - způsoby výměny produktů, způsob zaznamenávání změn, …

standardy jsou důležité při týmové práci, zabraňují vzniku modulů, kterým rozumí pouze autor

Dodržování standardů vývoje programů

aby byly efektivní, musí být standardy důsledně dodržovány za všech okolností
typickými situacemi, kdy vznikají tendence je porušovat jsou okamžiky, kdy projekt nabírá zpoždění proti plánu, po odchodu klíčových pracovníků apod.
dodržování standardů by měli podporovat pravidelné audity bezpečnosti (security audits) prováděné nezávislým bezpečnostním týmem

Rozdělení úkolů

je vhodné práci rozdělit mezi více lidí, kteří se znají co možná nejméně
omezuje se tak nebezpečí vzniku bezpečnost ohrožujících programů, navíc pokud programátor očekává, že jeho kód bude podroben zkoumání nezávislého testovacího týmu, omezí své nekalé aktivity

Charakter přijímaných pracovníků

firmy běžně sestavují profily svých potenciálních pracovníků a přijímané podrobují všestrannému zkoumání, zda nebudou představovat hrozbu pro bezpečnost - obvyklé jsou reference z dřívějších působišť, psychologické testy apod.
po přijetí má pracovník povětšinou velmi omezený přístup k senzitivním informacím, až časem získává důvěru a tím i rozsáhlejší bezpečnostní oprávnění

Sledování pracovníků

je vhodné vést co nejpodrobnější informace o pracovnících, zejména o:

- vybraných zálibách (hraní, drogy, sex, jiné finančně náročné koníčky)
- neobvyklých změnách chování pracovníka
- nenadálých změnách majetkových poměrů

Verifikace a validace software

ověřuje, zda požadavky na SW jsou korektně implementovány a odpovídají systémové specifikaci, cílem je důkladně analyzovat a otestovat
v&v se provádí v průběhu a po dokončení díla s souladu s připraveným plánem, který by měl připravit tým tvořící systémovou specifikaci
provádění v&v má být prováděno nezávisle na tvorbě SW po stránce:

- technické – jiní lidé, kt. sami pochopí specifikaci, ověří užité vývojové prostředky a nejsou zatíženi znalostí průběhu návrhu řešení
- řídící – tým si musí sám zvolit co a jakým způsobem bude testovat, pouze odpovídá za sprvánost výsledků (jejichž obsah je dán v rámci plánu)
- finanční – tým musí být závislý pouze na včasnosti a správnosti dodání svých výsledků, nikoliv na výsledcích celého projektu

Řízení v&v

Pareto efekt: 20% chyb spotřebuje 80% nákladů na předělávky

zajišťuje plánování, koordinaci, včasnou identifikaci problémů, vyhodnocování výkonnosti a odhady dopadu změn návrhu, navrhuje vhodnou metriku pro hodnocení kvality sw, poskytuje zpětnou vazbu vývojářům
pro odstranění největších problémů slouží analýza hazardů a kritických sekcí

Aktivity v&v

- validace požadavků na sw – ověřit, zda požadavky nejsou v rozporu s platnými standardy, zda nejsou vnitřně sporné
- v&v návrhu sw – ověřit, že návrh úplně a bezchybně implementuje požadavky
- v&v kódu – že byly správně užity předepsané postupy a standardy vývoje
- testování – (t. modulů, t. integrace, t. systému, t. instalace) na jednotlivých úrovních se zkouší, zda výsledek se chová přesně dle zadání, zejména pod tlakem
- v&v při správě a používání sw – co dělat při změně v systému,

Major Software V&V Activities

Aktivita/fáze	Úkoly
Řízení V&V	Plánování
	Monitoring
	Vyhodnocení výsledků, vliv změn
	Reporting
Software Requirements V&V	Kontrola dokumentace konceptů
	Analýza trasovatelnosti
	Vyhodnocení požadavků na software
	Analýza rozhraní
	Iniciální plánování systémových testů software
	Reporting
Software Design V&V	Analýza trasovatelnosti
	Vyhodnocení návrhu software
	Analýza rozhraní
	Iniciální plánování modulových testů
	Iniciální plánování integračních testů
	Reporting
Code V&V	Analýza trasovatelnosti
	Vyhodnocení kódu
	Analýza rozhraní
	Dokončení přípravy modulových testů
	Reporting
Unit Test	Provedení modulových testů
	Reporting
Software Integration Test	Dokončení přípravy integračních testů
	Provedení integračních testů
	Reporting
Software System Test	Dokončení přípravy systémových testů
	Provedení systémových testů
	Reporting
Software Installation Test	Auditování koordinace při instalaci
	Reporting
Software Operation and maintenance V&V	Analýza vlivu změny
	Opakování řízení V&V
	Opakování technických aktivit V&V

v&v techniky

statické – přímo zkoumají struktury a formu produktu bez jeho spuštění (reviews, inspekce, data-flow)
dynamické – analýza výsledků zkušebních běhů a simulací (testování, prototypování)
formální – matematická analýza zadání a funkčnosti (VDM, Z)

Analýza algoritmů *(Algorithm analysis)* – zpětným přepisem do běžného jazyka nebo formalismu je ověřována logika a správnost návrhu, zahrnuje znovuodvození vztahů, vhodnost návrhu, stabilitu, časování, přetypování atd. *Cíle: přesnost; efektivita; správnost;časování; prostorová náročnost.*

Analytické modelování *(Analytic modelling)* – vyhodnocení výkonu a plánování zdrojů. *Cíle: přesnost; efektivita; úzká místa; proveditelnost; predikce výkonu.*

Back-to-back testing porovnávání výstupu více programů se stejnou specifikací. *Cíle: anomality, rozdíly mezi verzemi.*

Analýza mezních hodnot *(Boundary value analysis)* –detekování chyb na mezních vstupech. (nuly, prázdná pole a řetězce, první a poslední položka…). *Cíle: analýza algoritmů; velikost polí; inkonzistence limitů; chyby specifikace.*

Čtení kódu *(Code reading)* – expertní pročítání cizího kódu. *Cíle: korektnost; užívání proměnných; nevolané funkce; testy parametrů; styl; redundance.*

Analýza toku řízení *(Control flow analysis)* na grafické reprezentaci se studuje předávání řízení běhu, hierarchie volání rutin, dosažitelnost stavů. *Cíle: úzká místa; testy hranic; struktura modulů; korektnost; vyhodnocení návrhu; souborové operace;vyhodnocení formální specifikace; tok informací a konzistence; interakční testy; invarinty cyklů; efektivita; predikce výkonu.*

Analýza pokrytí *(Coverage analysis)* určí míru otestování systému danou sadou testů. *Cíle: integrační testy, systémové testy.*

Kritická analýza *(Critical timing/flow analysis)* ověřuje dodržení časových nároků v návrhu. *Cíle: modelování; synchronizace; časování.*

Databázová analýza *(Database analysis)* ověřuje, že DB schéma a procedury odpovídají logickému návrhu. Integrita dat. *Cíle: ochrana přístupu; typování; vyhodnocování návrhu; souborové operace; tok informací; efektivita; prostorové nároky;testy modulů.*

Analýza toku dat *(Data flow analysis*) při návrhu globální struktury aplikace. Čtení proměnných až po inicializaci, opakované zápisy bez čtení, … *Cíle: úzká místa; testy mezí; hierarchie modulů; reakce na okolí; šíření chyb; souborové operace; tok informací; invarinty cyklů; efektivita; vyhodnocování návrhu; integrační testy; predikce výkonu; neinicializované hodnoty; odkazy.*

Rozhodovací tabulky *(Decision (truth) tables*) analýza komplexních logických vztahů a rozhodování. *Cíle: logické chyby.*

Desk checking – hledání zjevných chyb v kódu. *Cíle: volání neexistujících funkcí; meze polí; neshoda s návrhem; práce s registry; špatné linkování; nekonečné cykly; vadná inicializace; obrácené predikáty; neshoda parametrů; nekonečná rekurze; nedeklarované proměnné; nedosažitelný kód.*

Error seeding podsouváním známých chyb do programu zjišťuje, zda navržené testy jsou adekvátní

$$\frac{\text{Nalezených vsunutých chyb}}{\text{Celkem vsunutých chyb}} = \frac{\text{Nalezených skutečných chyb}}{\text{Celkem skutečných chyb}}$$

Cíle: adekvátnost testů.

