* rozpoznávání hlasu - telefon, mikrofon
 - minimální nároky na HW
 - používá se pro některé systémy objednávek po telefonu
 - problémy: schopnost rozpoznávání negativně ovlivňuje hlučné prostředí, nemoc, stáří, zranění uživatele

* rozpoznání pohybu pera při podpisu - je jedinečné, nelze napodobit jen na základě znalosti podpisu
 - speciální pero poskytuje informaci o rychlosti, směru a tlaku
 - při zápisu (enrollment) se uživatel 5x podepíše
 - zatím se příliš nepoužívá, protože jsou praktické problémy - senzory dlouhodobě nevydrží hrubé zacházení veřejnosti
 - má podporu v oblastech, kde se tradičně používá podpis, tj. bankovnictví, podpis smluv apod.

* integrace rozpoznávání tváře, hlasu a pohybu rtů
 - komerčně dostupný systém (BioID)

* další vývoj:
 - přenositelná (portable) nebo nositelná (wearable) zařízení - integrovaná např. do brýlí (testováno např. US armádou pro pohraniční stráž v Bosně)
 - čipy pro otisky prstů na bankovních kartách (přístup k tajnému klíči pouze pro osobu vlastníka)
 - pokusy o standardizaci - v současnosti na trhu více než 150 výrobců, každý vlastní HW a SW rozhraní - standard BioAPI
 - Microsoft původně BioAPI Consortium spoluzakládal, ale odpojil se a od r. 2000 integruje do MS Windows technologii získanou od I/O Software

Externí útoky na systém a obrana proti nim

* útok proti počítačovému systému probíhá obvykle v následující posloupnosti:
 - vyhledání vhodného počítače (uzlu sítě, počítače připojeného přes modem)
 - zjištění informací o počítači (OS, spuštěné servery apod.)
 - napadení systému

* pro všechny tyto kroky existují na síti nástroje, tj. útočník nemusí být žádným odborníkem na téma počítačová bezpečnost

Poznámka (cracking jako trestná činnost)

V ČR je cracking považován za trestnou činnost proti majetku. Ve všech krajích má Policie ”oddělení počítačové expertízy”.

[]

Vyhledávání vhodných počítačů

* útočníci používají nejčastěji následující tři příbuzné techniky:
 - mapování sítě
 - wardialing
 - hledání bezdrátových sítí

Mapování sítě

* dnes nejčastěji na TCP/IP sítě (Internet, intranet podniku)
* útočník nejprve zjišťuje IP adresu nebo množinu IP adres, které by mohl napadnout
* jeden způsob je dotaz do DNS, získá doménové jméno instituce a některé IP adresy

* pokud chce více informací, zjišťuje topologii sítě a které adresy jsou aktivní
* používají se mapovací programy - používají 2 základní způsoby:

```
 - ping
  . vynucuje si odpověď vzdáleného stroje pomocí ICMP datagramu ECHO_REQUEST
  . testovaný stroj nebo gateway odpoví ICMP ECHO_RESPONSE
 - zahájit TCP spojení zasláním TCP "SYN" na nějaký port
  . pokud je na portu server, odpoví TCP "SYN ACK", pokud není odpoví
    ICMP PORT_UNREACHABLE
  . používá se na sítích které blokují ICMP ECHO_REQUEST
* obrana proti mapování sítě:
 - administrátor by měl periodicky kontrolovat síť a zjišťovat zda nejsou
   připojeny nepotřebné systémy (systémy které nikdo nepoužívá)
 - pokud ano, nepotřebné systémy odpojit


Wardialing
..........

* wardialers = programy které vytáčejí množinu telefonních čísel, snaží se
  najít číslo na kterém je připojen modem

* útočník například zjistí telefonní číslo cílové instituce (např. z WWW
  stránek), pak volá na všechny klapky
* získá seznam čísel, na kterých jsou připojené modemy
* pak se může připojit pomocí terminálového programu, zjistit typ systému a
  zda je vyžadováno heslo

* obrana proti wardialing
 - instituce by měla mít pravidla pro připojování modemů na telefonní linky
 - všechny modemy by měly být centrálně registrovány (databáze by neměla být
   veřejně přístupná)
 - zodpovědná osoba by měla periodicky provádět kontrolu pomocí wardialing,
   neregistrovaná zařízení by měla být odpojena

Poznámka pro zajímavost (podle Denningové 1999)

 Experiment s obvoláním 2.6 milionu čísel Berkeley našel 20 000 modemů,
 z nichž 200 bylo zcela nezabezpečeno.

[]

* pokud je vyžadováno heslo, pokouší se ho uhodnout - zkouší statisticky
  pravděpodobné kombinace jméno/heslo (root/root, guest/guest, uucp/uucp, ...)

Hledání bezdrátových sítí
.........................

* často jsou použity neoficiálně nezkušenými uživateli => další přístupové
  body do sítě

* útočník může zjistit jejich přítomnost do vzdálenosti několika set metrů
* většina přístupových bodů v sítích založených na protokolu 802.11b odpoví
  na SSID broadcast své SSID (Services Set Identifier, tj. název přístupového
  bodu)
* problém i pokud nakonfigurovány tak, že neodpoví, protože SSID se posílá jako
  plaintext => lze odposlechnout
* útočníkovi postačuje laptop, bezdrátová karta, anténa a příslušný SW

* po nalezení sítě se pokusí získat IP adresu pomocí DHCP (v mnoha sítích
  dostane IP adresu kdokoli o ní požádá), pak může přistupovat k síti

* obrana proti útokům na bezdrátové sítě
 - podobně jako u modemů - pravidla pro připojování a registrace
 - nastavit SSID na nepředvídatelnou hodnotu a zakázat odpověď na SSID
   broadcast
 - filtrování MAC adres
 - použít virtuální soukromou síť - autentizace a šifrování
```

