Cisza w sieci

Autor: Michał Zalewski
Tłumaczenie: Zbigniew Banach
ISBN: 83-7361-659-4
Tytuł oryginału: Silence on the Wire
Format: B5, stron: 304

Praktyczne spojrzenie na zagadnienia bezpieczeństwa w sieci

- Poznaj zasady działania protokołów sieciowych
- Naucz się rozpoznawać zagrożenia
- Zastosuj techniki obronne

W Internecie zdrowy rozsądek i zasady moralne, które obowiązują w rzeczywistym świecie, tracą rację bytu. Z racji coraz głośniejszych i coraz częstszych kradzieży danych i włamań do komputerów rozsądek zostaje zastąpiony paranoją i obawą, a komputerowi przestępcy rzadko miewają wyrzuty sumienia. Bezpieczeństwa w sieci nie zapewnimy sobie, nie popełniając błędów czy też postępując w określony sposób. Prawie z każdym procesem informacyjnym wiążą się zagadnienia bezpieczeństwa, które należy zrozumieć.

„Cisza w sieci. Praktyczny przewodnik po pasywnym rozpoznaniu i atakach pośrednich" to bardzo nietypowa książka poświęcona technikom ochrony danych. Autor przedstawia w niej zupełnie inne spojrzenie na bezpieczeństwo. Pokazuje niezwykłe i niecodzienne zagadnienia ochrony danych, które nie mieszczą się w ramach tradycyjnego modelu haker – ofiara. Z tej książki dowiesz się o rzeczach, których istnienia nawet nie podejrzewałeś, na przykład o tym, że generator liczb losowych może ujawniać naciskane przez Ciebie klawisze, a postronny obserwator może zidentyfikować Twój system operacyjny, wyłącznie analizując pakiety sieciowe. Nauczysz się rozpoznawać takie zagrożenia i przeciwdziałać im.

- Bezpieczeństwo generatorów liczb losowych
- Ataki na sieci przełączane
- Działanie protokołu IP
- Pasywna identyfikacja systemów na podstawie pakietów IP i jej zapobieganie
- Właściwe stosowanie firewalli
- Techniki skanowania portów
- Identyfikacja użytkowników systemów

Spójrz na budowę sieci i pracę z komputerem z zupełnie nowej perspektywy

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Rozdział 11.

Rozpoznawanie anomalii

Czyli czego można się dowiedzieć z drobnych niedoskonałości w pakietach sieciowych.

W poprzednich rozdziałach omówiłem szereg sposobów pobierania cząstek potencjalnie użytecznych informacji z pozornie pozbawionych znaczenia parametrów technicznych towarzyszących każdemu pakietowi danych przesyłanemu przez sieć przez podejrzanego. Mam nadzieję, że udało mi się pokazać sposoby pozyskiwania znacznych ilości danych, których udostępniania nadawca nie jest świadom (a w najlepszym razie jest bardzo niezadowolony z powodu braku możliwości ich nieudostępniania). W idealnym świecie rozliczne metody analizy pakietów i strumieni danych pozwoliłyby nam zmierzyć wiele spośród parametrów zdalnego systemu i przypisać zaobserwowane zachowanie do sygnatury konkretnego systemu operacyjnego i konfiguracji sieci.

Rzeczywistość wygląda jednak nieco inaczej, gdyż wartości niektórych spośród obserwowanych parametrów zawsze przynajmniej trochę odbiegają od zakresu oczekiwanego dla konkretnego urządzenia lub konfiguracji sieci podejrzanego. Można wprawdzie zignorować te z pozoru przypadkowe i pozbawione znaczenia różnice i niezależnie od ich obecności poprawnie identyfikować system źródłowy czy też śledzić jego użytkowników, ale niekoniecznie jest to rozsądne. Na co dzień przywykliśmy ignorować tego typu irytujące drobiazgi, ale nic w informatyce nie dzieje się bez dobrego powodu (zakładając oczywiście dość luźną definicję słowa „dobry"). Zbadanie mechanizmu powstawania tych pozornie losowych anomalii i pomniejszych prawidłowości pozwoli nam uzyskać cenne informacje o niewidocznych dotąd elementach konfiguracji sieci.

W tym rozdziale przyjrzymy się bliżej niektórym procesom mogącym wpływać na obserwowane zachowanie systemu. Postaram się też wyjaśnić podstawową przyczynę, cel (lub bezcelowość) oraz konsekwencje stosowania mechanizmów odpowiedzialnych za to zachowanie.

Większość spośród przedstawionych dalej i dających się odtworzyć modyfikacji pakietów IP pochodzi oczywiście od co bardziej zaawansowanych pośrednich systemów przetwarzających dane IP. Zacznę więc od omówienia dwóch nieco zaniedbanych tematów: firewalli w ogólności i translacji adresów sieciowych (NAT) w szczególności.

Firewalle mają w założeniu być nienaruszalnymi bastionami bezpieczeństwa — w końcu im mniej wiadomo o systemie użytkownika, tym lepiej dla niego. Jednak nawet w przypadku rygorystycznego przestrzegania wszystkich reguł i ustawień wzrost złożoności tych urządzeń i coraz lepsze ich przystosowanie do sprostania współczesnym zagrożeniom sprawiają, że jednocześnie łatwiej je identyfikować za pomocą pośrednich lub pasywnych technik badawczych.

Podstawy firewalli sieciowych

Popularne firewalle [1] są — najprościej rzecz ujmując — pewnego rodzaju routerami, specjalnie zaprojektowanymi tak, by naruszać podstawową zasadę projektową routerów tradycyjnych. Zwykłe routery mają za zadanie podejmować decyzje o trasowaniu pakietów wyłącznie na podstawie informacji dostępnych w trzeciej warstwie modelu OSI, natomiast firewalle interpretują czy wręcz modyfikują dane wyższych warstw (na przykład TCP czy nawet HTTP). Pomimo swego względnie młodego wieku firewalle są powszechnie stosowanym i dobrze rozumianym rozwiązaniem zabezpieczającym, znajdującym miejsce zarówno w sieciach domowych, jak i w wielkich korporacjach. Odpowiednia konfiguracja firewalli pozwala odrzucać, dopuszczać lub przekierowywać określone rodzaje pakietów adresowanych do konkretnych usług, toteż są one stosowane do ograniczania dostępu do wybranych funkcji i zasobów z poziomu wszystkich pakietów przechodzących przez takie urządzenie. Firewalle są więc potężnym, choć niekiedy też przecenianym środkiem bezpieczeństwa sieciowego.

