

Bezpieczeństwo Systemów Operacyjnych Zagrożenia i Ataki na System Operacyjny

Tomasz R. Surmacz

Politechnika Wrocławska

27 kwietnia 2021



CYBERBEZPIECZEŃSTWO 2.0



Rzeczpospolita
Polska



Politechnika Wrocławska

Unia Europejska
Europejski Fundusz Społeczny



Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego, Program Operacyjny Wiedza Edukacja Rozwój, Priorytet III Szkolnictwo Wyższe dla gospodarki i rozwoju, Działanie 3.5 Kompleksowe programy szkół wyższych w ramach konkursu nr POWR.03.05.00-IP.08-00-PZ3/18 na Zintegrowane Programy uczelni - Ścieżka III, nr umowy POWR.03.05.00-00-Z308/18-00
Tytuł projektu: „Cyberbezpieczeństwo dla gospodarki przyszłości”

- ▶ Nieuprawniony dostęp do systemu
włamania, instalacja backdoorów (pendrive, uaktualnienia), hasła, sniffing, socjotechniki
- ▶ Nieuprawniony zdalny dostęp
programy sieciowe, serwery, spoofing, biblioteki sieciowe
- ▶ Nieuprawniony dostęp do danych (lub programów)
błędy w programach, błędy projektowe, luki w systemie, uprawnienia
- ▶ Eskalacja uprawnień
prawa dostępu, bity suid, euid,
- ▶ Zdalne wykonanie kodu
serwery sieciowe, skrypty serwera WWW, inne narzędzia

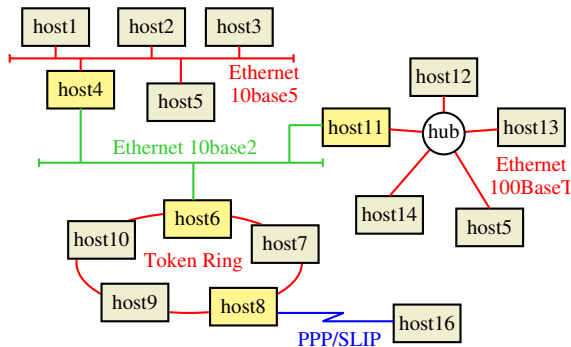


- ▶ Nieuprawniony dostęp do systemu
włamania, instalacja backdoorów (pendrive, uaktualnienia), hasła, sniffing, socjotechniki
- ▶ Nieuprawniony zdalny dostęp
programy sieciowe, serwery, spoofing, biblioteki sieciowe
- ▶ Nieuprawniony dostęp do danych (lub programów)
błędy w programach, błędy projektowe, luki w systemie, uprawnienia
- ▶ Eskalacja uprawnień
prawa dostępu, bity suid, euid,
- ▶ Zdalne wykonanie kodu
serwery sieciowe, skrypty serwera WWW, inne narzędzia



- ▶ Kontrola dostępu
 - ▶ Konta i hasła dostępowe
 - ▶ Terminal i terminale zdalne
 - ▶ Filtry pakietów
 - ▶ Firewall
- ▶ Serwery sieciowe
 - ▶ Zdalny dostęp interaktywny (`login`, `rlogind`, `telnetd`, `sshd`, itp.)
 - ▶ Serwery usług sieciowych (WWW, MySQL, inne)
 - ▶ Architektura frontend-backend
- ▶ Ataki
 - ▶ Sniffing (podśluchiwanie sieci)
 - ▶ Socjotechniki (wykradanie informacji)
 - ▶ Spoofing (podszywanie się)

- ▶ Warstwy sieciowe (1-4) w sieciach TCP/IP przesyłają dane w postaci jawnej.
- ▶ Metoda komunikacji w medium transmisyjnym – rozgłaszanie (broadcast).
- ▶ Mając fizyczny dostęp do medium transmisyjnego (kabel sieciowy, host) można podsłuchiwać cały przechodzący ruch.



Punkty narażone na podsłuch

- ▶ Kable w sieci lokalnej, huby, routery sieci lokalnych.
- ▶ Wszystkie routery i urządzenia sieciowe znajdujące się pomiędzy komunikującymi się hostami.
- ▶ Wszystkie urządzenia sieciowe zdolne do przekierowywania ruchu.
- ▶ Lokalna stacja robocza.
- ▶ W systemach z dostępem zdalnym: lokalny terminal lub pseudo-terminal (`/dev/ttyXX`, `/dev/ptyXX`).
- ▶ Pliki tymczasowe w katalogu `/tmp` z niewłaściwymi prawami dostępu.