Zjištění informací o počítači

Ve chvíli kdy útočník získá přístup k síti (prostřednictvím chyby v aplikaci nebo firewallu, nezabezpečeného modemu nebo bezdrátové karty) a zná topologii sítě, obvykle se pokusí zjistit něco o strojích přístupných na síti.

* zjištění OS stroje (OS fingerprinting), může být aktivní nebo pasivní
* scan portů - zjištění na kterých portech běží servery
* hledání zranitelných bodů.

Zjištění typu OS

* protokoly TCP/IP jsou sice definovány v příslušných RFC, RFC ale nepokrývají odpovědi na všechny typy nesprávně vytvořených paketů
  - např. odpověď na TCP SYN má být ACK, jak ale odpoví pokud jsou nastaveny nesmyslné příznaky (SYN-FIN-URG-PUSH)?
  - každý OS a protokolový zásobník odpoví jinak => lze určit typ OS

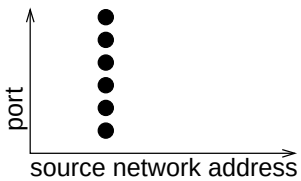
* pokud oponent umí síť odposlouchávat (broadcast technologie + je na stejné síti nebo na síti se kterou cílový stroj často komunikuje), může analyzovat probíhající provoz a na jeho základě určit typ OS
  - např. podle počátečního TTL, velikosti okénka, maximální velikosti segmentu atd.

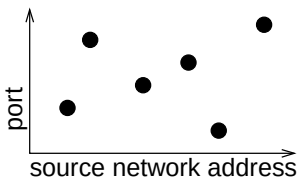
* obrana proti zjištění typu OS (sama o sobě příliš nepomůže)
  - v Linuxu 2.4 můžeme nastavit ”osobnost OS” pro chování IP stacku pomocí http://ippersonality.sourceforge.net
  - obrana proti pasivní analýze - proxy firewall => všechny systémy budou skryty za firewallem, informace v hlavičce neprojdou mimo lokální síť

Scan portů

* vzdálený konec komunikace je určený IP adresou, protokolem a portem
  - servery poslouchají na určeném portu (administrátor může určit)
  - nejčastější služby na dobře známých portech, např. port 80 je určený pro WWW server apod. (viz např. RFC1700)
* scan portů identifikuje na kterých TCP a UDP portech jsou spuštěné servery

* protože scan portů většinou předchází útoku, má smysl instalovat nástroje které ho detekují a pošlou např. mail administrátorovi (např. iplogger)
  - zjistí zda v určitém čase zaslány pakety na různé porty se stejné IP adresy
* naneštěstí možné zamlžit distribuovaným scanem
  - více strojů, každý pošle pouze jeden paket => obtížněji detekovatelné