Przyczyną popularności firewalli we wszystkich środowiskach sieciowych jest to, że pozwalają one chronić wiele różnych, złożonych systemów za pomocą pojedynczego, względnie odpornego na ataki elementu oraz stanowią dodatkowe, awaryjne zabezpieczenie na wypadek wystąpienia na chronionym serwerze problemu konfiguracyjnego powodującego ujawnienie podatności pewnej funkcji lub usługi. (W skrajnych przypadkach firewalle są po prostu wykorzystywane do ukrycia złej konfiguracji i braku konserwacji chronionego systemu, najczęściej z katastrofalnymi skutkami).

Filtrowanie bezstanowe a fragmentacja

Proste firewalle są bezstanowymi filtrami pakietów. Ich działanie polega na sprawdzaniu określonych cech każdego pakietu (na przykład portu docelowego w pakietach SYN, stanowiących próby nawiązania połączenia TCP), a następnie podejmowaniu decyzji o ich ewentualnym przepuszczeniu wyłącznie na podstawie odczytanych informacji. Analiza bezstanowa jest bardzo prosta, niezawodna i wiąże się z bardzo niewielkim zużyciem pamięci. Bezstanowy firewall może na przykład ograniczać połączenia przychodzące do serwera poczty wyłącznie do połączeń z portem 25 (czyli

SMTP), po prostu odrzucając wszystkie pakiety SYN adresowane do portów innych niż 25. Nawiązanie połączenia nie jest możliwe bez tego początkowego pakietu SYN, więc napastnik nie ma możliwości sensownej interakcji z aplikacjami na innych portach. Firewall może osiągnąć taki efekt przy znacznie mniejszej złożoności od samego serwera poczty, gdyż nie musi utrzymywać zapisów aktualnie nawiązanych połączeń i ich stanu.

Dyskretna ochrona tego typu ma tę wadę, że firewall i docelowy odbiorca mogą różnie rozumieć niektóre parametry. Napastnik może na przykład przekonać firewall, że łączy się z dozwolonym portem, a pakiety spreparować w taki sposób, by odbiorca odczytał inny port docelowy, tym samym nawiązując połączenie na porcie, który firewall miał chronić. W ten sposób agresor może uzyskać dostęp do podatnej na atak usługi czy nawet interfejsu administracyjnego, co oznacza poważne kłopoty.

Mogłoby się wydawać, że sprowokowanie takiego nieporozumienia jest mało prawdopodobne, jednak w rzeczywistości okazało się to zadaniem dość prostym dzięki pomocy naszego starego znajomego — fragmentacji pakietów. Podejście to nosi nazwę ataku z nachodzeniem fragmentów i zostało opisane w 1995 r. w dokumencie RFC1858 [2]. Atakujący zaczyna od wysłania na dozwolony port ofiary (na przykład wspomniany już port 25) pakietu inicjującego, zawierającego początek żądania SYN dla protokołu TCP. Pakietowi brakuje tylko kawałeczka danych i zawiera on w nagłówku IP znacznik sygnalizujący obecność dalszych fragmentów, ale dlaczego firewall miałby się interesować danymi pakietu?

Firewall bada pakiet i stwierdza, że jest to pakiet SYN o dozwolonym porcie docelowym, więc żądanie zostaje przepuszczone do odbiorcy, który jednak nie interpretuje go od razu (ze względu na omówiony w rozdziale 9. proces składania fragmentów). Pakiet jest przetrzymywany aż do zakończenia składania, czyli do momentu otrzymania ostatniego fragmentu.

Teraz napastnik wysyła drugi fragment pakietu, spreparowany tak, by zachodził na pierwszy pakiet, ale tylko tyle, by w buforze składania fragmentów nadpisać pole nagłówka TCP określające port docelowy. Przygotowany fragment zaczyna się od niezerowego przesunięcia i brakuje mu większości nagłówka TCP, z wyjątkiem części nadpisywanej.

Drugi fragment nie zawiera żadnych danych, na podstawie których firewall mógłby decydować o jego odrzuceniu lub przepuszczeniu (brakuje mu znaczników, typu pakietu i innych niezbędnych informacji), więc firewall bezstanowy najczęściej przepuści go bez ingerencji. Po złożeniu fragmentów pakietu przez odbiorcę drugi fragment nadpisuje oryginalne pole portu docelowego inną wartością, odpowiadającą bardziej niegrzecznemu portowi wybranemu przez napastnika. W rezultacie system odbiorcy nawiązuje połączenie na porcie, który powinien być chroniony przez firewall.

Nie za dobrze.

Starannie zaprojektowany firewall bezstanowy potrafi chronić przed tego typu atakiem, wykonując wstępne składanie pakietów przed ich analizą. Tym samym staje się on jednak nieco mniej bezstanowy i nieco bardziej zauważalny.

Filtrowanie bezstanowe a pakiety niezsynchronizowane

Inną wadą bezstanowych filtrów pakietów jest to, że wcale nie są one tak szczelne, jak moglibyśmy tego chcieć. Filtrowanie jest możliwe tylko wtedy, gdy pojedynczy pakiet zawiera wszystkie informacje niezbędne do podjęcia przez filtr decyzji o jego obsłużeniu. Po wstępnej negocjacji połączenia komunikacja TCP jest w dużej mierze symetryczna i obie strony wymieniają na takich samych prawach takie same dane (pakiety ACK), więc sensowne filtry da się zastosować wyłącznie do fazy negocjacji. Bez śledzenia i rejestrowania połączeń nie da się ustalić, kto (i czy w ogóle ktokolwiek) nawiązał połączenie, w ramach którego są wymieniane pakiety ACK. W praktyce oznacza to, że raczej ciężko jest zdefiniować sensowne reguły postępowania firewalla dla pakietów występujących podczas transmisji, a więc ACK, FIN czy RST.

Niemożność filtrowania pakietów po początkowym SYN nie stanowi na ogół problemu — jeśli napastnikowi nie uda się dostarczyć początkowego pakietu SYN, to nie będzie on w stanie nawiązać połączenia. Jest jednak pewien haczyk: różne systemy różnie reagują na pakiety inne niż SYN kierowane do konkretnego portu, w zależności od tego, czy dany port jest zamknięty, czy też system na nim nasłuchuje. Na przykład niektóre systemy odpowiadają pakietem RST na bezpańskie pakiety FIN, chyba że właśnie na danym porcie nasłuchują — wtedy w ogóle na takie pakiety nie reagują[1].

Oznacza to, że przeciwko bezstanowym filtrom pakietów można stosować techniki skanowania FIN i ACK (ta druga została po raz pierwszy opisana przez Uriela Maimona [3] w magazynie *Phrack*), jak również NUL i Xmas (metody skanowania wykorzystujące odpowiednio pakiety bez żadnych znaczników i ze wszystkimi znacznikami) pozwalające zbierać niezbędne do przeprowadzenia ataku informacje o otwartych portach w systemie zdalnym lub portach chronionych przez firewall. Samo ustalenie, że konkretny port jest otwarty (bez możliwości nawiązania połączenia), nie stanowi poważnego zagrożenia, jednak takie skanowanie często ujawnia niezwykle cenne informacje o wewnętrznej strukturze sieci (na przykład o systemie operacyjnym i aktywnych usługach), ułatwiające przeprowadzenie znacznie skuteczniejszego, wydajniejszego i trudniejszego w wykryciu ataku po przełamaniu lub ominięciu pierwszej linii obrony. Możliwość ta jest powszechnie uważana za wadę firewalli bezstanowych.