Punkty narażone na podsłuch – poziom protokołów (warstwy 5-7)

- ▶ SMTP: wszystkie hosty MX,
- ▶ WWW, FTP: serwery proxy,
- ▶ Firewall w sieci lokalnej – idealne miejsce do włamań i założenia „podsłuchu” całej sieci.
- ▶ radio przez internet: routery multicastowe
- ▶ IRC: serwery IRC

Sniffery

- ▶ Wiele różnych programów dostępnych jest zarówno w środowisku UNIX jak i MS-Windows.
 - ▶ Solaris: `snoop`
 - ▶ Linux, *BSD: `tcpdump`, `wireshark`
 - ▶ Wszystkie systemy UNIX: specyficzne sniffery ukierunkowane na zbieranie hasła z nawiązywanych połączeń FTP, POP3 i innych protokołów.
- ▶ Wiele z nich można znaleźć pod adresem <http://freshmeat.net/> lub <http://sourceforge.net/>
- ▶ Niektóre nazwy: `snort`, `IPgrab`, `ettercap`, `One Way Network SNiffer`, etc.

Przykładowy wynik działania:

```
49 asic ts/pub/src# snoop cyber
   asic -> cyber    TELNET C port=53218
   cyber -> asic     TELNET R port=53218 login:
   asic -> cyber    TELNET C port=53218
   asic -> cyber    TELNET C port=53218 t
   cyber -> asic     TELNET R port=53218 t
   asic -> cyber    TELNET C port=53218
   asic -> cyber    TELNET C port=53218 s
   cyber -> asic     TELNET R port=53218 s
   asic -> cyber    TELNET C port=53218
   cyber -> asic     TELNET R port=53218 s/key 90
                        cy11009\r\n
   asic -> cyber    TELNET C port=53218
   cyber -> asic     TELNET R port=53218 PASSCODE
                        or Password
   asic -> cyber    TELNET C port=53218
   asic -> cyber    TELNET C port=53218 a
   cyber -> asic     TELNET R port=53218
   asic -> cyber    TELNET C port=53218 b
   cyber -> asic     TELNET R port=53218
   asic -> cyber    TELNET C port=53218 c
   cyber -> asic     TELNET R port=53218
   asic -> cyber    TELNET C port=53218
```

Sniffery

- ▶ Mogą być zainstalowane praktycznie wszędzie;
- ▶ Duże sieci o płaskiej topologii dodatkowo ułatwiają podsłuchiwanie pakietów;
- ▶ Switchy i routery pozwalają zredukować ruch w sieci lokalnej, jednocześnie zmniejszając możliwości podsłuchiwania; W dalszym ciągu jednak istnieje możliwość modyfikacji ich działania przez routing dynamiczny lub dynamiczny ARP.
- ▶ Karta sieciowa musi zostać ustawiona w tryb *promiscuous*, pozwalający na odbieranie wszystkich pakietów nadchodzących z sieci;
- ▶ Bardzo trudne lub niemożliwe do wykrycia zdalnie;
- ▶ W sieci Internet można znaleźć wiele skryptów i programów automatycznie zbierających i sortujących dane pochodzące z interfejsów sieciowych.
- ▶ mogą być używane także we „właściwy” sposób, by zbierać statystyki działania sieci lub do oceny charakterystyki ruchu.

- ▶ Komputery znajdujące się w sieci lokalnej z reguły darzone są większym zaufaniem niż pozostałe.
- ▶ Dostęp do niektórych usług oparty jest często na nazwach łączących się komputerów, np. weryfikacji, czy należą do „lokalnej” domeny.
- ▶ W celu znalezienia tłumaczenia nazwa – adres IP używany jest system DNS (mapy „zwykłe” i „odwrotne”).
- ▶ Mapy „odwrotne” związane są z adresami IP i należą do właścicieli odpowiednich klas adresowych.
- ▶ Nie ma sposobu, by powstrzymać kogoś przed rozgłaszaniem fałszywych informacji, na przykład:

```
1.3.0.63.in-addr.arpa.      IN PTR sun1000.pwr.wroc.pl.
```

- ▶ Jak w takim przypadku zachowa się komputer, którego użytkownik dopisał swoje konto i nazwę „sun1000.pwr.wroc.pl” do pliku `~/.rhosts` albo odpowiednich skryptów IRC, dających na podstawie nazwy IP dodatkowe uprawnienia?
- ▶ Jedynym sposobem weryfikacji tej informacji jest sprawdzenie „zwykłej” mapy:

```
sun1000.pwr.wroc.pl      IN A    156.17.1.33
                          IN A    156.17.250.100
```

- ▶ Jeśli adres użyty do tłumaczenia IP-nazwa nie zostanie znaleziony wśród adresów uzyskanych po przetłumaczeniu nazwy na IP – ktoś się bawi w spoofing DNS.

