# Bezpieczeństwo w sieciach

## Prezentacja na systemy operacyjne

Michał Raczkowski

styczeń 2007

# Możliwości podsłuchiwania - programy podsłuchujące

**programy podsłuchujące (sniffery)** - są to programy, które przechwytują i analizują ruch sieciowy.
Parę przykładów:

# Możliwości podsłuchiwania - programy podsłuchujące

**programy podsłuchujące (sniffery)** - są to programy, które przechwytują i analizują ruch sieciowy.

Parę przykładów:

# Możliwości podsłuchiwania - programy podsłuchujące

**programy podsłuchujące (sniffery)** - są to programy, które przechwytują i analizują ruch sieciowy.
Parę przykładów:

# Programy podsłuchujące - tcpdump

**tcpdump** - bardzo ogólne konsolowe narzędzie do przechwytywania ruchu na interfejsie sieciowym. Domyślnie podgląda nagłówki pakietów przechodzących przez dany interfejs. Mało przyjazne użytkownikowi, ale ***DA SIĘ*** nawet posłuchiwać rozmowy **gg** w **ASCII :)**.

# Programy podsłuchujące - tcpdump

**tcpdump** - bardzo ogólne konsolowe narzędzie do przechwytywania ruchu na interfejsie sieciowym.
Domyślnie podgląda nagłówki pakietów przechodzących przez dany interfejs. Mało przyjazne użytkownikowi, ale ***DA SIĘ*** nawet posłuchiwać rozmowy **gg** w **ASCII :)**.

# Programy podsłuchujące - tcpdump

**tcpdump** - bardzo ogólne konsolowe narzędzie do przechwytywania ruchu na interfejsie sieciowym. Domyślnie podgląda nagłówki pakietów przechodzących przez dany interfejs. Mało przyjazne użytkownikowi, ale ***DA SIĘ*** nawet posłuchiwać rozmowy **gg** w **ASCII :)**.

# Programy podsłuchujące - tcpdump

**tcpdump** - bardzo ogólne konsolowe narzędzie do przechwytywania ruchu na interfejsie sieciowym. Domyślnie podgląda nagłówki pakietów przechodzących przez dany interfejs. Mało przyjazne użytkownikowi, ale ***DA SIĘ*** nawet posłuchiwać rozmowy **gg** w **ASCII :)**.

# Programy podsłuchujące - wireshark

**wireshark** - podobny do **tcpdump-a**, posiada **GUI** i sporo więcej opcji filtrowania i sortowania. Można w nim śledzić strumienie **TCP**, czyli podgladać cała komunikację hostów tekstowo, zamiast gąszczu pakietów z masą nagłówków. przechodząca przez podsłuchiwany interfejs sieciowy. Umożliwia podglądanie całego ruchu w trybie **promiscuous**.

# Programy podsłuchujące - wireshark

**wireshark** - podobny do **tcpdump-a**, posiada **GUI** i sporo więcej opcji filtrowania i sortowania. Można w nim śledzić strumienie **TCP**, czyli podgladać cała komunikację hostów tekstowo, zamiast gąszczu pakietów z masą nagłówków. przechodząca przez podsłuchiwany interfejs sieciowy. Umożliwia podglądanie całego ruchu w trybie **promiscuous**.

# Programy podsłuchujące - wireshark

**wireshark** - podobny do **tcpdump-a**, posiada **GUI** i sporo więcej opcji filtrowania i sortowania. Można w nim śledzić strumienie **TCP**, czyli podgladać cała komunikację hostów tekstowo, zamiast gąszczu pakietów z masą nagłówków. przechodząca przez podsłuchiwany interfejs sieciowy. Umożliwia podglądanie całego ruchu w trybie **promiscuous**.

# Programy podsłuchujące - wireshark

**wireshark** - podobny do **tcpdump-a**, posiada **GUI** i sporo więcej opcji filtrowania i sortowania. Można w nim śledzić strumienie **TCP**, czyli podgladać cała komunikację hostów tekstowo, zamiast gąszczu pakietów z masą nagłówków. przechodząca przez podsłuchiwany interfejs sieciowy. Umożliwia podglądanie całego ruchu w trybie **promiscuous**.

# Możliwości podsłuchiwania - tryb promiscuous

**tryb promiscuous** - konfiguracja interfejsu sieciowego (karty sieciowej), w której przekazuje ona cały ruch do hosta, a nie tylko pakiety do niego zaadresowane. Np. w sieciach z **HUB-ami** można podglądać cały ruch.