Potencjalnie poważniejsze zagrożenie wynika z połączenia filtrowania bezstanowego z mechanizmem podpisów SYN (ang. *SYN cookies*). Służy on do ochrony systemów operacyjnych przed atakami wyczerpania zasobów, polegającymi na wysyłaniu do ofiary dużych ilości fałszywych żądań nawiązania połączenia (co samo w sobie jest operacją bardzo prostą). Zmusza to odbiorcę to wysyłania bezcelowych odpowiedzi SYN+ACK, a w dodatku do alokowania pamięci i innych zasobów niezbędnych dla dodania każdego rzekomego połączenia do tablicy stanów TCP. Zaatakowany w ten

[1] Niektóre aspekty tego zachowania (czyli tendencji do odsyłania RST w odpowiedzi na pakiety kierowane do zamkniętych portów, a ignorowania identycznych pakietów adresowanych do portów, na których system nasłuchuje) wynikają z zaleceń RFC793, a inne są po prostu następstwem decyzji podjętych przez twórców danego systemu.

sposób system najczęściej albo zużywa większość dostępnych zasobów i tym samym pracuje coraz wolniej, albo w pewnym momencie odmawia obsłużenia kolejnych klientów aż do upłynięcia czasu oczekiwania na fałszywe połączenia.

Mechanizm podpisów SYN dostarcza rozwiązanie tego problemu, polegające na odesłaniu odpowiedzi SYN+ACK z podpisem kryptograficznym w polu początkowego numeru sekwencyjnego (dokładniej mówiąc, jest to skrót kryptograficzny, stanowiący jednoznaczny identyfikator połączenia), a następnie zapomnieniu o całej sprawie. Połączenie zostanie dodane do tablicy stanów dopiero wtedy, gdy nadawca odeśle odpowiedź ACK, której numer potwierdzający poprawnie przejdzie stosowaną procedurę kryptograficzną.

Metoda ta ma jednak pewną wadę: dopuszcza możliwość, że pierwotny pakiet SYN i odpowiedź SYN+ACK nie zostały w ogóle wysłane. Gdyby napastnikowi udało się stworzyć podpis ISN, który zostałby pomyślnie zweryfikowany przez algorytm weryfikacji podpisu po stronie odbiorcy (czy to ze względu na bardzo dużą liczbę prób, czy też z powodu słabości samego algorytmu), mógłby on wysłać pakiet ACK powodujący dodanie do tablicy stanów TCP odbiorcy nowego połączenia, choć — jak pamiętamy — żądanie SYN i odpowiedź SYN+ACK nigdy nie zostały wysłane. Bezstanowy firewall nie mógłby wtedy wykryć momentu nawiązania połączenia, gdyż po prostu nie otrzymał takiego żądania. Nie było początkowego pakietu SYN, więc firewall nie mógł sprawdzić jego portu ani odrzucić niebezpiecznego pakietu, a mimo to połączenie zostaje nagle nawiązane.

A to już naprawdę niedobrze.

Stanowe filtry pakietów

Rozwiązanie problemów filtrów bezstanowych wymaga składowania przez firewall niektórych informacji dotyczących dotychczasowego ruchu sieciowego oraz stanu bieżących strumieni danych. Tylko w ten sposób można niezauważalnie dla użytkownika przewidywać wyniki składania fragmentów czy też ustalać kontekst pakietów przesyłanych w ramach połączenia w celu określenia, czy należy je odrzucić jako niedopuszczalne, czy też przepuścić do oczekującego na nie odbiorcy.

Rozwój tanich i wydajnych technik komputerowych pozwolił tworzyć znacznie bardziej złożone i zaawansowane firewalle niż dotychczas. Oznacza to przejście do stanowego śledzenia pakietów, czyli sytuacji, w której firewall nie tylko bada pojedyncze pakiety, ale również pamięta kontekst każdego pakietu w ramach połączenia i sprawdza poprawność pakietu w tym kontekście. Dzięki temu firewall może szczelnie chronić sieć i odrzucać niepożądane lub niespodziewane pakiety bez konieczności polegania na umiejętności odróżniania danych pożądanych od niepożądanych przez odbiorcę. Stanowe filtry pakietów starają się śledzić poszczególne połączenia i dopuszczać do odbiorcy tylko te dane, które faktycznie należą do aktywnych sesji, dzięki czemu zapewniają one znacznie lepszą ochronę i większe możliwości rejestrowania ruchu sieciowego.

Filtrowanie stanowe jest oczywiście zadaniem znacznie trudniejszym od filtrowania bezstanowego, wymaga więc znacznie większych zasobów, zwłaszcza gdy firewall chroni dużą sieć. Ochrona większej ilości systemów wymaga, by firewall dysponował dużą ilością pamięci i szybkim procesorem, co pozwoli mu na bieżąco zapisywać i sprawdzać informacje dotyczące ruchu na kablu.

Stanowa analiza pakietów wiąże się też z większym prawdopodobieństwem występowania problemów i nieporozumień. Kłopoty pojawiają się w sytuacji, gdy firewall i komunikujące się systemy różnie rozumieją bieżący stan sesji TCP/IP, co niestety jest jak najbardziej możliwe, biorąc pod uwagę niejednoznaczność specyfikacji i różnorodność implementacji stosu TCP/IP. Gdyby na przykład firewall stosujący nieco luźniejsze zasady inspekcji numerów sekwencyjnych od końcowego odbiorcy otrzymał pakiet RST niemieszczący się w dopuszczalnym zakresie tych numerów, mógłby stwierdzić, że sesja została zamknięta, podczas gdy odbiorca mógłby traktować sesję jako wciąż otwartą i oczekiwać na dalsze dane w ramach tego połączenia — oczywiście podobny problem pojawia się w sytuacji odwrotnej. Tak czy inaczej, stanowa inspekcja pakietów ma swoją cenę.

Przepisywanie pakietów i translacja adresów sieciowych

Zwiększenie skuteczności interpretacji pakietów oraz ochrony przed atakami omijającymi reguły filtrujące (na przykład z wykorzystaniem mechanizmu fragmentacji) wymagało wyposażenia firewalli nie tylko w możliwość przesyłania pakietów, ale również modyfikacji niektórych przesyłanych danych. Jedno z rozwiązań polega na przykład na eliminowaniu wieloznaczności poprzez obowiązkowe składanie wszystkich fragmentów każdego pakietu przed dokonaniem analizy pakietu zgodnie z regułami zdefiniowanymi przez administratora.