Event tree analysis analýza zdola nahoru, sledování jak se rozšiřuje vliv události na systém. *Cíle: analýza hazardů; bezpečnost; analýza hrozeb; časování.*

Konečné automaty *(Finite state machines* FSM) modeluje system konečným automatem, popisuje reakce na vstupy *Cíle: úplnost specifikace; nekonzistentní požadavky; modelování.*

Funkční testování *(Functional testing*) spuštěním systému se zkoumá, zda jsou naplněny požadavky uživatelů. *Cíle: testy mezí; souborové operace; provozní charakteristika; pokrytí specifikace; predikce výkonu; systémové testy; test modulů; neinicializované hodnoty; odkazy; trasování proměnných.*

Inspekce *(Inspections)* skupina expertů nezávisle analyzuje kód, následně uspořádají monitorované sezení, kde jsou identifikovány a zaznamenány nalezené nedostatky. Následně se provádí kontrola jejich odstranění. *Cíle: přesnost; naplnění požadavků; vyhodnocení formální specifikace; konzistence toku informací; logické chyby; invarinty cyklů; prostorové nároky; syntaktické chyby; neinicializované hodnoty.*

Analýza rozhraní *(Interface analysis*) statické ověření, že rozhraní modulů jsou správná, reagují na všechny možné vstupy, jsou správně volány, dodržují dané normy atd. *Cíle: nesoulad parametrů; použití globálních proměnných; nesprávné užití statických a dynamických dat; volání nesprávných funkcí; chyby popisu I/O.*

Testování rozhraní *(Interface testing*) dynamická obdoba předchozí metody. *Cíle: nesoulad parametrů; použití globálních proměnných; nesprávné užití statických a dynamických dat; volání nesprávných funkcí; chyby popisu I/O.*

Analýza mutací *(Mutation analysis*) ověřuje vhodnost testů. Vnesením záměrných chyb je vytvořena řada mutací původního systému a na každou z nich se aplikují testy. Srovnávají se výsledky. *Cíle: testy mezí; retestování po změně; příprava testů.*

Testování výkonu *(Performance testing)* porovnává skutečný výkon s pořadavky, zatížení komponent, … *Cíle: alokace; synchronizace; časování.*

Petriho sítě *(Petri-nets*) ověřuje odolnost návrhu proti deadlockům, proveditelnost, dosažitelnost. Systém je modelován za použití stavů, událostí, přechodů, vstupů a výstupů. Je možné simulovat běh systému za různých podmínek. *Cíle: analýza hazardů; modeling; bezpečnost; analýza hrozeb; časování.*

Důkaz korektnosti *(Proof of correctness* - formální verifikace) matematickými metodami se modeluje systém a formálně se dokáže splnění nakladených předpokladů a tvrzení o funkčnosti. *Cíle: korektnost; ověření kritických sekcí.*

Prototypování *(Prototyping*) zkoumání pravděpodobných výsledků implementace - identifikace nekompletních a nepsrávných požadavků, ověřování vhodnosti návrhu. *Cíle: chování; nepokryté požadavky na funkcionalitu; úplnost specifikace; uživ. rozhraní.*

Regresní analýza a testy *(Regression analysis and testing*) znovuvyhodnocení naplnění (původních) požadavků po větších změnách. *Cíle: integrační testy; retestování po změně; systémové testy; testy modulů.*

Procházení požadavků *(Requirements parsing)* zkoumání, zda všechny požadavky jsou specifikovány jednoznačně a úplně *Cíle: přesnost; checklisty; kompletnost; 8zkonzistence; proveditelnost; vyhodnocení formální specifikace; konzistence toku informací; integrační testy; korektnost; vyhodnocení naplnění požadavků; retestování po změně; pokrytí specifikace; systémové testy; testy modulů.*

Reviews kontroly naplnění požadavků, návrhu software, často jako předpoklad provedení dané vývojové aktivity *Cíle: před valstními testy, logické chyby; syntaktické chyby.*

Sensitivity analysis je predikcí pravděpodobnosti, s jakou testování odhalí chyby. Umožňuje porovnávat různé testovací strategie a odhadovat, které oblasti kódu budou dotknuty změnou v rámci údržby. Lze použít na hodnocení odolnosti vůči chybám. *Cíle: korektnost; logické chyby; reliability; adekvátnost testů.*

Simulation – využívá se na zkoumání interakcí velkých komplexnícxh systémů. Využívá spustitelný model a zkoumá chování systému a reakci na práci administrátora. *Cíle: chování; testy mezí; testování verzí; sampling, support; proveditelnost; souborové operace; path testing; provozní charakteristika; retestování po změně; pokrytí specifikace; predikce výkonu; systémové testy; neinicializované hodnoty; odkazy; kontroly/trasování proměnných.*

Sizing a analýza časování – určování, zda alokace SW a HW jsou dostatečné pro daný návrh. *Cíle: efektivita algoritmů; úzká místa; testy mezí; testování verzí; integrační testy; efektivita; provozní charakteristika; retestování po změně; prostorové nároky; systémové testy; časování;testy modulů.*

Slicing je technika dekompozice programů, kdy výstupní hodnota je trasována zpětně programem za účelem identifikace všech relevantních součástí kódu. Užitečné pro demonstraci funkční diveristy. *Cíle: alokace V&V zdrojů; společný kód; konzistence toku informací; dekompozice programu; odkazy.*

Chybový mód, efekty, kritická analýza *(Software failure mode, effects and criticality analysis)* odhaluje nedostatečné naplnění požadavků zkoumáním důsledku chyby modulu (vč. vadných instrukcí) v závislosti na typu chyby.

Výsledkem je matice dopadů různých chyb na systém popisující rozsah, kritičnost, potřebné změny, preventivní opatření. *Cíle: analýza hazardů; bezpečnost; úplnost specifikace; analýza hrozeb.*

Analýza chybových stromů *(Software fault tree analysis)* určuje a analyzuje požadavky na bezpečnost. Užívá se k určení dopadů možných rizik. Cílem ověřit, že SW nepřivede systém do nebezpečného stavu a za jakých podmínek případně hrozí nebezpečí. Analytik modeluje situaci, ze hrozba byla realizována a zkoumá, jaké okolnosti ji mohly způsobit. *Cíle: analýza hazardů; bezpečnost; analýza hrozeb.*

Stress testing zkoumá odezvu systému až do extrémních podmínek, aby se dohalily slabiny a demonstrovala shopnost zvládat normální zátěž. *Cíle: chyby návrhu; plánování nastavení prosystém při přetížení.*

Structurální testování posuzuje logiku modulů, může být použito pro ověření naplnění požadavků pro testy pokrytí, tj. jak moc byl program „prohnán". *Cíle: úzká místa; šíření chyb; kontrola parametrů; provozní charakteristika; retestování po změně.*

Symbolické spouštění *(symbolic execution)* ověřuje shodu mezi kódem a specifikací. Spuštění programu je simulováno nad symboly namísto dat, výstup je vyjádřen jako výraz nad těmito symboly. *Cíle: testování tvrzení; provozní charakteristika; důkaz korektnosti; retestování po změně.*

Certifikace testů zajišťuje, že výsledky testů odpovídají skutečným nálezům. Použité nástroje, media a dokumentace jsou certifikovány pro zajištění udržovatelnosti a opakovatelnosti. Ověření, že dodaný produkt je identický s objektem V&V. *Cíle: dodávka nesprávné verze produktu; nesprávné výsledky; vynechané zprávy a testy.*

Procházky *(Walkthroughs)* podobné reviews, méně formální, více detailní. Autor vede skupinu expertů návrhem, nebo kódem modulu, ostatní komentují použitou techniku, styl, hledají chyby, porušení standardů, ptají se na detaily. *Cíle: checklisty; šíření chyb; vyhodnocení formální specifikace; go-no-go rozhodnutí; logické chyby; manuálí simulace; kontrola parametrů; retestování po změně; malé ale složité nebo k chybám náchylné sekce; kontroly stavu; syntaktické chyby; systémové testy; technické kontroly.*

V&V pro znovupoužitý SW

Většina V&V technik je vhodná pro znovupužitý SW, následující specifické techniky je vhodně doplňují

Analýza konzistence porovnává požadavky na veškerý stávající SW s novou specifikací pro zajištění konzistence. *Cíle: konzistence.*

Analýza rozhraní *(Interface analysis)* (viz analýza rozhraní a testování rozhraní výše) mimořádně důležité pro zkoušky rozhraní znovupoužitých modulů pro zajištní korektnosti adaptace a zjištění případných rozdílů.

Techniky specifické pro báze znalostí (KBS)

Alternativní model srovnává doménový model implementovaný daným KBS s alternativním modelem na úplnost a přesnost.
Control groups mohou být použity v rámci testování pro porovnáví výkonu při zpracování úlohy s/bez KBS
Analýza kredibility porovnává výsledky systému s odpověďmi experta.
Field testing – KBS se reálně používá, zaznamenávají s vyhodnocují se výsledky.
Testování nepovolených atributů - kontrola pravidel na omezení nepovolených hodnot atributů.
Logická verifikace jde o ověření kompletnosti expertní znalosti a konzistence v době vytváření doménového modelu.
Meta modely - porovnávání znalostí a pravidel s metamodelem domény.
Partition testing - vybírá příklady pro různé oblasti vstupních a výstupních hodnot a ověřuje, že specifikace pokrývá tyto případy
Verifikace pravidel zkoumá se kompletnost, vnořená/reduntantní pravidla, nekonzistentní pravidla, pravidla, která nemohou být splněna, cyklická pravidla, nedosažitelné výsledky, apod.
Statistická validace zkoumá, jak často používá pravidla nebo skupiny pravidel v bázi znalostí.
Turingův test - naslepo porovnává výkon systému s výkonem experta.
Weight analysis porovnává statistické informace spojené s pravidlem se statickou znalostí o dané doméně.