# Możliwości podsłuchiwania - tryb promiscuous

**tryb promiscuous** - konfiguracja interfejsu sieciowego (karty sieciowej), w której przekazuje ona cały ruch do hosta, a nie tylko pakiety do niego zaadresowane. Np. w sieciach z **HUB-ami** można podglądać cały ruch.

# Możliwości podsłuchiwania - tryb promiscuous

**tryb promiscuous** - konfiguracja interfejsu sieciowego (karty sieciowej), w której przekazuje ona cały ruch do hosta, a nie tylko pakiety do niego zaadresowane. Np. w sieciach z **HUB-ami** można podglądać cały ruch.

# Ogólnie o dsniff-ie

Pakiet programów do dogłębnej penetracji i podsłuchu w sieci lokalnej. Jest zbiorem dość luźno ze sobą związanych programów do testowania zabezpieczeń lub innych niecnych celów. Zawiera też opcje do podszywania się i podsłuch sesji **SSH** i **SSL**.

# Ogólnie o dsniff-ie

Pakiet programów do dogłębnej penetracji i podsłuchu w sieci lokalnej. Jest zbiorem dość luźno ze sobą związanych programów do testowania zabezpieczeń lub innych niecnych celów. Zawiera też opcje do podszywania się i podsłuch sesji **SSH** i **SSL**.

# Skład dsniff-a - arpspoof

**arpspoof** - przez fakowe rozgłoszenia **ARP** powoduje, że komputer ofiara wysyła do intruza wszystkie pakiety, zamiast do zadanego **IP**. Działa to tak, że jest podmieniany adres fizyczny z prawdziwego na adres intruza dla pewnego konkretnego **IP**.

# Skład dsniff-a - arpspoof

**arpspoof** - przez fakowe rozgłoszenia **ARP** powoduje, że komputer ofiara wysyła do intruza wszystkie pakiety, zamiast do zadanego **IP**. Działa to tak, że jest podmieniany adres fizyczny z prawdziwego na adres intruza dla pewnego konkretnego **IP**.

# Skład dsniff-a - arpspoof

**arpspoof** - przez fakowe rozgłoszenia **ARP** powoduje, że komputer ofiara wysyła do intruza wszystkie pakiety, zamiast do zadanego **IP**. Działa to tak, że jest podmieniany adres fizyczny z prawdziwego na adres intruza dla pewnego konkretnego **IP**.

# Skład dsniff-a - dnsspoof

**dnsspoof** - Wysyła fakowe odpowiedzi na zapytania DNS.

# Skład dsniff-a - dnsspoof

**dnsspoof** - Wysyła fakowe odpowiedzi na zapytania **DNS**.

# Skład dsniff-a - dsniff

**dsniff** - sniffer interpretujący wiele popularnych protokołów, takich jak **FTP**, **Telnet**, **SMTP**, **HTTP**, **POP3** i dosłownie wiele innych. Wygodny do przechwytywania prywatnych danych, np. haseł.

# Skład dsniff-a - dsniff

**dsniff** - sniffer interpretujący wiele popularnych protokołów, takich jak **FTP**, **Telnet**, **SMTP**, **HTTP**, **POP3** i dosłownie wiele innych. Wygodny do przechwytywania prywatnych danych, np. haseł.

# Skład dsniff-a - dsniff

**dsniff** - sniffer interpretujący wiele popularnych protokołów, takich jak **FTP**, **Telnet**, **SMTP**, **HTTP**, **POP3** i dosłownie wiele innych. Wygodny do przechwytywania prywatnych danych, np. haseł.

# Skład dsniff-a - filesnarf

**filesnarf** - potrafi zapisywać pliki przesyłane **NFS-em**.

# Skład dsniff-a - filesnarf

**filesnarf** - potrafi zapisywać pliki przesyłane **NFS-em**.

# Skład dsniff-a - macof

**macof** - przepełnia pamięć **switch-a** tak, że **switch** zaczyna działac jak **HUB**, co może ułatwić sniffowanie.

# Skład dsniff-a - macof

**macof** - przepełnia pamięć **switch-a** tak, że **switch** zaczyna działac jak **HUB**, co może ułatwić sniffowanie.

# Skład dsniff-a - mailsnarf

**mailsnarf** - pogląda **email-e** przez analizowanie podglądanych protokołów **POP3** i **SMTP**.

# Skład dsniff-a - mailsnarf

**mailsnarf** - pogląda **email-e** przez analizowanie podglądanych protokołów **POP3** i **SMTP**.

# Skład dsniff-a - msgsnarf

**msgsnarf** - bardzo efektowne narzędzie przechwytujące wiadomości przesyłane przez **IRC-a** oraz komunikatory internetowe.