W miarę rozwoju coraz to bardziej zaawansowanych rozwiązań stało się oczywiste, że przepisywanie pakietów nie tylko jest korzystne dla sieci, ale może wręcz stanowić przełom w kwestii bezpieczeństwa i możliwości sieci, pozwalając na wprowadzanie tak użytecznych rozwiązań, jak na przykład translacja adresów sieciowych (NAT, od ang. *network address translation*). Technika ta polega na odwzorowaniu określonych adresów IP na inne adresy przed ich wysłaniem i zastosowaniu operacji odwrotnej w przypadku pakietów odsyłanych przez chroniony system. Stanowa translacja pozwala między innymi implementować odporne na awarie infrastruktury sieciowe, w których jeden publiczny adres IP jest w rzeczywistości obsługiwany przez kilka serwerów wewnętrznych. NAT może też posłużyć do wprowadzenia w sieci wewnętrznej puli prywatnych (publicznie niedostępnych) adresów IP, pozwalając jednocześnie wszystkim systemom sieci korzystać z Internetu poprzez jeden adres publiczny — mamy wtedy do czynienia z maskaradą.

W pierwszym przypadku NAT przepisuje adresy docelowe w przychodzących pakietach na adresy pewnej ilości różnych systemów po wewnętrznej stronie firewalla. Pozwala to stworzyć odporną na awarie infrastrukturę, w ramach której żądania wyświetlenia popularnej witryny internetowej (na przykład *http://www.microsoft.com*)

lub udostępnienia innej popularnej usługi są rozkładane na wiele systemów — jeśli jeden zawiedzie, inne będą mogły przejąć jego funkcję. Można to osiągnąć stosując specjalne urządzenia wyrównujące obciążenie, ale podobne możliwości oferuje wiele firewalli z obsługą NAT.

Drugie zastosowanie, zwane popularnie maskaradą, polega na przepisywaniu adresów źródłowych wychodzących pakietów w taki sposób, by systemy w chronionej sieci wewnętrznej (potencjalnie korzystające z prywatnych adresów, które nie zostałyby skierowane do tej sieci z Internetu) mogły łączyć się ze światem zewnętrznym dzięki przechwytywaniu i przepisywaniu ich połączeń wychodzących przez firewall. Systemy sieci wewnętrznej są dla świata niewidoczne, a wszystkie ich połączenia wychodzące wydają się odbiorcom pochodzić od firewalla. Każde połączenie jest przypisywane do konkretnego, publicznego adresu IP i określonego portu, po czym jest wysyłane w świat. Pakiety powracające do tego adresu są następnie przepisywane z powrotem, by wskazywały na system, który faktycznie nawiązał dane połączenie, a na koniec przesyłane do sieci wewnętrznej. Dzięki takiemu rozwiązaniu cała prywatna sieć stacji roboczych, które w założeniu nie mają udostępniać żadnych usług, nie jest bezpośrednio dostępna z zewnątrz, co znacznie zwiększa jej bezpieczeństwo i ukrywa niektóre informacje o jej strukturze, a w dodatku pozwala zaoszczędzić na kosztownych, publicznych adresach IP dla poszczególnych stacji roboczych. Wprowadzenie maskarady pozwala organizacji posiadającej tylko jeden adres publiczny stworzyć sieć wewnętrzną liczącą setki czy wręcz tysiące komputerów i zapewnić każdemu z nich dostęp do Internetu.

Niedokładności translacji

Również translacja adresów jest zadaniem znacznie trudniejszym, niż mogłoby się wydawać — działanie niektórych protokołów wyższego poziomu jest znacznie bardziej skomplikowane od prostego podłączenia się do zdalnego systemu i przesłania mu zestawu poleceń. Na przykład pradawny, lecz wciąż szeroko rozpowszechniony protokół przesyłania plików FTP [4] (File Transfer Protocol) w swym najprostszym i najpopularniejszym trybie działania wymaga nawiązania zwrotnego połączenia od serwera do klienta w celu przesyłania żądanych plików, a początkowe połączenie nawiązane przez klienta jest używane jedynie do wydawania poleceń. Wiele innych protokołów (w szczególności obsługujących rozmowy, połączenia peer-to-peer, mechanizmy współużytkowania danych, transmisje multimedialne i tym podobne) korzysta z własnych, niekiedy dość dziwnych rozwiązań, wymagających połączeń zwrotnych, przeskakiwania portów czy też dopuszczania niesesyjnej transmisji określonych danych (na przykład pakietów protokołu UDP) z powrotem do klienta.

Z tego też względu każda implementacja maskarady, która ma poprawnie obsługiwać takie protokoły, musi posiadać odpowiednią liczbę protokołów-pomocników. Zadaniem tych pomocników jest analiza danych aplikacji wymienianych w ramach połączenia, a w razie potrzeby przepisywanie pakietów i otwieranie tymczasowych szczelin w firewallu w celu wpuszczenia połączeń zwrotnych.

I tu właśnie tkwi kolejny problem, po raz pierwszy zauważony kilka lat temu w pomocniku FTP przez Mikaela Olssona [5], a później badany również dla innych pomocników, między innymi przez autora tej książki [6]. Problem polega na tym, że pomocnicy

podejmują decyzje o otwarciu szczeliny w firewallu na podstawie informacji przesyłanych od stacji roboczej do serwera zgodnie z określonym protokołem. Automatycznie zakładają też, że dane wysyłane przez system są transmitowane w imieniu użytkownika i za jego wiedzą. Nie muszę chyba mówić, że niektóre programy — na przykład przeglądarki internetowe — można podstępem zmusić do wysyłania określonych rodzajów danych, w tym nawet informacji „wyglądających" na protokół nieobsługiwany przez program, a odpowiednie spreparowanie takich danych i przesłanie ich aplikacji pozwala osiągnąć ten cel automatycznie. Sfałszowane dane mogą posłużyć do nakłonienia pomocnika, by na moment otworzył firewall.

Klasycznym przykładem takiego ataku jest wykorzystanie mechanizmu działania przeglądarki internetowej. Podając odniesienie do strony internetowej (lub elementu takiej strony) rzekomo dostępnej na niestandardowym porcie HTTP na komputerze atakującego (będącym jednocześnie jak najbardziej standardowym portem FTP), można zmusić klienta do połączenia się z tym zasobem i wysłania odpowiedniego żądania HTTP. Połączenie jest nawiązywane na porcie używanym na ogół przez FTP, więc pomocnik FTP firewalla zaczyna śledzić tę transmisję, oczekując na okazję do udzielenia pomocy.