- ▶ Warstwa sieciowa IP nie gwarantuje żadnej poufności danych (ani szyfrowania). W IPv4 wszystkie dane przesyłane są „otwartym tekstem”. Szyfrowanie i kontrola integralności danych muszą być implementowane w wyższych warstwach, jeśli są konieczne.
- ▶ Dane przesyłane siecią komputerową mogą zostać podsłuchane. Skoro przesyłamy dane otwartym tekstem, a pakiety przechodzą przez wiele routerów i sieci, to we wszystkich tych miejscach może znajdować się ktoś, kto podsłuchuje nasze pakiety.
- ▶ Często musimy zaufać danym uzyskiwanym z serwerów znajdujących się poza naszą kontrolą. Wiele informacji o kluczowym znaczeniu (np. nazwy hostów zwracane przez DNS) jest zwracane przez serwery znajdujące się poza kontrolą lokalnego administratora, a więc niekoniecznie godnych zaufania.
- ▶ IPv4 i protokoły sieciowe projektowano z reguły nie biorąc pod uwagę zagrożeń bezpieczeństwa. Pakiety IP mogą być fałszowane, przechwytywane, modyfikowane i przekierowywane, podsłuchiwane. Można się też z ich pomocą podszywać pod inne urządzenia sieciowe lub hosty.
- ▶ Wiele protokołów projektowanych z myślą o poprawieniu bezpieczeństwa jest tak naprawdę tylko obejściem problemu, a nie jego prawdziwym rozwiązaniem, które często jest niemożliwe.
- ▶ Największa zaleta sieci IP – nieograniczona enkapsulacja warstw oprogramowania i elastyczność stosowania poszczególnych rozwiązań – są także najsłabszymi elementami, jeśli chodzi o bezpieczeństwo.
- ▶ Rozszerzenia protokołów związane z bezpieczeństwem muszą być wprowadzane jako osobne warstwy protokołów i dopiero po powszechnej akceptacji (do tego czasu – eksperymentalnie, tak jak np. SSL, IPSec)

- ▶ Mechanizm ochrony bezpieczeństwa rozgraniczający użytkowników i administratora systemu
- ▶ Podział na pracę w trybie użytkownika i w trybie jądra systemu (wspomagany mechanizmami ochrony procesora)
- ▶ W trybie jądra wykonują się wszystkie funkcje systemowe
- ▶ Przejście do trybu jądra realizowane przez przerwania programowe
- ▶ System sterowany zdarzeniami (a nie „polling”) – aplikacje nie tracą czasu, oddając go innym procesom, a same są usypiane, biernie czekając na wystąpienie oczekiwanego zdarzenia
- ▶ Wszystkie¹ urządzenia, mechanizmy komunikacji wewnętrznej i sieciowej itp. dostępne są przez pliki i deskryptory plików
- ▶ System plików umożliwiający kontrolę dostępu do plików i blokowanie prawa do zapisu pomiędzy procesami.
- ▶ Pamięć wirtualna i system plików stosują te same mechanizmy – pełna integracja.
- ▶ Separacja procesów, ochrona procesów przed wzajemnym zakłócaniem sobie pracy, ściśle zdefiniowane mechanizmy komunikacji.

¹prawie. Wyjątek: kolejki komunikatów, semafony

Procesy i prawa dostępu

- ▶ W systemie wielozadaniowym może być wykonywane wiele zadań (procesów) jednocześnie.
- ▶ Każdy proces może być wykonywany przez innego użytkownika systemu.
- ▶ Procesy nie powinny „przeszkadzać sobie” nawzajem.
- ▶ Konieczna jest ochrona procesów przed sobą nawzajem:
 - ▶ Ochrona procesów różnych użytkowników przed usunięciem czy modyfikacją ich danych.
 - ▶ Ochrona procesów tego samego użytkownika – przed modyfikacją danych i kontekstu.
 - ▶ Ochrona dostępu do prywatnych danych procesu.
 - ▶ Ochrona dostępu do wspólnych zasobów.
- ▶ Ściśle określone metody komunikacji międzyprocesowej
 - ▶ Wspólna pamięć w obrębie wątków (zmienne globalne i wskaźniki).
 - ▶ Procesy – pełna separacja pamięci, pamięć wirtualna, `fork()`.
 - ▶ Segmenty pamięci wspólnej (mechanizm IPC `shmget()`), pliki mapowane w pamięci funkcją `mmap()`
 - ▶ Komunikacja za pomocą strumieni pipe, FIFO, gniazdek UNIX i gniazdek sieciowych
 - ▶ Kontrola procesów (sygnały, funkcja `kill()` w obrębie grupy procesów i w ramach tego samego `uid`).