# Skład dsniff-a - msgsnarf

**msgsnarf** - bardzo efektowne narzędzie przechwytujące wiadomości przesyłane przez **IRC-a** oraz komunikatory internetowe.

# Skład dsniff-a - webspy

**webspy** - chyba najbardziej widowiskowe z całego pakietu: wysniffowane z protokołu **HTTP URL-e** przekazuje do lokalnej przeglądarki i na bieżąco wyświetla stony, które odwiedza host podglądany **:)**.

# Skład dsniff-a - webspy

**webspy** - chyba najbardziej widowiskowe z całego pakietu: wysniffowane z protokołu **HTTP URL-e** przekazuje do lokalnej przeglądarki i na bieżąco wyświetla stony, które odwiedza host podglądany **:)**.

# Skład dsniff-a - webspy

**webspy** - chyba najbardziej widowiskowe z całego pakietu: wysniffowane z protokołu **HTTP URL-e** przekazuje do lokalnej przeglądarki i na bieżąco wyświetla stony, które odwiedza host podglądany **:)**.

# Skład dsniff-a - krótko pozostałe

- **tcpkill** - jak nazwa wskazuje, zabija sesję **TCP**.
- **tcpnice** - zmniejsza szybkość połączenia **TCP**.
- **urlsnarf** - sniffuje **HTTP** w poszukiwaniu **URL-i**.
- **sshmitm** - serwer pośredniczący połączeniom **SSH**, umożliwia ich sniffowanie.
- **webmitm** - program sniffujący jako serwer pośredniczący dla połączeń **HTTP** i **HTTPS** przekierowanych przez **dnsspoof**.

# Skład dsniff-a - krótko pozostałe

- **tcpkill** - jak nazwa wskazuje, zabija sesję **TCP**.
- **tcpnice** - zmniejsza szybkość połączenia **TCP**.
- **urlsnarf** - sniffuje **HTTP** w poszukiwaniu **URL-i**.
- **sshmitm** - serwer pośredniczący połączeniom **SSH**, umożliwia ich sniffowanie.
- **webmitm** - program sniffujący jako serwer pośredniczący dla połączeń **HTTP** i **HTTPS** przekierowanych przez **dnsspoof**.

# Skład dsniff-a - krótko pozostałe

- **tcpkill** - jak nazwa wskazuje, zabija sesję **TCP**.
- **tcpnice** - zmniejsza szybkość połączenia **TCP**.
- **urlsnarf** - sniffuje **HTTP** w poszukiwaniu **URL-i**.
- **sshmitm** - serwer pośredniczący połączeniom **SSH**, umożliwia ich sniffowanie.
- **webmitm** - program sniffujący jako serwer pośredniczący dla połączeń **HTTP** i **HTTPS** przekierowanych przez **dnsspoof**.

# Skład dsniff-a - krótko pozostałe

- **tcpkill** - jak nazwa wskazuje, zabija sesję **TCP**.
- **tcpnice** - zmniejsza szybkość połączenia **TCP**.
- **urlsnarf** - sniffuje **HTTP** w poszukiwaniu **URL-i**.
- **sshmitm** - serwer pośredniczący połączeniom **SSH**, umożliwia ich sniffowanie.
- **webmitm** - program sniffujący jako serwer pośredniczący dla połączeń **HTTP** i **HTTPS** przekierowanych przez **dnsspoof**.

# Skład dsniff-a - krótko pozostałe

- **tcpkill** - jak nazwa wskazuje, zabija sesję **TCP**.
- **tcpnice** - zmniejsza szybkość połączenia **TCP**.
- **urlsnarf** - sniffuje **HTTP** w poszukiwaniu **URL-i**.
- **sshmitm** - serwer pośredniczący połączeniom **SSH**, umożliwia ich sniffowanie.
- **webmitm** - program sniffujący jako serwer pośredniczący dla połączeń **HTTP** i **HTTPS** przekierowanych przez **dnsspoof**.

# Skład dsniff-a - krótko pozostałe

- **tcpkill** - jak nazwa wskazuje, zabija sesję **TCP**.
- **tcpnice** - zmniejsza szybkość połączenia **TCP**.
- **urlsnarf** - sniffuje **HTTP** w poszukiwaniu **URL-i**.
- **sshmitm** - serwer pośredniczący połączeniom **SSH**, umożliwia ich sniffowanie.
- **webmitm** - program sniffujący jako serwer pośredniczący dla połączeń **HTTP** i **HTTPS** przekierowanych przez **dnsspoof**.