Przetworzenie poniższego przykładowego adresu URL sprawiłoby, że klient HTTP spróbowałby się połączyć z portem FTP i wysłać coś, co wygląda na polecenie FTP `PORT`, przechwycone następnie przez pomocnika firewalla:

```
HTTP://SERVER:21/FOO<RETURN>PORT MY_IP,122,105<RETURN>
```

Wysłane przez klienta żądanie byłoby dla prawdziwej usługi FTP kompletnym bełkotem, a jej odpowiedź byłaby niezrozumiała dla przeglądarki zgłaszającej żądanie — ale nie o to tu chodzi. Najważniejsze, że napastnik ma kontrolę nad częścią takiego żądania, a mianowicie nazwą pliku, którego klient zażąda z serwera. Ową fikcyjną nazwę wybiera agresor i może ona zawierać dowolne dane. Połączenia nasłuchuje pomocnik firewalla dla protokołu FTP, oczekujący konkretnego polecenia tekstowego (`PORT`). Umieszczając w nazwie pliku ciągi znaków właściwe żądaniom FTP, napastnik może przekonać pomocnika, że użytkownik usiłuje w rzeczywistości pobrać konkretny plik z serwera. W ten oto sposób zdalny serwer uzyskuje tymczasowo możliwość połączenia się z komputerem klienta (w tym przypadku na wysoce podejrzanym porcie 31337 — 122*256+105 = 31337), a więc atakujący zostaje wpuszczony do systemu bez wiedzy ofiary. Ups — tego też się nie spodziewaliśmy.

Konsekwencje maskarady

Wszystkie wspomniane scenariusze ataku wiążą się z wykorzystywaniem słabości maskarady, ale już sama jej obecność może stanowić źródło ciekawych informacji o zdalnym systemie.

Jak już wiemy, maskarada jest mechanizmem inwazyjnym, gdyż istotą jej działania jest modyfikacja wychodzącego ruchu sieciowego poprzez przepisywanie wybranych części pakietów. Proces ten wykracza poza samą podmianę adresu i pozwala uważnemu obserwatorowi nie tylko wykryć obecność maskarady, ale również zidentyfikować konkretny firewall. Pakiety pochodzące z maskarady mogą wykazywać następujące własności:

- Czas życia pakietów docierających do odbiorcy będzie się różnił od oczekiwanej lub zmierzonej odległości od sieci docelowej. Pakiety pochodzące zza maskarady będą zawsze co najmniej o jeden przeskok „starsze" od pakietów pochodzących z systemu bezpośrednio wykorzystującego adres IP chronionej sieci.
- W większości przypadków w sieci źródłowej występują różne systemy operacyjne, różne konfiguracje takich samych systemów lub przynajmniej różne czasy aktywności poszczególnych instalacji. Każdy taki system charakteryzuje się nieco innymi parametrami TCP/IP, omówionymi szczegółowo w rozdziałach 9. i 10. Analizując różne portrety TCP/IP dla połączeń z pozoru pochodzących z tego samego adresu IP, możemy z dużą dozą pewności rozpoznać obecność mechanizmu translacji adresów w konkretnym systemie, kryjącym za sobą sieć wewnętrzną.
- Zewnętrzny obserwator prawdopodobnie zwróci uwagę na przesunięcie portów źródłowych — nietypowe zjawisko spowodowane faktem, że połączenia wychodzące z sieci korzystają z portów efemerycznych, znajdujących się poza normalnym zakresem danego systemu operacyjnego.

  Każdy system operacyjny rezerwuje określony zakres portów źródłowych w celu identyfikacji połączeń wychodzących. Firewall często odwzorowuje maskaradowane połączenia, korzystając z innego zakresu portów, właściwego jego systemowi operacyjnemu. Jeśli więc zakres obserwowanych numerów portów różni się od zakresu właściwego wykrytemu systemowi (na przykład jeśli Linux, na ogół korzystający z portów od 1024 do 4999, nagle wydaje się używać znacznie wyższych numerów portów), można wnioskować, że pakiety są poddawane translacji adresów, a niekiedy nawet ustalić konkretny model firewalla.

Techniki te są często stosowane i stanowią podstawę wykrywania maskarad oraz rozpoznawania chronionych przez nie sieci, ale istnieje też kilka innych sposobów stwierdzenia obecności przepisywania pakietów.

Segmentowa ruletka

Jednym z mniej oczywistych — a tym samym mniej popularnych — sposobów wykrywania obecności przepisywania pakietów i pozyskiwania dodatkowych informacji o konfiguracji sieci jest analiza wartości pola maksymalnego rozmiaru segmentu w otrzymywanych pakietach.

Fragmentacja pakietów IP wiąże się z dość znacznym kosztem przetwarzania pofragmentowanych danych, stąd też jest często uważana za zjawisko niepożądane z punktu widzenia wydajności i wielu implementatorów stara się jej zapobiegać. Jak już jednak wiemy, wyeliminowane fragmentacji jest zadaniem bardzo trudnym, gdyż jak dotąd nie istnieje żaden sposób dokładnego, szybkiego i niezawodnego określania maksymalnej jednostki danych (MTU) dla ustalonej trasy jeszcze przed nawiązaniem połączenia. Nawet najlepsza z dostępnych metod, czyli wykrywanie MTU trasy (PMTU), jest daleka od doskonałości, a jej włączenie i tak powoduje spadek wydajności — stosowane tu wykrywanie odpowiedniego MTU metodą prób i błędów oznacza, że niektóre zbyt duże pakiety trzeba odrzucić i wysłać ponownie.

Aby zapobiec spadkowi wydajności i niezawodności wynikającemu z włączenia wykrywania PMTU oraz zmniejszyć narzut związany z fragmentacją, wiele firewalli stosujących translację adresów i przepisywanie niektórych parametrów pakietów wychodzących modyfikuje również wartość pola maksymalnego rozmiaru segmentu (MSS) w nagłówkach TCP połączeń wychodzących z sieci prywatnej, co ma na celu dostosowanie tej wartości do MTU łącza zewnętrznego sieci. Nowa wartość jest najczęściej mniejsza od MTU sieci lokalnej, dzięki czemu nadawca nie będzie wysyłał danych, które nie zmieściłyby się w łączu zewnętrznym. W przypadku ustawienia MTU odpowiadającego najmniejszej wartości na całej trasie zmniejsza to ryzyko fragmentacji. (Podejście to zakłada, że niezgodności MTU najczęściej występują w pobliżu systemu nadawcy lub odbiorcy, w obrębie tzw. ostatniej mili, gdzie często występują różne łącza o niskim MTU, na przykład łącza DSL lub bezprzewodowe, z powodu których większe pakiety musiałyby być poddawane fragmentacji).