Bezpečnost v operačních systémech

Něco z historie

v počátcích výpočetní techniky byly používány kompaktní kódy přímo ovládající jednotlivá zařízení výpočetního systému
prvními předchůdci dnešních OS byly tzv. *exekutivy* (executives) - jednoduché linkery, loadery, …
zlom představoval nástup *monitorů* - prvních multiprogramových systémů, sami řídí přidělování prostředků systému, nastupuje problém vzájemné ochrany uživatel

Chráněné objekty

1. paměť
2. procesor
3. spustitelné programy
4. sdílená zařízení typu disky
5. seriově znovupoužitelná zařízení - tiskárny, pásky
6. sdílená data

Poskytované služby

- autentizace
- autorizace
- řízení přístupu/logická separace
- auditovatelnost
- dostupnost
- zotavení

Metody ochrany objektů v operačních systémech

- fyzická separace - procesy pro vykonávání operací různého stupně utajení používají oddělená zařízení
- časová separace - procesy různého stupně utajené jsou prováděny v různém čase
- **logická separace** - operační systém zajišťuje oddělení jednotlivých procesů tak, že pro každý vytváří iluzi, že má celý počítač pro sebe
- kryptografická separace - použitím kryptografických metod procesy provádějí ukrytí svých dat (např. sdílení komunikačních linek)

samozřejmě je možná kombinace několika metod separace
metody jsou seřazeny dle rostoucí (implementační) složitosti a zároveň dle klesající spolehlivosti

Ochrana paměti a adresování

Ochrana paměti je základním požadavkem pro zajištění bezpečnosti, má-li být spolehlivá, je nutná hardwarová podpora.
HW podpora navíc poskytuje dostatečnou efektivitu ochrany.

Ohrada (fence)

stanoví se hranice, operační paměť na jednu stranu od této hranice používá OS, na druhou stranu aplikační programy
metoda je vhodná pro jednoduché jednouživatelské systémy, umožňuje však pouze chránit OS, nemůže být použita pro vzájemnou ochranu uživatelů většího systému
implementace velmi jednoduchá:

- stroj má pevně zabudovanou tuto hranici
- stroj má tzv. *fence register*, jehož hodnotu porovnává s každou adresou, kterou aplikační program vygeneruje

Relokace

programy jsou vytvořeny tak, jako by v paměti ležely od adresy 0, v rámci procesu spouštění programu je ke všem odkazům v programu připočten relokační faktor
aplikace tak nemůže zasahovat do oblastí, v nichž leží systém
metoda má stejné nevýhody, jako předchozí způsob ochrany paměti

Base/Bound registry

… přenesení myšlenek předchozích metod do prostředí multiuživatelských systémů
k adresám generovaným programem je připočítávána hodnota bázového registru, každý odkaz je oprovnáván s hodnotou bound registru, zda je menší
program má tak shora i zdola omezen prostor, v němž může pracovat
metoda umožňuje vzájemné oddělení jednotlivých uživatelů, nechrání však kód aplikačního programu před chybou
možným rozšířením je používat dva páry registrů, jeden pro vymezení oblasti pro kód procesu, druhý pro datovou zónu
nevýhodou je nemožnost selektivního sdílení pouze některých dat

Značkovaná (Tagged) architektura

s každou adresou (slovem) v paměti stroje je spojeno několik tag bitů, jejichž obsah určuje typ zde uložených dat a povolené operace
obsah tag bitů je testován při každém přístupu k obsahu této adresy, tag bity mohou být měněny pouze privilegovanými instrukcemi
alternativou může být používání jednoho tagu pro celý blok slov

Segmentace

celý program sestává z několika bloků - segmentů - které mohou být nezávisle uloženy do paměti
program potom generuje odkazy ve tvaru <*jméno_segmentu*><*ofset*>
jméno_segmentu je pomocí systémem udržovaného segmentového adresáře převedeno na adresu počátku segmentu, ke které je přičten ofset
často je též ofset porovnán s velikostí segmentu, aby se zajistilo, že program nesahá "za segment"
Metoda již poskytuje dostatečné prostředky pro sdílení dat, navíc umožňuje ochranu kódu programu a případně i vybraných dat. Rovněž je schopna chránit uživatele navzájem.

Stránkování

metoda velmi podobná segmentaci, jen předpokládáme segmenty konstantní velikosti = *stránky*
opět dvousložkové adresování <číslo_stránky><offset> , ochrana proti adresování za stránku je vyřešena samovolně tím, že nezle udělat větší ofset, než je velikost stránky
možnost ochrany obsahu stránek je poněkud slabší než v případě segmentů, neboť není příliš jasná vzájemná souvislost obsahů stránek a rozdělení programu a dat do stránek

Ochrana procesoru

- privilegované instrukce – přístup ke klíčovým operacím (ovládání periferií a dalších autonomních zařízení, nastavení bezpečnostního mechanismu, přepínání úrovní, správa procesů, ovládání přerušení, ...)
- úrovně oprávnění procesu – rozdělení běžících procesů do různých tříd s diferencovanými oprávněními, přístupy k privilegovaným instrukcím, omezení přístupu k paměti, ...
- správa procesorového času

- správa využívání systémových prostředků – využití autonomních subsystémů, alokace, ...

Ochrana obecných objektů

s rozvojem multi-užívání vzrůstá škála objektů, které je třeba chránit

- soubory a data na záznamových zařízeních
- běžící programy
- adresáře souborů
- hardwarová zařízení
- různé datové struktury (stack,…)
- interní tabulky OS
- hesla a autentizační mechanismy
- vlastní ochranný mechanismus

na rozdíl od problému ochrany paměti, zde nemusí existovat centrální arbitr, přes kterého jsou směrovány všechny přístupy, navíc typů přístupů může být celá řada

cíle ochrany objektů
Kontrolovat každý přístup - subjekt může pozbýt přístupová práva a tedy je nutno mu zabránit v dalším používání objektu
Povolení co nejmenších práv - subjekt by měl mít pouze nejmenší možná oprávnění nutná ke korektnímu plnění jeho úkolu a to i v případě, že případná další práva by pro něj byla bezcenná - toto uspořádání snižuje možnost průniku v případě selhání části ochranného mechanismu
Ověření přijatelného používání - někdy je daleko podstatnější než přidělení či odepření přístupu moci kontrolovat, co subjekt s daným objektem provádí

Autentizace subjektů

shora popsané bezpečnostní mechanismy odvíjejí svoji činnost od informace, ***kdo*** žádá jaký přístup
je tedy nutné mít mechanismus pro autentizaci subjektů

Hesla

jednoduché myšlenkově i implementačně, ověřuje se platnost páru
<identifikace:heslo>
mechanismus přijetí této dvojice by neměl poskytovat útočníkovi zbytečné informace o systému

- je vhodné, aby vyhodnocoval až korektnost zadání celé dvojice

- někdy je proces autentizace záměrně pomalý - sekundy až desítky sekund - což znemožňuje hádání hesel
- neměl by podávat informace o příčině chyby,
- neměl by umožňovat neomezené zkoušení

přihlašování do systému lze navíc omezit na určitý čas či místo, jistý omezený počet současných přihlášení daného uživatele do systému

Hledání hesel

Systém musí mít možnost kontrolovat korektnost zadaného hesla, tedy je nutné aby udržoval informace o všech heslech

Textové soubory hesel

soubor obsahuje v textové podobě dvojice <identifikace:heslo>
velmi nevhodné, hesla lze zjistit z odcizených záloh, z dumpů paměti při pádech systému, dojde-li chybou některé komponenty systému v vyzrazení souboru hesel

Zašifrované soubory hesel

pro šifrování je možné použít konvenčních šifer, nebo lépe kryptografických hašovacích funkcí
soubory zašifrovaných hesel mohou být volně přístupné
protože by v případě dvou uživatelů se stejným heslem vyšla stejná šifra, je vhodné před šifrováním k heslu přidat náhodný řetězec (salt), salt je uchováván zároveň se zašifrovaným heslem a při verifikaci vždy přidán k zadanému heslu

One time passwords

řeší problémy úschovy hesel
namísto konstantní fráze má uživatel přiřazenu konstantní funkci (vhodný matematický výpočet, dešifrování soukromým k.líčem, …)
v procesu autentizace obdrží od systému náhodně zvolené vstupní parametry a odpoví výsledkem
metoda obzvláště vhodná pro vzájemnou autentizaci strojů

Návrh bezpečných operačních systémů

na kvalitě operačního systému závisí bezpečnost celého mechanismu ochrany dat
OS kontroluje chování uživatelů a programů a v konečném důsledku zpřístupňuje utajované informace
proces vývoje bezpečného OS lze rozdělit do několika fází

- *bezpečnostní modely* - vytvoří se formální modely prostředí a zkoumají se způsoby, jak v tomto prostředí zajistit bezpečnost
- *návrh* - po zvolení vhodného modelu je hledán vhodný způsob implementace
- *ověřování* - je třeba ověřit, že navržená implementace skutečně odpovídá teoretickému modelu
- *implementace* - praktické a důkladné provedení shora uvedených teoretických úvah

Návrh bezpečného operačního systému

implementace bezpečnostních mechanismů je v přímém rozporu s efektivitou systému

OS vykonává několik s bezpečnostní úzce souvisejících činností:

- autentizace uživatelů
- ochrana paměti - mezi uživateli i v rámci jednoho uživatelského prostoru
- řízení přístupu k souborům a I/O zařízením - ochrana před neautorizovaným přístupem
- alokace a řízení přístupu k obecným objektům - zajištění bezproblémového současného přístupu více uživatelů k stejnému objektu
- zabezpečení sdílení - zejména zajištění integrity a konzistentnosti
- zajištění spravedlivého přístupu - o HW prostředky se opírající mechanismus zajišťující, že všichni uživatelé dostávají přidělen procesor a ostatní systémové zdroje
- meziprocesová komunikace a synchronizace - systém poskytuje mechanismus pro bezpečné předávání zpráv mezi procesy, procesy nekomunikují přímo ale via systém

Bezpečnost musí být brána v potaz ve všech aspektech návrhu systému a musí být zapracována již v prvotním návrhu. Je velmi obtížné ji "přidat" do hotového návrhu.