# Sposoby szyfrowania

**szyfrowanie** - jest chyba jedynym sensownym sposobem zachowania poufności ruchu sieciowego. Intruzi mogą odczytać zawartość pakietów, jednak są one bezużyteczne bez klucza.
Przykłady mechanizmów:

# Sposoby szyfrowania

**szyfrowanie** - jest chyba jedynym sensownym sposobem zachowania poufności ruchu sieciowego. Intruzi mogą odczytać zawartość pakietów, jednak są one bezużyteczne bez klucza.
Przykłady mechanizmów:

# Sposoby szyfrowania

**szyfrowanie** - jest chyba jedynym sensownym sposobem zachowania poufności ruchu sieciowego. Intruzi mogą odczytać zawartość pakietów, jednak są one bezużyteczne bez klucza.

Przykłady mechanizmów:

# Sposoby szyfrowania

**szyfrowanie** - jest chyba jedynym sensownym sposobem zachowania poufności ruchu sieciowego. Intruzi mogą odczytać zawartość pakietów, jednak są one bezużyteczne bez klucza.
Przykłady mechanizmów:

# Mechanizmy szyfrowania - SSH

**SSH** - (*secure shell*) to standard szyfrowanych protokołów komunikacyjnych używanych w sieciach **TCP-IP**, w architekturze klient-serwer. Jest nastepcą nieszyfrowanego protokołu **Telnet**, służy do terminalowego łączenia się ze zdalnym komputerem. **SCP** oraz **SFTP** z rodziny **SSH** są powszechnie używane do przesyłania plików. Wspólną cechą tych protokołów jest technika szyfrowania danych i rozpoznawania użytkownika.

# Mechanizmy szyfrowania - SSH

**SSH** - (*secure shell*) to standard szyfrowanych protokołów komunikacyjnych używanych w sieciach **TCP-IP**, w architekturze klient-serwer. Jest nastepcą nieszyfrowanego protokołu **Telnet**, służy do terminalowego łączenia się ze zdalnym komputerem. **SCP** oraz **SFTP** z rodziny **SSH** są powszechnie używane do przesyłania plików. Wspólną cechą tych protokołów jest technika szyfrowania danych i rozpoznawania użytkownika.

# Mechanizmy szyfrowania - SSH

**SSH** - (*secure shell*) to standard szyfrowanych protokołów komunikacyjnych używanych w sieciach **TCP-IP**, w architekturze klient-serwer. Jest nastepcą nieszyfrowanego protokołu **Telnet**, służy do terminalowego łączenia się ze zdalnym komputerem. **SCP** oraz **SFTP** z rodziny **SSH** są powszechnie używane do przesyłania plików. Wspólną cechą tych protokołów jest technika szyfrowania danych i rozpoznawania użytkownika.

# Mechanizmy szyfrowania - SSH

**SSH** - (*secure shell*) to standard szyfrowanych protokołów komunikacyjnych używanych w sieciach **TCP-IP**, w architekturze klient-serwer. Jest nastepcą nieszyfrowanego protokołu **Telnet**, służy do terminalowego łączenia się ze zdalnym komputerem. **SCP** oraz **SFTP** z rodziny **SSH** są powszechnie używane do przesyłania plików. Wspólną cechą tych protokołów jest technika szyfrowania danych i rozpoznawania użytkownika.

# Mechanizmy szyfrowania - TLS

**TLS** - (*Transport Layer Security*) to internetowy standard, rozwinięcie protokołu **SSL** (*Secure Socket Layer*). Ma na celu zapewnienie poufności i integralności transmisji danych oraz zapewnienie uwierzytelnienia, stosuje szyfry asymetryczne oraz certyfikaty standardu **X.509**. Zaletą protokołu jest fakt, że działa on na warstwie **TCP**, więc można go łatwo zastosować do zabezpieczenia protokołów warstwy aplikacyjnej (np.: **HTTP**, **POP3**).

# Mechanizmy szyfrowania - TLS

**TLS** - (*Transport Layer Security*) to internetowy standard, rozwinięcie protokołu **SSL** (*Secure Socket Layer*). Ma na celu zapewnienie poufności i integralności transmisji danych oraz zapewnienie uwierzytelnienia, stosuje szyfry asymetryczne oraz certyfikaty standardu **X.509**. Zaletą protokołu jest fakt, że działa on na warstwie **TCP**, więc można go łatwo zastosować do zabezpieczenia protokołów warstwy aplikacyjnej (np.: **HTTP**, **POP3**).