Samą zmianę ustawienia MSS niełatwo wykryć — prawdę mówiąc, nie istnieje żaden sposób stwierdzenia, czy wartość MSS ustawił nadawca, czy też została ona zmodyfikowana gdzieś po drodze. Jest jednak pewien drobny kruczek. Jak już wspominałem w rozdziale 9., algorytmy wyboru rozmiaru okna wielu współczesnych systemów mają pewną ciekawą cechę:

> Parametr rozmiaru okna określa ilość danych, jaka może być wysłana bez konieczności potwierdzenia. Konkretna wartość tego parametru jest często wybierana zgodnie z osobistymi wierzeniami voodoo programisty lub innymi jego przekonaniami religijnymi. Dwie najpopularniejsze zasady głoszą, że rozmiar okna powinien być albo wielokrotnością MTU bez nagłówków (maksymalnego rozmiaru segmentu, w skrócie MSS), albo po prostu wartością dostatecznie dużą i „okrągłą". Starsze wersje Linuksa (2.0) używały wartości będących potęgami dwójki (na przykład 16 384). Linux 2.2 przerzucił się na wielokrotności MSS (z jakiegoś powodu 11 lub 22 razy MSS), a nowsze wersje tego systemu często używają wartości od 2 do 4 razy MSS. Sieciowa konsola Sega Dreamcast używa wartości 4096, a systemy Windows często korzystają z 64 512.

Rosnąca liczba współczesnych systemów (w tym nowsze wersje Linuksa i Solarisa, niektóre wersje Windows i SCO UnixWare) korzystają z rozmiaru okna będącego wielokrotnością MSS. Dzięki temu można łatwo stwierdzić, czy ustawienie MSS pakietu zostało zmienione gdzieś po drodze, gdyż rozmiar okna takiego pakietu nie będzie już konkretną wielokrotnością MSS, a najprawdopodobniej w ogóle nie będzie się dzielił przez MSS.

Porównując MSS z rozmiarem okna, można skutecznie wykrywać obecność firewalli stosujących wyrównywanie wartości MSS (czyli dostosowywanie jej do możliwości łącza) w wielu różnych systemach. Wyrównywanie MSS jest wprawdzie zachowaniem opcjonalnym dla Linuksa i FreeBSD, ale w wielu firewallach domowych i inteligentnych routerach DSL odbywa się ono automatycznie. Obecność niespodziewanej wartości MSS sygnalizuje więc obecność systemu nie tylko przepisującego pakiety, ale również zdolnego do translacji adresów, co z kolei może posłużyć za wskazówkę odnośnie połączenia sieciowego nadawcy.

Śledzenie stanowe i niespodziewane odpowiedzi

Inną istotną konsekwencją stanowego śledzenia połączeń i przepisywania pakietów jest fakt, że niektóre odpowiedzi wymagane przez specyfikację pochodzą od firewalla, a nie od nadawcy, co pozwala napastnikowi całkiem skutecznie zidentyfikować ów firewall. Gdy połączenie jest usuwane z tablicy stanów NAT (czy to z powodu przekroczenia czasu połączenia, czy też wysłania przez jedną ze stron pakietu RST, który nie dotarł do odbiorcy), dalszy ruch sieciowy w ramach kończonej sesji nie jest już przekazywany odbiorcy, lecz jest obsługiwany bezpośrednio przez firewall.

Zgodnie ze specyfikacją TCP/IP odbiorca powinien odpowiadać na wszelkie niespodziewane pakiety ACK odpowiedzią RST, co ma na celu poinformowanie nadawcy, że sesja, którą usiłuje kontynuować, nie jest już obsługiwana przez odbiorcę (lub nigdy nie była). Niektóre firewalle naruszają zalecenia specyfikacji i w ogóle nie odpowiadają na takie pakiety, po prostu ignorując wszelkie pakiety, które nie wydają się należeć do żadnej trwającej sesji. (Nie zawsze jest to postępowanie rozsądne, gdyż może prowadzić do zbędnych opóźnień w przypadku zaniechania poprawnego połączenia z powodu problemów transmisyjnych).

Wiele firewalli odpowiada jednak poprawnym i spodziewanym pakietem RST, co otwiera kolejną drogę ich wykrywania i identyfikacji. Pakiet ten jest tworzony od zera przez firewall, więc jego parametry dostarczają informacji o samym firewallu, a nie o chronionym systemie. Pozwala to wykorzystać opisane w rozdziale 9. tradycyjne techniki identyfikacji (badanie znaczników DF, wartości TTL, rozmiaru okna, doboru, wartości i kolejności opcji itd.) do uzyskania informacji o firewallu.

Istnieje też inna możliwość, wynikająca z następującego zapisu w dokumencie RFC1122 [7]:

> 4.2.2.12 Pakiet RST: RFC-793 sekcja 3.4
>
> Protokół TCP POWINIEN dopuszczać obecność danych w otrzymanym pakiecie RST.
>
> OMÓWIENIE: Istnieją propozycje umieszczania w pakiecie RST tekstu ASCII kodującego i tłumaczącego przyczynę wysłania RST. Nie ustalono jak dotąd standardu dla takich informacji.

I rzeczywiście — pomimo braku standardu niektóre systemy dołączają do pakietów RST wysyłanych w odpowiedzi na zbłąkane ACK opisowe (choć nierzadko tajemnicze) komunikaty w nadziei, że informacja o przyczynie błędu uspokoi nadawcę. Odpowiedzi te często zawierają wewnętrzne słowa kluczowe lub przejawy czegoś, co wygląda na dziwną odmianę komputerowego humoru i są niekiedy specyficzne dla konkretnego systemu operacyjnego, na przykład `no tcp, reset`; `tcp_close, during connect` (Mac OS); `tcp_fin_wait_2_timeout`; `No TCP` (HP/UX); `new data when detached`; `tcp_lift_anchor, can't wait` (SunOS).

Jeśli w reakcji na problemy sieciowe lub niespodziewany pakiet przesłany do zdalnego systemu zobaczymy pakiet RST z takim opisem problemu, a wiemy z innych źródeł, że system docelowy nie używa opisowych komunikatów RST, to mamy kolejną wskazówkę. Możemy na jej podstawie wydedukować, że między nami a odbiorcą znajduje się inny system — prawdopodobnie stanowy firewall — i zidentyfikować jego system operacyjny, porównując treść komunikatu ze znanymi odpowiedziami różnych systemów.

Obie techniki identyfikacji okazują się być bardzo skuteczne w wykrywaniu obecności stanowych filtrów pakietów, jeśli tylko możliwa jest obserwacja ruchu sieciowego podczas krótkotrwałych problemów sieciowych. Metody te mogą również posłużyć do aktywnej identyfikacji bez konieczności namierzania samego firewalla — wystarczy wysłać do ofiary zbłąkany pakiet ACK, co pozwoli odróżnić filtry stanowe od bezstanowych. Na podstawie otrzymanej odpowiedzi atakujący może dobrać najlepszy sposób poradzenia sobie z firewallem (lub wykorzystać tę wiedzę w inny sposób).