Následující principy je vhodné mít na paměti:

Nejmenší práva - každý subjekt by měl mít pouze nezbytná práva

Ekonomický návrh - bezpečnosní systém má být malý a jednoduchý, pak je testovatelný a věrohodný

Otevřený návrh - bezpečnostní mechanismus by měl být veřejně známý (a oponovaný) a měl by záviset na bezpečnosti co nejméně objektů

Úplné zprostředkování - veškeré přístupy k objektům zprostředkovává a testuje OS

Povolování operace - co není výslovně povoleno, je zakázáno

Rozdělené oprávnění - přístup k objektům by měl záviset na více podmínkách (např. správná autentizace a vlastnictví klíče)

Nejmenší sdílené prostředky - sdílené moduly jsou potenciálním kanálem pro únik informací, mělo by jich být co nejméně

Snadná použitelnost - mechanismus není obcházen, když se neplete více, než je únosné

Virtuální adresní prostor

IBM MVS provádí logické oddělení uživatelů, které poskytuje dojem fyzické separace

pomocí mechanismu stránkování jsou cela odděleny adresní prostory jednotlivých uživatelů, každý uživatel vidí pouze svůj prostor, do každého z uživatelských prostorů je mappvána oblast paměti obsahující vlastní systém, čímž vzniká dojem, že uživatel má celý systém sám pro sebe

Virtual machine

operační systém IBM VM poskytuje nejen virtuální pamět, ale provádí virtualizaci celého počítače - I/O zařízení, file-systému a dalších zdrojů

simulovaný stroj může mít naprosto odlišné vlastnosti od vlastností počítače, na kterém OS VM běží

poskytovaná ochrana je tedy daleko silnější

VM byl navrhován jako systém, umožňující na jednom fyzickém počítači provozovat zároveň několik různých operačních systémů

tak představuje další vrstvu ochrany, neboť pokud se uživateli podaří proniknout ochrannými mechanismy operačního systému, který používá, získá přístup pouze k této jediné doméně, neboť VM mu zabrání v přístupu k celému počítači

Kernel

… část OS provádějící nejzákladnější funkce - standardně synchronizace, meziprocesová komunikace, zasílání zpráv a obsluha přerušení

tzv. *security kernel* poskytuje základ pro vybudování bezpečnostního mechanismu, často bývá implementován uvnitř kernelu

uzavřít bezpečnostní funkce systému do security kernelu má několik důvodů:

- oddělení od zbytku systému zjednodušuje ochranu mechanismu
- všechny bezpečnostní funkce jsou shromážděny v jednom kusu kódu, tedy implementace bezpečnosti je kompaktní

- kernel nebývá velký, tedy implementace je snadno ověřitelná
- je snazší provádět testování a změny bepečnostního mechanismu
- přes kernel procházejí veškeré žádosti o přístup ke všem objektům (volání odpovídajících modulů), tedy je možno zachytit každý přístup

kernel hlídá zejména:

- aktivaci procesů - zajišťování context switchingu, realokací paměti, access kontrol listů, …
- střídání domén - procesy často provádějí volání procesů běžících v jiné bezpečnostní doméně, za účelem získání senzitivních informací
- ochrana paměti - je nutné hlídat všechny odkazy na paměť, aby nedocházelo k narušení bezpečnostních domén
- I/O operace

Vrstvový model (Layered design)

již operační systémy s kernelem obsahují několik vrstev - hardware, kernel, zbytek OS, uživ. procesy

tyto vrstvy lze dále dělit - např. na uživatelské úrovni můžeme oddělit semi-sytémové programy jako různé databázové systémy, shelly apod.

vrstvy le chápat jako soustředné kruhy, čím blíže je vrstva středu, tím je důvěryhodější a bezpečnější

ne všechny bezpečnostní funkce (např. autentizace uživatele) jsou implementovány uvnitř bezpečnostního jádra

bezpečnostní jádro spolupracuje s okolními spolehlivými vrstvami, které by měli být formálně ověřeny a přinejmenším dobře otestovány

každá vrstva používá služby nižších vrstev a sama vyšším vrstvám poskytuje služby jisté úrovně bezpečnosti, stejna funkce může být implementována v několika vrstvách zároveň

Kruhová struktura (Ring structured)

kruhy číslovány od 0 (kernel), čím důvěryhodnější proces, tím nižší číslo kruhu, do kterého patří

kruhy jsou soustředné a překrývající se - proces patří do kruhu k a všech dalších, ve středu je HW počítače

každá procedura, nebo oblast obsahující data se nazývá *segment*

ochrana segmentu založena na trojici $\langle b_1, b_2, b_3 \rangle$, $b_1 \leq b_2 < b_3$, nazývané závora kruhu (ring bracket), (b_1, b_2) nazýváme přístupová závora (access bracket), (b_2, b_3) potom závora volání (gate extension, call bracket)

nechť programová rutina patří do kruhu k, pokud $k = b_1$, může pracovat přímo s daty tohoto segmentu, pokud $b_1 < k \leq b_2$, může pracovat přímo s kopií dat a pokud $b_2 < k \leq b_3$, může k datům přistupovat pouze prostřednictvím definovaného rozhraní (gate)

tento základní mechanismus, nazývaný též nondiscretionary nebo mandatory control může být dále doplněn o další doplňkové (discretionary) mechanismy - např. k daným datům smějí přistupovat pouze jmenovité procesy, procesy patřící do okruhu přístupové závory mohou volně číst, ale zapisovat pouze za specifických podmínek apod.

Průniky operačním systémem

1. místem největšího počtu průniků je mechanismus zpracování I/O operací
   - mnohá I/O zařízení jsou do značné míry inteligentní a nezávislá na zbytku systému, provádějí optimalizaci své činnosti, jejich řadiče často spravují více takovýchto zařízení
   - kód I/O operací je často velmi rozsáhlý, je těžké jej řádně testovat, někdy je dokonce nutné používat kód dodaný výrobcem zařízení …
   - v zájmu rychlosti a efektivity I/O operace občas obcházejí bezpečnostní mechanismy operačního systému, jako stránkování, segmentaci apod.
   - velká část I/O operací je znakově orientovaná, v zájmu efektivity se často příslušné kontroly neprovádějí s každým přijatým znakem, ale pouze při startu operace
2. HW současných procesorů poskytuje rozsáhlé prostředky pro vytvoření kvalitní bezpečnosti (úrovně oprávnění, privilegované instrukce, trasovací režimy, systém obsluhy výjimek, systém ochrany segmentů a stránek, ...), ne vždy je však využíván
3. dalším problémem je hledání kompromisu mezi důkladnou izolací uživatelů a nutností umožnit sdílení dat, tento kompromis zhusta bývá obtížně formalizovatelný, nejasnosti návrhu pak mohou být příčinou “děr” v implementaci
4. ne vždy je možné provádět kontroly oprávněnosti s každou operací, často je kontrola prováděna pouze jednou během provádění celého bloku akcí, pokud se v této době uživateli podaří změnit parametry, může dojít k průniku
5. další skulinu v bezpečnosti může způsobit snaha o obecnost možného nasazení systému - aby bylo možno systém používat pro nejrůznější úkoly, ponechají návrháři často mechanismus, pomocí kterého si uživatel může systém přizpůsobit
   tento mechanismus ovšem může být zneužit

6. klasickým místem průniku se stávají programy, běžící v rámci OS s rozsáhlými oprávněními
7. vzhledem k současné "prosíťovanosti" světa je častým místem útoku právě obsluha komunikačního protokolu

Bezpečnost v databázích

bezpečnostní problémy, se kterými se potýkají databázové systémy jsou obdobné, jako problémy operačních systémů

- fyzická integrita databáze - odolnost proti výpadkům napájení, zotavení z poškození
- logická integrita databáze - musí být zachována struktura dat a vzájemné vazby
- elementární integrita - data obsažená v jednotlivých položkách jsou korektní
- auditabilita - kdo a jak přistupoval k položkám v databázi
- kontrola přístupu - kdo co může s čím dělat
- autentizace uživatelů - každý, komu je povolen přístup musí být pozitivně identifikován
- dostupnost - uživatelé mohou přistupovat k datům tak, jak jsou oprávněni

Integrita báze dat

správce báze musí zajistit, že změny dat mohou provádět pouze oprávnění uživatelé
systém musí obsahovat prostředky překonávající nedostupnost položky nebo dokonce celé bázez hlediska OS a správce systému jde o ochranu relevantních souborů a programů, zálohy, kontroly zařízení atp.
z pohledu SŘBD přistupuje systém transakcí a logů, umožňující rekonstruovat stav databáze

Elementární integrita

autorizovaní uživatelé mohou vkládat data, ale činí chyby - ty musí SŘBD zachycovat a vyžádat si opravu
metody:

- field checks - test vhodnosti vkládaných dat: zda je to číslo, zda jde o jméno, …
- kontrola přístupu - mechanismus řešící kdo co může měnit, jak naložit s kolizními případy, následnost úprav
- log změn - záznam o všech provedených změnách, obsahuje původní a novou hodnotu
- kontrola čtyř očí
- vícenásobné pořízení dat

Auditabilita

je třeba vést záznamy o tom, kdo co dělal s kterými položkami - nejen pro to, abychom byli schopni sledovat přístupy a změny, ale i pro dlouhodobé sledování uživatelů a následné rozhodování, zda vyhovět žádosti
je nutné zvolit vhodnou granularitu - bloky, záznamy, položky
zde přistupuje tzv. *pass through problem* - uživatel smí přistupovat k objektu ale tento mu nesmí být předán (např. při vyhledávání)
uživatel může zjistit hodnotu položky i bez přímého dotazu - nestačí log žádostí o přístup k odhadu toho, co ví

Autentizace uživatelů

SŘBD potřebuje přesně vědět, komu odpovídá
protože však zpravidla běží jako uživatelský proces, nemá spolehlivé spojení s jádrem OS a tedy musí provádět vlastní autentizaci

Dostupnost

SŘBD má vlastnosti systému a aplikačního programu zároveň - běží jako program a používá služeb systému, pro mnoho uživatelů je však jediným pracovním prostředím
musí rozhodovat současné žádosti více uživatelů
musí být schopen odepřít poskytnutí i nechráněných dat aby nedošlo ke kompromitaci utajovaných informací

Spolehlivost a integrita

v prostředí databází jsou tyto pojmy ještě důležitější, než obvykle
jde nám o zachování těchto tří vlastností:

1. integrita databáze - celková správnost, ochrana před technickými závadami a poničením globálních struktur
2. elementární integrita - že změny a záznamy mohou provádět pouze autorizované entity
3. elementární správnost - že jsou přijata jen korektní data odpovídající typem, hodnotou, …

Dále se budeme zabývat mechanismy pro udržování těchto vlastností.
- tyto nejsou absolutní, např. nemohou zabránit oprávněným entitám vkládat nesprávná, leč typem a ostatními atributy akceptovatelná data

Ochrana poskytovaná OS

DBS jsou soubory a programy - tedy spadají pod standardní obranné mechanismy OS:

ochrana souborů, kontrola přístupu, zálohy, testy integrity na úrovni systému, …

Dvoufázový update

problémem je selhání výpočetního systému během provádění modifikace dat
proces modifikace rozdělíme na dvě fáze:

1. v průběhu první fáze - *záměr* (intent) se provede načtení relevantních dat, uzamčení záznamů, vytvoření pomocných záznamů a kalkulace výsledků
   poslední událostí první fáze je operace *commit*
2. v rámci druhé fáze jsou prováděny trvalé změny dat s ohledem na data získaná v první fázi (= zapsání výsledků)

Pokud některá z fází nedoběhne, může být snadno opakována, po ukončení fáze je systém konzistentní.

Třífázový update

řešení zápisů a čtení z distribované DB
oproti dvoufázovému update definuje ještě další „nultou fázi“ modifikace
0. ustanovení kvóra – zjišťuje se, je-li k dispozici dostatečný počet instancí distribuované databáze pro provedení operace

kvórum pro čtení – počet instancí, které musí souhlasit s provedením čtecí operace
kvórum pro zápis – počet instancí databáze, které musí souhlasit s provedením zápisu
součet obou kvór musí být větší než celkový počet instancí , kvórum pro zápis > 1/2

Cílem je zabránit stavu, kdy by část distribuované databáze poskytovala zastaralá data, resp. došlo k „rozdvojení“ vývoje datového obsahu

Redundance / Vnitřní konzistence

databáze často obsahují různé redundantní informace za účelem odhalení chyb, zvýšení spolehlivosti, nebo docílení potřebné rychlosti

Detekční a samoopravné kódy

jde o vhodné způsoby zakódování informací přidáním vhodné redundance tak, aby bylo s co možná největší pravděpodobností detekovat náhodné změny
samoopravné kódy mají navíc schopnost lokalizovat a vyčíslit jisté množství chyb, takže mohou být opraveny
vždy při ukládání je záznam zakódován, při načítání je kontrolována jeho správnost
k dispozici je celá ředa kódů pro různé účely

Stínové záznamy (Shadow fields)

vybrané atributy nebo věty jsou uloženy v několika kopiích
v případě nedostupnosti nebo chyby v originálu je použita kopie
metoda účinná leč náročná na prostor

Zotavení

SŘBD musí vést log o všech akcích, zejména o změnách
v případě havárie potom na základě těchto záznamů a vhodné záložní kopie znovu vygeneruje aktuální stav

Paralelismus / Konzistence

SŘBD musí zajistit současný přístup více uživatelů, vyřešit konflikty plynoucí z situací, kdy dochází k současnému zápisu více hodnot do stejné položky, nebo zápisu závisejícím po předchozím čtení položky

Monitory

jednotky SŘBD zajišťující strukturální integritu databáze - testují, zda vkládaná data typově odpovídají, zda jsou konzistentní s ostatními daty v systému atd.

Porovnání mezí

vkládaná hodnota je testována na příslušnost do jistého intervalu, meze intervalu mohou být určeny i dosti komplikovanou funkcí závisící na jiných hodnotách v databázi
tento test může být rovněž použit pokud je podezření, že uložená data jsou poškozena

Stavová omezení

popisují podmínky, které musí celá databáze splňovat
nejsou-li stavová omezení splněna, jsou data v databázi vadná
např. ve skladu nelze mít -5,1 rohlíku

Tranzitivní omezení

jsou omezení, které musí splňovat obsah databáze před provedením určité operace
např. před prodejem 1 rohlíku nesmí být sklad rohlíků prázdný

Senzitivní data

… data, která by neměla být veřejně známa
data se stávají senzitivními z mnoha různých důvodů:

- *přirozeně senzitivní* (inherently s.) - informace sama o sobě je utajovaná (plat toho kterého ministra)
- *ze senzitivního zdroje* - např. pokud je z informace patrné, kdo ji poskytl
- *deklarované jako senzitivní* - správce, nebo majitel DB prohlásí informaci za utajovanou
- *senzitivní atribut nebo záznam* - v DB může být jistý sloupec nebo řádek prohlášen za tajný
- *senzitivní ve vztahu k dříve vyzrazeným skutečnostem* - např. prozradím-li zeměpisnou šířku tajné základny, neměl bych už povědět nic o zem. délce

největší problémy nastávají v případě, kdy pouze *některé* informace v databázi jsou senzitivní

Rozhodování o přístupu

zavedeme následující značení:
správce SŘBD je program zajišťující dodržování rozhodnutí učiněných administrátorem o tom, kdo z uživatelů má mít jaká přístupová práva
pro jednoduchost budeme uvádět též správce rozhodne …

Dostupnost dat

kromě obvyklých problémů s dostupností informací, zde přistupuje mechanismus zamykání záznamů, který brání načtení nekonzistentních dat
uvažme případ, kdy dojde k poruše stanice, která právě provedla uzamčení záznamů (t.j. provedené zámky neuvolní)

Akceptovatelnost přístupu

SŘBD musí mít stále na zřeteli, které záznamy jsou senzitivní a nepřístupné uživateli, musí zajistit, aby nedošlo k jejich vyzrazení
problém rozhodování o poskytnutí přístupu je složitý, systém nesmí vydat informace, ze kterých by uživatel mohl dovodit utajované skutečnosti
přesto v některých případech by měl vydat výsledek, závisící na senzitivních informacích - např. různé statistické hodnoty z údajů v databázi

Zajištění autenticity

kromě správné identifikace lze opět omezit přístup uživatele časem, místem apod.
navíc, rozhodnutí o poskytnutí přístupu by mělo záviset též na předchozích dotazech, které uživatel kladl

Vyzrazení dat

Přesné hodnoty

uživatel se záměrně nebo omylem dotáže na tajná data, vlivem špatného mechanismu rozhodování nebo nestability v systému je dotaz zodpovězen

Meze

vydání mezí intervalu, ve kterém se utajované hodnoty nacházejí může být rovněž vážným nedostatkem - zvláště pokud systém tuto informaci vydá o libovolné podmnožině záznamů
na druhou stranu někdy může být vhodné moci podat informace o spodním a horním odhadu pro specifickou množinu záznamů

Negativní výsledek

uživatel může pokládat dotazy směřující k ověření záporné skutečnosti - hodnota specifické položky není rovna hodnotě *x*, časté zejména při ověřování, žehodnota je nenulová
zatímco informace, že hodnota je různá od 42 je téměř bezcenná, informace, že hodnota je nenulová sama o sobě již může znamenat vážné porušení utajení

Existence

stejně jako v případě obecných dat vrámci OS, sama existence nějaké informace může být senzitivní informace ☺
uživatel např. vůbec nesmí zjistit existenci určitého atributu - ani v projekcích, ani dotazy na strukturu databáze

Pravděpodobné hodnoty

opsaný příklad:
dotaz - Kolik lidí bydlí na Pennsylvania Avenue 1600?
odpověď - 4
dotaz - Kolik obyvatel Pennsylvania Avenue 1600 je registrovanými komunisty?
odpověď - 1
-> S pravděpodobností 25% je prezident USA komunista.