# Mechanizmy szyfrowania - TLS

**TLS** - (*Transport Layer Security*) to internetowy standard, rozwinięcie protokołu **SSL** (*Secure Socket Layer*). Ma na celu zapewnienie poufności i integralności transmisji danych oraz zapewnienie uwierzytelnienia, stosuje szyfry asymetryczne oraz certyfikaty standardu **X.509**. Zaletą protokołu jest fakt, że działa on na warstwie **TCP**, więc można go łatwo zastosować do zabezpieczenia protokołów warstwy aplikacyjnej (np.: **HTTP**, **POP3**).

# Warstwa SSL


Rysunek: Schemat warstw sieciowych

# Kontrola

oczywiście należy we własnym zakresie kontrolować sytuację.

- np. warto wiedzieć, czy adres **IP** na który wysyłamy odpowiada hostowi, któremu powinien. Zeby uchronić się przed **arpspoof-em** należy kontrolować **ARP**, np. dobrym pomysłem jest podanie na sztywno adresu **MAC** bramy internetowej.
- również nie należy podawać istotnych haseł w nieszyfrowanych protokołach np. **Telnet**, **FTP**, **HTTP** i.t.p. oraz nie pozwalać na nieszyfrowane połaczenia ze swoim hostem, jeśli klient może być potencjalnie niebezpieczny.
- stosowanie **firewall-a**

# Kontrola

oczywiście należy we własnym zakresie kontrolować sytuację.

- np. warto wiedzieć, czy adres **IP** na który wysyłamy odpowiada hostowi, któremu powinien. Zeby uchronić się przed **arpspoof-em** należy kontrolować **ARP**, np. dobrym pomysłem jest podanie na sztywno adresu **MAC** bramy internetowej.
- również nie należy podawać istotnych haseł w nieszyfrowanych protokołach np. **Telnet**, **FTP**, **HTTP** i.t.p. oraz nie pozwalać na nieszyfrowane połaczenia ze swoim hostem, jeśli klient może być potencjalnie niebezpieczny.
- stosowanie **firewall-a**

# Kontrola

oczywiście należy we własnym zakresie kontrolować sytuację.

- np. warto wiedzieć, czy adres **IP** na który wysyłamy odpowiada hostowi, któremu powinien. Zeby uchronić się przed **arpspoof-em** należy kontrolować **ARP**, np. dobrym pomysłem jest podanie na sztywno adresu **MAC** bramy internetowej.
- również nie należy podawać istotnych haseł w nieszyfrowanych protokołach np. **Telnet**, **FTP**, **HTTP** i.t.p. oraz nie pozwalać na nieszyfrowane połaczenia ze swoim hostem, jeśli klient może być potencjalnie niebezpieczny.
- stosowanie **firewall-a**

# Kontrola

oczywiście należy we własnym zakresie kontrolować sytuację.

- np. warto wiedzieć, czy adres **IP** na który wysyłamy odpowiada hostowi, któremu powinien. Zeby uchronić się przed **arpspoof-em** należy kontrolować **ARP**, np. dobrym pomysłem jest podanie na sztywno adresu **MAC** bramy internetowej.
- również nie należy podawać istotnych haseł w nieszyfrowanych protokołach np. **Telnet**, **FTP**, **HTTP** i.t.p. oraz nie pozwalać na nieszyfrowane połaczenia ze swoim hostem, jeśli klient może być potencjalnie niebezpieczny.
- stosowanie **firewall-a**

# Kontrola

oczywiście należy we własnym zakresie kontrolować sytuację.

- np. warto wiedzieć, czy adres **IP** na który wysyłamy odpowiada hostowi, któremu powinien. Zeby uchronić się przed **arpspoof-em** należy kontrolować **ARP**, np. dobrym pomysłem jest podanie na sztywno adresu **MAC** bramy internetowej.
- również nie należy podawać istotnych haseł w nieszyfrowanych protokołach np. **Telnet**, **FTP**, **HTTP** i.t.p. oraz nie pozwalać na nieszyfrowane połaczenia ze swoim hostem, jeśli klient może być potencjalnie niebezpieczny.
- stosowanie **firewall-a**

# Bibliografia


- http://en.wikipedia.org
- http://pl.wikipedia.org
- http://www.dyskretny.pl/site.php?id=informatyka

# Koniec

Dziękuję za uwagę.