Niezawodność czy wydajność — spór o bit DF

Wykrywanie MTU trasy (PMTU) jest kolejnym kierunkiem identyfikacji, blisko spokrewnionym z omówioną już w rozdziale 9. kwestią unikania fragmentacji pakietów IP.

Nowsze wersje jądra Linuksa (2.2, 2.4, 2.6) i systemów Windows (2000 i XP) domyślnie implementują i włączają wykrywanie PMTU. Jeśli to ustawienie nie zostanie zmienione, wszystkich pakiety pochodzące z takich systemów mają ustawiony bit DF, zapobiegający fragmentacji. Algorytm wykrywania PMTU może jednak powodować problemy w rzadkich, ale mimo to możliwych sytuacjach.

Niepowodzenia wykrywania MTU trasy

Problem wykrywania PMTU polega na tym, że jego skuteczność jest całkowicie uzależniona od zdolności nadawcy do odebrania komunikatu ICMP „fragmentation required but DF set" i ustalenia optymalnych ustawień dla danego połączenia. Pakiet, który spowodował wysłanie komunikatu, zostanie odrzucony przed dotarciem do celu, więc musi być wysłany ponownie po odpowiednim zmniejszeniu jego rozmiaru.

Jeśli nadawca nie otrzyma tego komunikatu, to nie może wiedzieć, że pakiet nie dotarł do celu. Powoduje to w najlepszym razie opóźnienie transmisji, a w najgorszym razie zablokowanie połączenia na czas bliżej nieokreślony, gdyż retransmitowane pakiety również raczej się nie będą mieścić w łączu, dla którego maksymalny dopuszczalny rozmiar pakietu jest za mały dla wysyłanych danych.

Rzecz w tym, że wcale nie ma gwarancji, że nadawca otrzyma komunikat ICMP generowany w reakcji na pakiet niemieszczący się w łączu. W niektórych sieciach wszystkie

komunikaty ICMP są po prostu ignorowane w źle pojętym interesie bezpieczeństwa. Czyli nawet gdy router wyśle odpowiedni komunikat, to i tak wcale nie musi on dotrzeć do celu.

Skąd pomysł odrzucania komunikatów ICMP? Po prostu w przeszłości zdarzało się, że niektóre takie komunikaty stanowiły zagrożenie dla bezpieczeństwa. Zbyt duże lub pofragmentowane pakiety ICMP powodowały w wielu systemach błąd pamięci jądra (tak zwany „ping of death"), natomiast wysyłanie komunikatów ICMP na adresy rozgłoszeniowe służyło do wywołania burzy odpowiedzi na podstawiony adres IP (atak „Smurf") i do wykonywania ataków DoS. Co więcej, błędnie skonfigurowane systemy często interpretowały ogłoszenia routerów[2] — specjalny rodzaj komunikatów rozgłoszeniowych — jako polecenia modyfikacji ustawień sieciowych. Systemy takie akceptowały ogłoszenia bez sprawdzania ich wiarygodności, co stwarzało jeszcze jeden ciekawy kierunek ataku. I tak właśnie doszło do tego, że wielu obawia się ICMP i odrzuca jego pakiety.

W co bardziej naiwnych poradnikach bezpieczeństwa można niekiedy znaleźć zalecenie odrzucania wszystkich pakietów ICMP, i niektórzy administratorzy tak właśnie czynią. Widziałem nawet taką rekomendację w profesjonalnym raporcie z testów penetracyjnych, sporządzonym ręką znanego audytora (którego nazwiska niestety nie mogę tu wymienić).

Inną kwestią potencjalnie zmniejszającą niezawodność wykrywania PMTU jest fakt, że niektóre komunikaty o błędach pochodzą z urządzeń korzystających z prywatnych adresów IP. Zdarza się, że w celu zaoszczędzenia ograniczonej (i na ogół kosztownej) przestrzeni adresów publicznych interfejsy fizycznie łączące router z firewallem zdalnej sieci otrzymują adresy z lokalnej puli prywatnej zamiast z puli publicznych adresów kierowanych do tej sieci ze świata zewnętrznego.

Wykorzystanie adresów prywatnych może niestety całkowicie uniemożliwić wykrywanie PMTU. Dlaczego? Dlatego że jeśli firewall odbiorcy otrzyma z zewnątrz pakiet o rozmiarach uniemożliwiających jego przekazanie do celu, to wyśle do nadawcy komunikat o błędzie ze swojego własnego adresu, należącego do puli adresów prywatnych. Firewall nadawcy oryginalnego pakietu może taki pakiet odrzucić, gdyż pozornie pochodzi on z zewnątrz, ale ma adres źródłowy z puli prywatnej (potencjalnie nawet takiej samej, jak pula adresów w sieci prywatnej nadawcy). Firewall odrzuci taki pakiet, gdyż wygląda on na typową próbę podszycia się pod zaufany system lokalny. W tym jednak przypadku decyzja firewalla spowoduje sparaliżowanie stosowanego od dość niedawna mechanizmu wykrywania PMTU i pozostawi nadawcę w błogiej nieświadomości faktu, że wysłany przez niego pakiet nie dotarł do celu.

Co gorsza, nawet jeśli żaden ze wspomnianych problemów nie wystąpi i pakiet poprawnie dotrze do celu, to wiele współczesnych urządzeń sieciowych ogranicza tempo wysyłania odpowiedzi ICMP i nie dopuszcza do przekroczenia określonej liczby

[2] Ogłoszenia routerów miały na celu umożliwienie automatycznej konfiguracji komputerów w sieci, bez konieczności ręcznego wprowadzania ustawień. Router rozgłaszał co jakiś czas (lub na żądanie) komunikat oznaczający „jestem tutaj, skorzystaj ze mnie". Niektóre systemy domyślnie przyjmowały niezamawiane ogłoszenia routerów bez większego zastanowienia, co oczywiście było złym pomysłem.

pakietów w ustalonym czasie. Również ten zabieg stanowi dodatkowe zabezpieczenie. Komunikaty ICMP miały pierwotnie przeznaczenie czysto informacyjne i przed wprowadzeniem algorytmów wykrywania PMTU nie były istotne dla procesu komunikacji, więc ograniczenie tempa ich wysyłania wydawało się dobrym sposobem obrony przed niektórymi atakami blokowania usługi czy zapychania łącza.