Zmiany euid w programach

- ▶ Programy typu set-user-id w szczególny sposób muszą sprawdzać prawa dostępu do plików, urządzeń itp.
- ▶ Najlepsza metoda – korzystać ze specjalnych uprawnień tylko wtedy, gdy jest to niezbędne i zrezygnować z nich kiedy tylko się da.
- ▶ Przykład serwera pocztowego:
 - ▶ Startuje jako program suid
 - ▶ Zmienia euid na nobody, by przeczytać plik konfiguracyjny.
 - ▶ Zmienia euid na 0, by zacząć nasłuchiwać na porcie 25 (lub innym, określonym w konfiguracji)
 - ▶ Zmienia euid na nobody i działa dalej
 - ▶ Przyjmuje pocztę, wysyła dalej
 - ▶ Zapis poczty do kolejki wymaga zmiany grupy na mail, po zapisaniu można z tej grupy zrezygnować.
 - ▶ Zapis poczty do skrzynki użytkownika wymaga ponownej zmiany euid – na root lub (znacznie bezpieczniej) – ID użytkownika
 - ▶ Po zapisaniu poczty do skrzynki – powrót do ID nobody
- ▶ Trzy niezależne wartości UID:
 - ▶ euid – effective uid (efektywny numer użytkownika)
 - ▶ ruid – real uid (rzeczywisty numer użytkownika)
 - ▶ suid – saved uid (zachowany numer użytkownika)

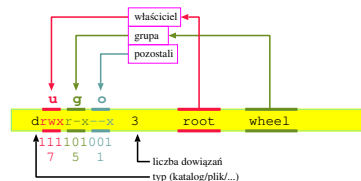
Funkcje mające wpływ na prawa dostępu

- ▶ Zmiana numeru użytkownika:
 - ▶ `getuid()`
 - ▶ `setuid()`
 - ▶ `geteuid()`
 - ▶ `seteuid()`
 - ▶ `setreuid()`
- ▶ Zmiana grupy:
 - ▶ `getgid()`
 - ▶ `setgid()`
 - ▶ `getegid()`
 - ▶ `setegid()`
 - ▶ `setregid()`
- ▶ Jeśli `ruuid` nie jest zmieniany, proces może wielokrotnie zmieniać `euid` między `ruuid` a `saved uid`
- ▶ W przeciwnym razie² nowy `euid` jest kopiowany do `saved uid`, a dokonana zmiana jest ostateczna (i pozwala na całkowite pozbycie się dodatkowych uprawnień do końca działania procesu).

²jednoczesna zmiana `ruuid` i `euid` za pomocą `setreuid()`

▶ Uprawnienia użytkowników w systemie

- ▶ „Zwykły” użytkownik i administrator systemu
- ▶ UID i efektywny UID (**eu**id)
- ▶ Uprawnienia procesów związane z **uid/euid**
- ▶ Procesy set-user-id i set-group-id
- ▶ Kontrola dostępu/uprawnień w dużej mierze odbywa się poprzez system plików (programy, urządzenia specjalne, mechanizmy komunikacji)



▶ Aplikacje systemowe

- ▶ „demony” systemowe (system daemons)
- ▶ serwery sieciowe
- ▶ z reguły działają z uprawnieniami użytkownika **root**
- ▶ błędy w oprogramowaniu mogą prowadzić do eskalacji uprawnień
- ▶ „dziury bezpieczeństwa” – nieprzewidziane przez twórców działanie programu prowadzące do eskalacji uprawnień lub innych naruszeń bezpieczeństwa

▶ „Nietypowe” metody lub błędy konfiguracji systemu

- ▶ Dostęp przez urządzenia specjalne – np. zmodyfikowane prawa dostępu do **/dev/kmem** lub bezpośrednio do urządzeń dyskowych (np. **/dev/sda** itp.)
- ▶ Pseudo-system plików **/proc**

Eskalacja uprawnień – przykłady

- ▶ Wykonanie zewnętrznego programu z uprawnieniami *root*
 - ▶ *sendmail* v.8.8.5
 - ▶ możliwość wystartowania przez dowolnego użytkownika
 - ▶ brak odłączenia się od grupy procesów
 - ▶ „standardowy” sposób przeładowywania programu przez *kill -HUP*
 - ▶ *Apache* – skrypty CGI
 - ▶ Błędy typu „code injection” (np. *bind* v.4.x – buffer overflows)
- ▶ Wykorzystanie uprawnień istniejącego programu systemowego
 - ▶ błędna (celowo?) konfiguracja
 - ▶ nieprawidłowe dane wejściowe
 - ▶ błędy w sposobie dostępu do plików tymczasowych – wyścig (race condition)
 - ▶ błędy systemowe sprzętu (np. błędy obsługi cache procesora – błędy typu Spectre)