Bezpečnost versus přesnost

Systém by se měl snažit udržet *bezpečnost* senzitivních dat, a to i proti nepřímým dotazům - což vede ke "konzervativní" strategii v poskytování dat, která způsobí odmítnutí mnoha neškodných dotazů.

Z pohledu uživatele je naopak vhodné poskytovat co nejúplnější a nejpřesnější odpovědi - tedy udržet bezpečnost, ale vydat co nejvíce nesenzitivních informací.
Ideální stav by byl umět vydat právě všechny ne-senzitivní informace.

Problém odvoditelnosti

jde o možnost odvodit senzitivní informace ze znalosti (velkého množství) ne-senzitivních
Mějme následující DB:

Jméno	Pohlaví	Hodnocení	Podpora	Pokuty	Drogy	Kolej
Adams	M	C	5000	45	1	Holmes
Bailey	M	B	0	0	0	Grey
Chin	Ž	A	3000	20	0	West
Dewitt	M	B	1000	35	3	Grey
Earthart	Ž	C	2000	95	1	Holmes
Fein	Ž	C	1000	15	0	West
Groff	M	C	4000	0	3	West
Hill	Ž	B	5000	10	2	Holmes
Koch	Ž	C	0	0	1	West
Liu	Ž	A	0	10	2	Grey
Majors	M	C	2000	0	2	Grey

Přímý útok

útočník se snaží získat informace přímými dotazy, jejichž odpověď závisí na velmi malém počtu vět, které vyhověly podmínkám dotazu

```
list Jméno where
        Pohlaví = M & Drogy = 1
```

takový dotaz může obsahovat velké množství uměle vložených nesplnitelých podmínek

```
list Jméno where
        (Pohlaví = M & Drogy = 1) or
        (Pohlaví ≠ M & Pohlaví ≠ Ž) or
        (Kolej = Grey)
```

neb každý ví, že na koleji Grey o drogy nikdo ani nezavadí

Nepřímý útok

často je z databáze obsahující např. senzitivní osobní údaje povoleno zveřejňovat statistické údaje, které není třeba považovat za tajné
z těchto údajů lze ovšem za vhodných okolností získat původní utajované informace

Součet

Na první pohled nevinný dotaz na součet finanční podpory studentů dle pohlaví a koleje na které bydlí vede k vyzrazení tajné informace o výši finanční podpory studentky Liu

Počet

dotaz na počet může být kombinován s dotazem na součet
kombinujeme-li předchozí dotaz s dotazem na počet studentů toho kterého pohlaví bydlících na jednotlivých kolejích, zjistíme, že na koleji Holmes bydlí kdosi, kdo dostává 5000 finanční podpory
pokud se ještě zeptáme na seznam obyvatel této koleje (což pravděpodobně není tajné) opět jsme získali senzitivní informace

Medián

utajované hodnoty lze získat z dotazu na medián:
sekvence dotazů

```
q = median( Podpora where Pohlaví = M)
p = median( Podpora where Drogy = 2)
```

vede k vyzrazení přesné hodnoty finanční podpory studenta Majorse

Tracker attack

účinnou obranou je, pokud správce databáze odepře odpověď na dotazy, jejichž výsledek závisí na malém množství záznamů
útočník však může získat informace porovnáním výsledků několika dotazů:
namísto dotazu

count ((pohlaví = Ž) & (Hodnocení = C) & (Kolej = Holmes))

útočník položí následující dotazy

count (pohlaví = Ž)
count ((pohlaví = Ž) & (Hodnocení ≠ C) & (Kolej ≠ Holmes))

Odečtením výsledků získáme výsledek prvního dotazu, který správce nevydal

Ochrana proti odvoditelnosti

- rozbor dotazů i s ohledem na minulost - velice komplikovaný, nákladný a málo spolehlivý, efektivní pouze proti přímým útokům.
- ochrana vlastních dat - pasivní ochrana, hlavními metodami jsou potlačení (suppression) a skrytí (concealing) výsledků
  - potlačení - systém nevydá odpověď na všechny dotazy, ale případné odpovědi jsou přesné

  - skrytí - systém odpovídá na všechny dotazy, ale odpovědi jsou záměrně nepřené

Potlačení malých výsledků (limited response suppression)

systém nevydá odpověď, pokud výsledná hodnota nepřekračuje stanovený limit, případně závisí na příliš malém, nebo příliš velkém oboru (tzn. $<k$ nebo $>n-k$ řádků z celkových n pro zvolené k)

Kombinování výsledků

systém nevydává výsledky týkající se jednotlivých hodnot z dané domény ale pouze souhrné výsledky pro intervaly těchto hodnot

Modifikace výsledků(response modification)

systém spočítá přesný výsledek, který následně mírně poškodí
možností je zaokrouhlování výsledků, navracení průměru výsledků pro interval hodnot z dané domény apod.

Náhodný šum

před vyhodnocením výsledku je ke každé použité položce připočtena malá náhodna chyba, chybové hodnoty volíme jako náhodnou veličinu se střední hodnotou 0

Náhodný výběr (random sample)

systém neodpovídá na základě všech hodnot v databázi ale pouze na základě provedeného náhodného výběru ze všech relevantních položek
aby nebylo možné počítáním průměrných hodnot výsledků opakovaných dotazů získat přesné hodnoty, měl by se pro ekvivalentní dotazy používat stále stejný výběr

Náhodné zmatení (random data perturbation)

ke každé položce v databázi přičteme náhodnou chybu *e,* přičemž na rozdíl od náhodného šumu pro opakované dotazy systém zajišťuje, že použitá chyba je stále stejná
pokud je chyba z okolí nuly, je vliv této úpravy na statistické výsledky typu součet, průměr, … malý
metoda je vhodnější než předchozí, neboť lze snáze zajistit stejné výsledky ekvivalentních dotazů.

Víceúrovňové databáze (multilevel db)

opět se budeme zabývat případy, kdy je nutno uvažovat několik úrovní důvěrnosti či utajení zpracovávaných informací

je velmi důležitá granilarita, s jakou je ochrana prováděna - je jasné, že hodnoty jednotlivých atributů mohou mít rozdílnou citlivost, rovněž citlivost tak řádků se může lišit
v případě databází je dále třeba brát v úvahu

- senzitivita dané položky může být jiná než senzitivita všech ostatních položek v daném řádku nebo sloupci - je tedy třeba implementovat bezpečnost na úrovni položek
- obecně bude nutno počítat s několika stupni citlivosti a též s rozdělením dle "tematických okruhů" obdobně jako v případě mřížkového modelu
- citlivost agregovaných hodnot může být odlišná od citlivosti individuálních hodnot
- nestačí sledovat pouze jednotlivé hodnoty, ale je nutné chránit i různé kombinace uchovávaných hodnot, které mohou mít opět různou citlivost

SŘBD musí zachovávat integritu a utajení dat, ale sám se nemůže řídit např. *-vlastností popsanou v Bell-LaPadulově modelu (musí být schopen číst a zapisovat všechny položky)
utajování dat má i druhou stránku - pokud pracovník zjistí, že údaje poskytované databází nejsou z jeho pohledu úplné (byly vypuštěny utajené záznamy) může provést doplnění těchto "chybějících" záznamů

Metody ochrany ve víceúrovňových databázích

Parcelizace (partitioning)

databáze je rozdělena dle stupně citlivosti informací na několik subdatabází

- vede k zvýšení redundance s následnou ztíženou aktualizací
- neřeší problém nutnosti současného přístupu k objektům s různým stupněm utajení

Šifrování

senzitivní data jsou chráněna šifrováním před náhodným vyzrazením
zná-li útočník doménu daného atributu, může snadno provést chosen plaintext attack (zašifrováním všech hodnot z domény)
řešením je používat jiný klíč pro každý záznam, což je však poměrně náročné
v každém případě nutnost neustálého dešifrování snižuje výkon systému

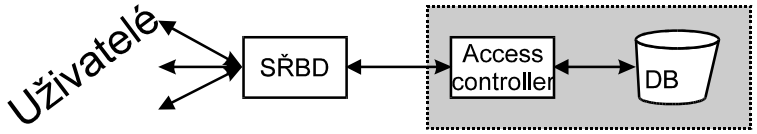
Integrity lock

každá položka v databázi se skládá ze tří částí:

<vlastní data : klasifikace : checksum>

- vlastní data jsou uložena v otevřené formě
- klasifikace musí být nepadělatelná, nepřenositelná a skrytá, tak aby útočník nemohl vytvořit, okopírovat ani zjistit klasifikaci daného objektu
- checksum zajišťuje svázání klasifikace s daty a integritu vlatních dat

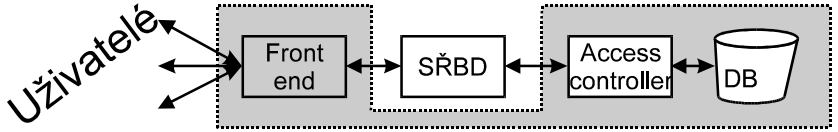
model byl navržen jako doplněk (access controller) komerčního SŘBD, který měl zajistit bezpečnost celého systému