Walka z wykrywaniem PMTU i jej następstwa

W świetle powyższego wiele osób uważa wykrywanie PMTU na kiepski pomysł. Technika ta daje wprawdzie pewien przyrost wydajności, ale kosztem rzadkich, lecz długotrwałych i często trudnych w wykryciu problemów, powodujących niemożność połączenia się w pewnymi serwerami lub nagłe zawieszanie połączeń. Opracowano wprawdzie wiele różniących się skutecznością algorytmów wykrywania „czarnych dziur", czyli obszarów sieci, dla których należy wyłączyć wykrywanie PMTU, jednak ich stosowanie nie rozwiązuje podstawowego problemu, a może powodować dodatkowe opóźnienia transmisji — najczęściej wtedy, gdy są one najmniej pożądane.

Aby sprostać tym trudnościom i zmniejszyć ilość zażaleń, niektórzy producenci komercyjnych firewalli korzystają w konfiguracji swych produktów z pewnej chytrej sztuczki: usuwają znacznik DF ze wszystkich pakietów wychodzących. Jest to drobna i nierzadko skuteczna modyfikacja, dostarczająca jednak kolejnego sposobu wykrycia obecności systemu filtrującego i przepisującego pakiety. Jeśli obserwowany adres lub sieć wykazuje cechy charakterystyczne dla systemów z wykrywaniem PMTU, ale w otrzymywanych pakietach brakuje spodziewanego znacznika DF, uważny obserwator może wykryć obecność firewalla i określić jego typ, tym samym zyskując kolejną cząstkę informacji bez konieczności bezpośredniej interakcji z ofiarą.

Do przemyślenia

Tak kończy się moja krótka opowieść o tym, jak to próby ulepszenia firewalli i zwiększenia ich możliwości w celu utrudnienia ataku i bezpośredniego rozpoznania miały też skutek uboczny w postaci ułatwienia ich pasywnej identyfikacji. W tym miejscu pozwolę sobie jeszcze na małą dygresję.

Z jednym z najdziwniejszych i najciekawszych zjawisk sieciowych miałem do czynienia w 1999 r. Nie jest ono wprawdzie bezpośrednio związane z działaniem firewalli, ale mimo to stanowi ciekawy materiał do przemyślenia dla każdego, kto interesuje się zagadnieniami pasywnej identyfikacji systemów pośrednich.

Jacek P. Szymański, z którym przez krótki czas współpracowałem, a później miałem przyjemność omawiać nietypowe i podejrzane wzorce ruchu sieciowego[3], zauważył nagły wzrost liczby poważnie uszkodzonych pakietów TCP/IP przychodzących na

[3] Współpraca ta zaowocowała w pewnym momencie powstaniem luźnej grupy polskich badaczy, która w latach 1999 – 2000 zajmowała się korelacją, śledzeniem i próbami wyjaśnienia wielu dziwnych prawidłowości w ruchu sieciowym.

port 21536 (a w mniejszym stopniu również porty 18477 i 19535). Wadliwe pakiety zawsze pochodziły z portów 18245, 21331 lub 17736, należących do licznych systemów korzystających z przestrzeni adresów przydzielanych połączeniom wdzwanianym obsługiwanym przez Telekomunikację Polską.

Gdy udało się przechwycić kilka takich pakietów, okazało się, że zawarte w nich dane są dziwnie i bardzo poważnie zniekształcone. Pakiety docierały z nienaruszonymi nagłówkami IP (i typem protokołu prawidłowo ustawionym na TCP), ale natychmiast po tych nagłówkach następowały dane TCP — nagłówków TCP po prostu nie było. Zaobserwowane kombinacje portów wynikały z interpretacji pierwszych czterech bajtów danych jako pary liczb (gdyby pakiety posiadały prawidłowe nagłówki TCP, w tych miejscach znajdowałyby się numery portów źródłowego i docelowego). Para 18245 i 21536 była po prostu numeryczną reprezentacją ciągu „`GET `" — czterech znaków otwierających większość żądań HTTP przesyłanych w Internecie. Kombinacja 18477 i 21331 odpowiadała ciągowi `SSH-`, rozpoczynającemu każdą sesję Secure Shell, a wartości 19535 i 17736 odpowiadały poleceniu `EHLO`, otwierającemu sesje protokołu ESMTP (Extended SMTP).

Pochodzenie takich właśnie pakietów pozostawało jednak tajemnicą, podobnie jak powód, dla którego przychodziły one z tej właśnie sieci. A jeśli rzeczywiście zdarzały się urządzenia sieciowe, które zniekształcały pakiety w ten sposób, to dlaczego zjawisko to nie powodowało problemów z połączeniami czy też innych niedogodności dla użytkowników?

Odpowiedź wkrótce nadeszła. Jak się okazało, wszystkie kalekie pakiety pochodziły z wprowadzonych nieco wcześniej przez TP systemów obsługi modemów Nortel CVX. Problem występował sporadycznie, wyłącznie pod dużym obciążeniem, więc tylko nieliczne zniekształcone pakiety były w ogóle wysyłane, a jeszcze mniej spośród nich trafiało do zaskoczonych odbiorców. Prawdopodobną przyczyną błędu było nieprawidłowe blokowanie kolejki lub błędne zarządzanie buforem — problemy dające się zauważyć dopiero podczas prawie jednoczesnego przetwarzania wielu sesji. Wyglądało na to, że niektóre pakiety były wysyłane za wcześnie, zanim zostały do końca zbudowane, czy też były w inny sposób zniekształcane.

Niedługo po wdrożeniu tych systemów w Polsce Nortel naprawił implementację TCP/IP w swoich urządzeniach i wszyscy żyli długo i szczęśliwie, ale nie one pierwsze (i z pewnością nie ostatnie) niechcący pozostawiały swój niepowtarzalny odcisk na przesyłanych pakietach.

Morał z tej opowieści jest taki sam, jak z poprzednich wywodów: niemądrze jest lekceważyć informacje, które na co dzień ignorujemy. We współczesnym świecie systemów sieciowych drobne wskazówki oraz obserwacje nietypowych, nieoczekiwanych lub niewyjaśnionych zachowań są niezwykle cenne — łatwo je znaleźć, ale trudno przeanalizować.

Potencjalnym tematem do przemyślenia i obszarem wartym dokładniejszego zbadania są metody stosowane w celu zapobiegania identyfikacji systemów. Niektórzy producenci firewalli próbują dołączać do swych produktów różne techniki zapobiegawcze,

polegające na wprowadzaniu zmian w wybranych parametrach TCP/IP (identyfikatorach IP, numerach sekwencyjnych TCP i tym podobnych). Nie muszę chyba mówić, że takie rozwiązania w rzeczywistości pomagają atakującemu i dają efekt wręcz przeciwny do zamierzonego. Jeśli zmianom i ujednoliceniu nie zostaną poddane absolutnie wszystkie parametry wykorzystywane do identyfikacji (w tym numery sekwencyjne, czasy retransmisji, wartości datowników i inne), to możliwa będzie nie tylko identyfikacja systemu operacyjnego ofiary, ale również firewalla chroniącego całą sieć.

C'est la vie.