Hledání zranitelných bodů

* ve chvíli kdy zná cílové systémy a jejich servery, bude útočník vyhledávat slabá místa
  - existují tisíce chyb, aby je nebylo nutné zkoušet ručně existují scanery
  - mají obvykle databázi do které je možné snadno přidávat nové testy pomocí skriptovacího jazyka

* obrana proti hledání zranitelných bodů:
  - zrušit všechny služby které nejsou nutné
  - periodicky kontrolovat vlastní síť, detekované potíže včas vyřešit

Napadení sítě

* pokud ruční napadení, útočník se pokouší přihlásit - odhadnout jméno a heslo
  - přihlašovací jméno je často příjmení, případně křestní jméno
  - podle klasického článku (Morris & Thompson 1979) je cca 90% hesel jména a příjmení, jména měst, slova ze středně velkého slovníku apod.
  - podle (Kabay 1997) na bylo finančním úřadu v Londýně 82% hesel snadno uhodnutelných - neslušná slova, jména lidí (nejčastěji členové rodiny nebo oblíbený sportovec), místo dovolené nebo běžné objekty vyskytující se v okolí úřadu

* ve chvíli kdy je útočník v systému a má administrátorská práva, může nainstalovat program pro odposlech sítě apod.

Odposlechy sítě

* pokud je síť založená na broadcast technologii, jako je Ethernet + HUBy, mohou všechny počítače na segmentu monitorovat provoz
* v některých OS programy defaultně, např. v UNIXu program tcpdump, ngrep apod. (používají další utility pro sledování provozu sítě, např. pro detekci scanů portů)
* mnoho protokolů zasílá hesla v OT (telnet, ftp, POP) - je možné vyfiltrovat hledané sekvence, např. v protokolu ftp:

```
user luki
331 Password required for luki.
pass ******
230 User luki logged in.
```

* útočník může odposlechnuté kombinace jméno/heslo použít pro získání přístupu k dalším systémům

* obrana proti odposlechům:
  - eliminace broadcast sítí, např. pro Ethernet použít místo HUBů switche
  - zakázat nešifrované služby, šifrovat přenášená data, např. SSH, SSL Telnet

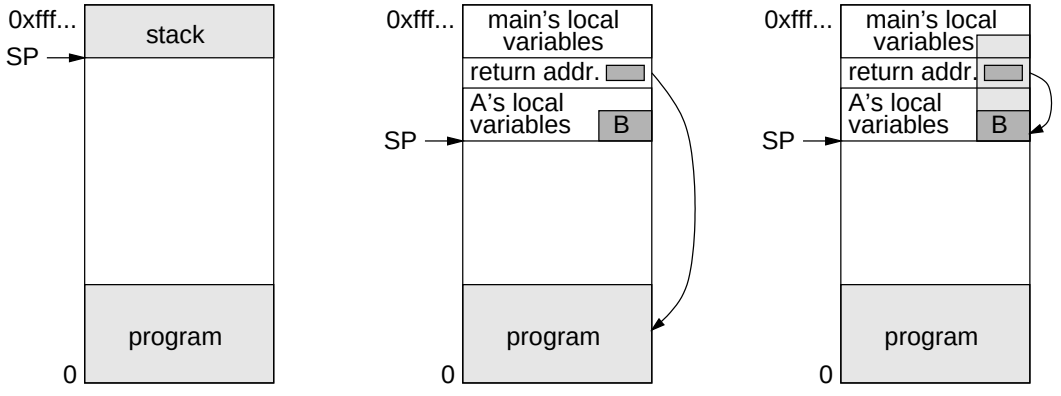
* protože mnoho sítí přechází na technologii přepínaného Ehernetu, objevily se programy pro tzv. aktivní odposlech (active sniffing)
* některé používané techniky:
  - zaplavení přepínače MAC adresami - přepínač autodetekuje MAC adresy související s jednotlivými rozhraními; pokud se vyčerpá paměť, často přepne do broadcast módu
  - přesměrování provozu pomocí protokolu ARP (RFC 826, RFC 920)
    . používá se pokud systém zná IP adresu cíle, který je na stejné podsíti, ale nezná jeho HW adresu: pošle ARP dotaz jako broadcast, cíl odpoví, odpověď se uloží do cache
    . na stroji sloužícím pro odposlech IP forwarding na bránu sítě, ostatním strojům ARP že jsem brána
  - falešné DNS odpovědi - předá vlastní IP adresu, může pracovat jako relay (tímto způsobem je možný i man-in-the-middle útok proti SSL)