šedá oblast vyznačuje bezpečnostní perimetr systému

Spolehlivý front-end (guard)

systém je opět zamýšlen jako doplněk komerčních SŘBD, které nemají implementován u bezpečnost



uživatel se autentizuje spolehlivému front-endu, který od něho přebírá dotazy, provádí kontrolu autorizace uživatela pro požadovaná data, předává dotazy k vyřízení SŘBD a na závěr provádí testy integrity a klasifikace výsledků před předáním uživateli
SŘBD přistupuje k datům prostřednictvím spolehlivého access kontroleru

Komutativní filtr (Commutative Filter)

jde o proces, který přebírá úlohu rozhraní mezi uživatelem a SŘBD

- filter přijímá uživatelovy dotazy, provádí jejich přeformulování a upravené dotazy posílá SŘBD k vyřízení
- z výsledků, které SŘBD vrátí, odstraní data, ke kterým uživatel nemá přístupová práva a takto upravené výsledky předává uživateli

filter je možno použít k ochraně na úrovni záznamů, atributů a jednotlivých položek

v rámci přeformulování dotazu může např. vkládat další podmínky do dotazu, které zajistí, že výsledek dotazu závisí jen na informacích. ke kterým má uživatel přístup

Pohled (View)

pohled je část databáze, obsahující pouze data, ke kterým má daný uživatel přístup
pohled může obsahovat i záznamy nebo atributy, které se v původní databázi nevyskytují a vznikly nějakou funkcí z informací původní databáze
pohled je generován dynamicky, promítají se tedy do něho změny původní DB
uživatel klade dotazy pouze proti svému pohledu - nemůže dojít ke kompromitaci informací, ke kterým nemá přístup
záznam / atribut původní databáze je součástí pohledu pokud alespoň jedna položka z tohoto záznamu / atributu je pro uživatele viditelná, ostatní položky v tomto jsou označeny za nedefinované
uživatel při formulování dotazu může používat pouze omezenou sadu povolených funkcí
tato metoda je již návrhem směřujícím k vytvoření bezpečného SŘBD

Bezpečnost v aplikačních serverech

aplikační server využívá OS a databázi jako persistentní repository vlastních dat včetně nastavení bezpečnostního mechanismu
nezbytný vlastní bezpečnostní mechanismus, zahrnující:

- autentizaci
- autorizaci
- auditní záznamy
- správu prostředků
- zajištění dostupnosti
- ochranu komunikace
- konzistenci dat
- řízení změn
- testování
- ...,

Standardem oddělení vývoje od testování (školení, sandbox, ...) a produktivního prostředí
Nezbytností řešit globálně personální politiku a separaci rolí
Vhodné dedikovat kapacity OS(DB) výhradně pro alikační server

Bezpečnost počítačových sítí

sítí budeme rozumět soustavu několika výpočetních systémů, uživatelé přistupují k síti prostřednictvím některého z těchto systémů
v této přednášce se budeme zabývat pouze bezpečnostní politikou, ne její implementací
zdroje bezpečnostních obtíží:

- Sdílení - potenciální přístup má velmi velké množství lidí, různé stroje mohou být řízeny různými ne nutně bezpečnými systémy
- Složitost - v síti se vyskytují nejrůznější operační systémy komunikující spolu via spojovací mechanismus, který by měl zajišťovat ochranu, tento mechanismus však musí být dostatečně obecný, navíc síť jako celek nelze podrobit testování či dokonce certifikaci
- Neznámý perimeter - nikdy nevíme, kdo všechno je připojen, není jasné, jak se ostatní stroje chovají
- Množství zranitelných míst - je nutné uvěřit bezpečnostním mechanismům na všech strojích, mnohé části sítě leží mimo jakýkoliv dohled provozovatelů
- Neznámá cesta - většinou nelze ovlivnit, kudy budou data přenášena, tedy není k dispozici žádná informace, kdo s nimi může přijít do styku

Ochrana komunikace

principiálně je možné chránit komunikaci jakožto:

- proud dat – někdy nazýváno jako „stream enciphering“, tj. šifrování proudu dat, kdy se vytváří dojem, že komunikační kanál je spolehlivý z hlediska možného útoku
- jednotlivé zprávy – odpovídá dnes moderní volné vazbě systémů pomocí „messagingu“, šifrují se aplikační zprávy, nebo jejich relevantní části

proudové šifrování lze provádět mezi dvěma uzly sítě, nebo mezi dvěma aplikacemi běžícími na těchto uzlech

Šifrování na úrovni linky (Link Encryption)

data jsou šifrována těsně před vstupem do komunikačního media, dešifrována ihned po příchodu na druhý počítač
toto šifrování probíhá na úrovni fyzické případně linkové vrstvy referenčního modelu
výhodou je, že tento mechanismus je pro uživatele transparentní a může být i velmi rychlý, navíc je snadno připojitelný k stávajícím zařízením

je zcela nevhodný pokud nejsme schopni ovlivnit, kudy budou data přenášena

End-to-End šifrování

poskytuje kryptografickou ochranu po celou dobu přenosu
toto šifrování probíha přibližně na úrovni aplikační nebo prezentační vrstvy referenčního modelu
toto šifrování však již nebývá transparentní a má-li být účinné, musí být vhodně zakomponováno do celého systému
další výhodou je, že není nutno šifrovat veškerou komunikaci, ale pouze citlivá data na rozdíl od šifrování linky je schopno zajistit autentizaci a integritu (end-to-end)

někdy jsou používány obě zmíněné metody zároveň - šifrování linky za účele běžné preventivní ochrany dat a End-to End šifrování k docílení skutečně kvalitní ochrany senzitivních dat

se zavedením šifrování souvisí nutnost existence mechanismu distribuce a správy nezbytných šifrovacích klíčů, potřebných centrálních autorit pro zajištění provozu systému kryptografické ochrany, vhodných kryptografických zařízení zajišťujících základní funkce kryptografické ochrany

Kontrola přístupu

v případě sítí přistupují k obvyklým problémům kontroly přístupu ještě následující okruhy

Ochrana komunikačních portů (Port protection)

před mechanismus autentizace uživatele lze předřadit ještě ochranu vlastního komunikačního portu

Automatické zpětné volání

metoda vhodná pro komutované (dial-up) spoje
Poté, co je navázáno spojení a uživatel se identifikuje, systém ukončí spojení, v interních tabulkách zjistí adresu (tfn. číslo) daného uživatele a pokusí se o navázání spojení na tuto adresu.
Tímto způsobem je zajištěno, že přístup je možný pouze z omezeného množství jiných uzlů (adres) a tedy výrazně omezena či alespoň zkomplikována možnost průniku ‘zvenčí’.

Odstupňovaná přístupová práva

Přístup k senzitivním datům může být omezen na pouze některé uzly. Pokud i autorizovaný uživatel žádá o přístup z jiného uzlu, mohou jeho přístupová práva být výrazně omezena, nebo může být zcela odepřen přístup k datům.

Tichý modem (Silent Modem)

Po přijetí volání modem nezačne bezprostředně generovat nosnou, ale počká, až se druhá strana pokusí o negotiation.
Tím je přístup do jisté míry omezen pouze na uživatele, kteří vědí, že jde o linku vedoucí k počítači, metoda omezuje možnost náhodného nalezení tohoto portu.
Obdobou tichého modemu může být v případě IP protokolu služba, dostupná na daném stroji na jiném než obvyklém portu.

Řízení přístupu z vnějšího prostředí

- Firewally – filtry, aplikační brány
- Překlad adres
- Kontrola přenášených dat – antiviry, java, scripty,
- Omezení přístupu ke zdrojům / obsahu dat
- Prioritizace, qos
- IDS, IPS (intrussion detection/prevention system) – síťové, aplikační
- Demilitarizované zóny
- Content filtry

Parcelizace vnitřní sítě

- Oddělení kritických zdrojů, zónování
- Vydělení zvláštní sítě pro senzitivní informace
- Traffic shaping

Autentizace uzlů

Je třeba, aby existovaly mechanismy umožňující vzájemnou autentizaci jednotlivých uzlů, ne pouze uživatelů.

Autentizace v síti

Protože síťové prostředí zpravidla není považováno za bezpečné, je třeba využívat autentizační mechanismy odolné vůči odposlechu, resp. aktivním útokům
Často bývá žádoucí řešit *jednotné přihlášení (single sign on)*

- cookies

- tickety
- certifikáty, PKI
- čipové karty
- ...

S procesem integrace autentizačních mechanismů souvisí nutnost zavedení centrální správy uživatelů nebo alespoň synchronizace záznamů o uživatelích

Aktivní útočník

v případě jednotlivých strojů může být útočníkem člověk, v prostředí sítí však již útočník může používat počítač a pokoušet se aktivně poškozovat systém ochrany dat

Playback starších zpráv

pokusy o znovupoužívání starších zachycených zpráv
vhodnou metodou ochrany jsou časová razítka v kombinaci s šifrováním, různé tokeny s omezenou časovou platností, notarizace, nebo ofsetování zpráv.

Narušení služeb

velmi snadným způsobem útoku je přetěžování sítě nesmyslnými zprávami
rovněž účinnou metodou je pokusit se pozměňovat routovací informace
rovněž je možné zachycovat, nebo alespoň poškozovat zprávy zasílané určitému uživateli
proti mnohým útokům je možno se bránit vytvořením duplicitních linek, po kterých mohou být zprávy posílány, pokud je to možné, lze se omezit pouze na důvěryhodné uzly

Vkládání poškozených zpráv

útočník může vkládat poškozené zprávy, při jejichž zpracování může dojít ke zhroucení službu konajícího stroje, nebo k jeho nesprávné funkci

Řízení zátěže

útočník může zachycovat veškerou komunikaci a provádět rozbor, kdo s kým jak často komunikuje - jde o tzv. *analýzu zátěže* : z náhlých změn zátěže lze usuzovat na nadcházející výrazné události
vhodnou metodou ochrany je generování *vycpávací (pad) zátěže* v době, kdy nedochází ke skutečné komunikaci

Vycpávací zátěž

generovaná vycpávací zátěž může být prostředkem pro vytvoření skrytého kanálu
administrátor tedy musí zajistit generování dalších vycpávacích zpráv doplňujících komunikaci mezi libovolnými dvěma uzly sítě

Kontrola routování

administrátor může aktivně zasahovat do procesu routování a náhodně měnit způsob routování některých zpráv, čímž se dosáhne větší náhodnosti do procesu přenosu zpráv a omezí možnost předchozích útoků

Další metody ochrany

administrátor může aktivními zásahy zvyšovat bezpečnost:

- náhodně zachycovat a mazat zprávy
- náhodně měnit adresáta zprávy na nejnižší úrovni
- pozdržovat doručení náhodně vybraných zpráv

Integrita dat

přenos zpráv je řízen přenosovými protokoly zajišťujícími integritu dat, pořadí doručených částí, detekci duplicit apod.
pro účely ochrany dat nedostačující - tyto informace jsou v plaintextu bez potřebné detekce modifikací
různé zabezpečovací kódy je snadné replikovat
vhodnější je použití kryptografických kontrolních součtů - zdeje vhodné do každého šifrovaného bloku zprávy přidat jeho pořadové číslo, aby útočník nemohl provádět záměny pořadí
notarizace zpráv - každou posílanou zprávu je možné nechat ověřit centrální autoritou

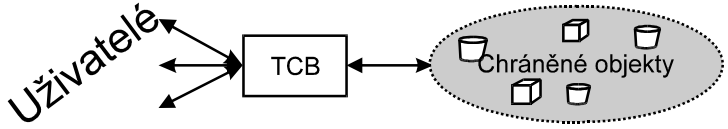
Lokální sítě

jsou v mnohém specifické - jejich uživatelé často jsou lidé pracující ve společném oboru, bývají laici v oblasti počítačů, povětšinou si mezi sebou do značné míry důvěřují
problémy nastávají při vzájemném propojování těchto sítí
lokální síť má většinou jednotnou topologii, dle které lze modifikovat použité ochranné mechanismy
vzhledem k tomu, že lokální síť bývá umístěna uvnitř jedné budovy, či dokonce pouze její části, lze uplatnit různé metody fyzické ochrany
lokální síť rovněž mívá administrátora, který může efektivně vynucovat dodržování stanovené bezpečnostní politiky ve všech uzlech

zvýšená míra důvěry však vede ke snížení obranyschopnosti v případě náhlého útoku či zvýšení jeho hrozby

Víceúrovňová bezpečnost

rovněž v počítačových sítích mohou pracovat uživatelé s různým stupněm prověření, síť obsahuje data různých stupňů utajení
nejčastěji se používá nějaká modifikace military security modelu
operační systémy, navrhované pro vysokou bezpečnost bývají rozděleny na moduly, na jejichž bezpečnost nejsou kladeny nároky, které přistupují k chráněným objektům prostřednictvím spolehlivých modulů, jež tvoří *spolehlivou výpočetní bázi (TCB)*

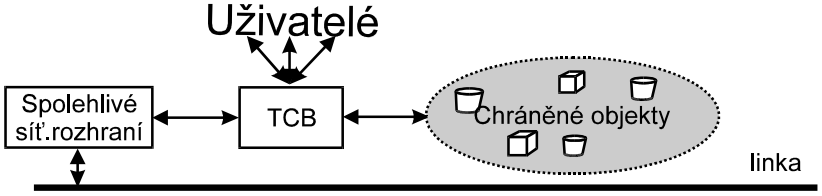


obdobně pro sítě:

Spolehlivé síťové rozhraní (trusted network interface)

každý uzel sítě musí být "opatrný" vůči ostatním uzlům, měl by zajistit, že spojení naváže pouze s dalším uzlem, který má spolehlivé síťové rozhraní
funkce spolehlivého síťového rozhraní:

1. zajištění bezpečnosti vlastního uzlu - před útoky zvenčí
2. veškerá výstupní data musí být označena příslušnou bezpečnostní klasifikací
3. před uvolněním dat je provedena verifikace oprávněnosti žadatele a jeho autentizace
4. ověření konzistence došlých dat
5. nesmí docházet k míchání dat různého stupně utajení, nebo samovolnému předávání informací ostatním uzlům
6. bezpečnost dat nesmí záviset na bezpečnosti linky



Bezpečná komunikace

vlastní síť včetně příslušných řídících modulů není uvažována
bezpečnou komunikaci zajišťují samy komunikující procesy ve spolupráci s operačním systémem
jsou dodržována pravidla Bell-LaPadula bezpečnostního modelu
pokud chceme zavést potvrzování zpráv, je nutné, aby na každém uzlu běžel pro každou bezpečnostní úroveň komunikační server - posílá-li proces zprávu procesu vyšší úrovně na jiném uzlu, zašle ji tamnějšímu kom. serveru své úrovně, od kterého obdrží potvrzení a který ji předá
posílání zpráv procesům nižší úrovně probíhá prostřednictvím spolehlivé centrální autority - *network manažera*, který zkoumá, zda nedochází k únikům klasifikovaných informací

Bezpečné síťové spojení

spolehlivá síťová rozhraní rozdělíme na moduly se vstupnímy a výstupnímy sokety
pokud u daného modulu můžeme dokázat, že jeho výstupy závisí pouze na některých vstupech - *multilevel modul* - může mít výstupní sokety různých úrovní citlivosti
jinak má modul výstupy odpovídající nejvyšší citlivosti vstupu

opět budeme dbát na zachování pravidel Bell-LaPadula modelu, tzn. výstup modulu může být připojen pouze na vstup jiného s nejméně stejným stupněm citlivosti
každý proces prohlásíme rovněž za modul se specifickým stupněm citlivosti
takto lze definovat povolená spojení v rámci celé sítě

Bezpečnost komunikace

bezpečnost je do určité míry závislá na použitém přenosovém mediu
útok proti komunikačním linkám může být *pasivní* (pouze odposlech), nebo *aktivní* (vkládání dalších informací do komunikace)

Kabely

častým útokem je tzv. *napíchnutí* (wiretaping)
proti tomuto způsobu útoku jsou obzvláště bezbranné metalické vodiče, je však možné monitorovat i optické kabely
obecně lze mezi metalickými kabely považovat za bezpečnější kabely koaxiální
vyrábí se celá řada kabelů s omezeným vyzařováním případně s detekcí napíchnutí
k napíchnutí jsou náchylnější pevné linky (leased lines), obecně je útok pravděpodobnější u některého z konců linky

Mikrovlny

svazek není možno zcela přesně měrovat, navíc se mírně rozbíhá
komunikace může být zachycena kdekoliv mezi vysílačem a přijímačem, nebo v prostoru za přijímačem
obdobné nedostatky z hlediska možnosti aktivního útoku

Satelitní přenos

poznamenejme, že ta samá technologie je používána k **šíření** TV signálu

Celulární radio

nebezpečí útoku je velké, vlastní zejména pasivní útok je snadno proveditelný

Analogové sítě

většina těchto sítí původně navržena pro přenos hlasu,
častým problémem autentizace, sítě většinou neposkytují informace o zdroji přenášené informace
problém lze řešit zejména použitím kryptografie v zařízeních tvořících rozhraní těchto sítí a počítače, dosud však neexistuje dostatek standardů

X.25

veřejné datové sítě již poskytují prostředky pro autentizaci entit, umožňují vytváření uzavřených logických podsítí celé sítě
vlastní komunikační lilnky bývají spolehlivější
lze rovněž provádět end-to-end šifrování přenášených dat, bohužel již ne šifrování záhlaví zpráv, tedy je možná analýza zatížení
nebezpečí představují připojení via trojici protokolů X.3, X.28, X.29 - po analogových linkách - zde nelze spoléhat na autentizaci
obecným problémem potom je propustnost těchto sítí

ISDN

poskytuje celou řadu identifikačních služeb počínajíc identifikací volajícího či volaného účastníka a končíc možností omezit nebo zcela vyloučit spojení z (jiných než) určitých směrů
daleko propracovanější vytváření logiských podsítí

Pevné linky

výhodou je, že linku používá pouze nájemce, je jisté spojení
na druhou stranu okruh vede stále stejnou cestou, je snadněji k nalezení a následnému odposlechu

X.400 - message handling

protokol poskytuje kompletní škálu bezpečnostních funkcí - autentizace původu zpráv, důkazy přijetí, security labeling, utajení toku dat a spojení, autentizace entit, bezpečnost přenášené informace, zajištění integrity a neopakovatelnosti