* obrana proti aktivnímu odposlechu:
  - šifrovat data, pokud SSH klient vypíše varování že se změnil veřejný klíč serveru je zapotřebí zjistit, co se stalo
  - na kritických podsítích používat statické nastavení přepínačů, statické ARP tabulky

Převzetí relace

* nástroje pro násilné převzetí relace umožňují útočníkovi buď převzít existující relaci (a vyhodit z ní původního uživatele) nebo se k existující relaci připojit a vkládat do ní vlastní příkazy
* spoléhá opět na broadcast technologii
  - Oskar je na síťovém segmentu Alice, Boba, nebo mezi nimi
  - program Oskara odposlechne sekvenční čísla paketů atd. putujících mezi Alicí a Bobem

```
 - při převzetí relace používá IP spoofing - předstírá IP adresu Alice

* obrana proti převzetí relace:
 - nepoužívat nezabezpečené protokoly pro citlivé relace (systémová administrace)

* s převzetím relace souvisí jednodušší IP spoofing - podvržení IP adresy
 - nastavení zdrojové IP adresy podle potřeby
 - problém tam kde ověřování podle IP adresy, např. rlogin, rsh, rexec, rcp

* obrana: nepoužívat IP adresu jako základ pro autentizaci


Napadení systému
----------------

* v systémech je mnoho chyb, prostřednictvím kterých je možné získat
  neautorizovaný přístup do systému

* jako první příklad uvedu chybu pocházející z tradičních BSD UNIXových
  systémů (v nyní používaných systémech je pochopitelně opravena)
 - normální přihlašování v UNIXových systémech:
  . pro každý terminál proces getty, který otevře terminál a vypíše uživateli
    přihlašovací výzvu login:
  . uživatel zadá přihlašovací jméno -> getty spustí místo sebe
    "login username"
  . program login si vyžádá zadání hesla, pokud bylo jméno a heslo zadáno
    správně, login místo sebe spustí příkazový interpret - shell
 - v BSD má administrátor možnost spustit login -f username => administrátor
   je přihlášen jako uživatel
 - chyba: login akceptoval i bez mezery, tj. -fusername => bylo možno se
   přihlásit jako root: login: -froot

Přetečení bufferu uloženého na zásobníku
........................................

* téměř všechny OS a většina systémových programů je vytvořena v jazyce C
  (programátoři ho mají rádi a lze efektivně přeložit)
 - naneštěstí C nekontroluje meze polí - pole jsou ve skutečnosti pouze jiný
   zápis operace nad ukazatelem, tj. a[i] je totéž co (a+i)
 - umožňuje přepsat části paměti, což může způsobit potíže
* vede k útokům následujícího charakteru
 - program běží, volá proceduru A (viz obrázek a)
 - při volání vloží na zásobník návratovou adresu, spustí proceduru A a ta
   vytvoří na zásobníku místo pro lokální proměnné, např. buffer B velikosti
   1024 bytů (viz obrázek b)
 - uživatel zadá delší řetězec, např. 2000 bytů a tím přepíše část další
   paměti, včetně návratové adresy (viz obrázek c)
 - při vykonání instrukce RET se začnou vykonávat instrukce od adresy dané
   daty, která přepsala původní návratovou adresu - pokud náhodné, program
   většinou havaruje
```




```
* data ovšem nemusejí být náhodná, může to být program + návratová adresa na
```

```
   začátek programu (resp. bufferu B)
  - po návratu z podprogramu se program v B spustí, obvykle spustí shell
  - lze pro servery v UNIXu i ve Windows
  - notorickým zdrojem těchto problémů byl podprogram gets(s), který
    nekontroluje délku bufferu (proto by měl používat raději fgets(); např.
    gcc při použití gets() vypíše varování "the 'gets' function is dangerous
    and should not be used.")

* obrana:
  - programy by měly kontrolovat zda se vstup vejde do alokované paměti
  - programů které mají speciální práva by v systému mělo být co nejméně
  - OS nakonfigurovat tak, aby nespouštěl kód umístěný na zásobníku (lze v
    mnoha UNIXových systémech, např. v Linuxu a Solarisu; některé programy to
    ale potřebují)
```

❇