

Testowanie bezpieczeństwa - metody, narzędzia, błędy...

Wojciech Dworakowski
ABA, sp. z o.o.
wojtekd@aba.krakow.pl

„Ufamy Bogu - wszystko inne sprawdzamy”

- Motto NSA (National Security Agency)

Abstrakt: Podczas wykładu, przedstawione zostaną cele i metodologie testowania bezpieczeństwa sieci, ze szczególnym uwzględnieniem testów penetracyjnych. Omówimy również najczęściej popełniane błędy w projektowaniu sieci i konfiguracji urządzeń oraz oprogramowania serwerowego, na podstawie doświadczenia zebranego przy prowadzeniu testów penetracyjnych. Referat zostanie poparty praktycznymi przykładami z przeprowadzanych przez naszą firmę testów.

1. Co to jest system bezpieczny?

Wykład ma dotyczyć testowania bezpieczeństwa. Na początek, zadajmy więc sobie pytanie: „Co to jest system bezpieczny?”. Czy w ogóle istnieje takie pojęcie?

Na codzień spotykamy się z informacjami marketingowymi, rozmaitych firm zajmujących się produkcją systemów operacyjnych, oprogramowania, narzędzi, itp. W większości z nich znajdziemy informacje, że produkty te zapewniają bezpieczeństwo. Jednak co to znaczy? Niestety "bezpieczeństwo systemów" stało się w obecnych czasach, słowem - wytrychem.

W zespole konsultantów naszej firmy, definiujemy bezpieczny system informatyczny, jako system bezpieczny „fizycznie”. Inaczej mówiąc jest to system, którego przełamanie jest fizycznie nie możliwe.

Przykład 1: Bezpieczny serwer WWW - Serwer, którego zawartość umieszczono na nośniku *read-only*, np. na płycie CD-ROM.

Przykład 2: Jak zabezpieczyć stacje robocze? - Stosować terminale w miejsce komputerów PC.

Na powyższej definicji opiera się prawidłowe podejście do zabezpieczania systemów informatycznych: **Usuwanie zbędnych elementów, zamiast stosowania wyszukanych szpcepionek.**

Co to jest w takim razie niebezpieczeństwo, czy też luka w systemie zabezpieczeń?

Dla nas jest to nie tylko błąd, który da się w tej chwili, przy określonym zasobie wiedzy i środków wykorzystać, ale również błędy, które mogą być dużym zagrożeniem przy zaistnieniu odpowiednich okoliczności.

Za przykład niech posłuży pozostawienie możliwości zdalnego podłączenia się do bazy danych, w systemie, w którym taka funkcjonalność nie jest w ogóle potrzebna. Normalnie, przy założeniu że produkt którego używamy jest bezpieczny, nie stanowi to zagrożenia. Ale, czy możemy być pewni że produkt jest bezpieczny? Czy możemy zagwarantować że jutro nie zostanie opublikowany exploit¹ pozwalający na wykonywanie poleceń na bazie danych bez zalogowania się, lub może nawet dostęp z uprawnieniami administratora? Nawet jeżeli informacja o wykryciu błędu nie zostanie szeroko opublikowana, to skąd wiemy że zagrożenie nie istnieje? Czy mamy pełne zaufanie do autorów oprogramowania? Czy warto ryzykować, skoro dana funkcjonalność nie jest nam w ogóle potrzebna lub jest mało przydatna?

¹ program pozwalający na wykorzystanie zagrożenia

2. Po co testować?

Większość administratorów, lub nawet osób odpowiedzialnych za bezpieczeństwo informacji, ślepo wierzy w produkty lub usługi renomowanych firm. Działa tu prosta logika - czym produkt lub usługa droższe, tym bardziej bezpieczne. Niestety często bywa to później weryfikowane przez mniej lub bardziej uzdolnionych intruzów - hakerów.

Każdy stosowany produkt, jak również całą instalację informatyczną, należy testować, głównie po to by uprzedzić potencjalnych intruzów. Lepiej jest wykryć niebezpieczeństwo samemu, niż czekać aż zrobią to hakerzy.

Inne powody dla których warto testować bezpieczeństwo, to:

- Weryfikacja przestrzegania polityki bezpieczeństwa.
- Wzbudzenie czujności pracowników.
- Możliwość wykrycia innych problemów „przy okazji” wybiórczego testu.
- Odpowiedzialność prawna - testy są jednym ze środków zapewniających ochronę informacji niejawnych.

3. Rodzaje testów

Testować można różne aspekty bezpieczeństwa. Przystępując do testów należy ściśle określić co będzie przedmiotem testów, oraz jaki jest cel testów.

3.1. Kto testuje?

Oczywistym podziałem, jest podział ze względu na to kto wykonuje test. Mamy więc testy prowadzone wewnątrz przez samych administratorów lub też osoby odpowiedzialne za bezpieczeństwo informacji, oraz testy prowadzone przez audytorów zewnętrznych. Należy podkreślić że testy „wewnętrzne” i „zewnętrzne” nie konkurują ze sobą. W dobrze skonstruowanej polityce bezpieczeństwa powinno być miejsce zarówno dla testów przeprowadzanych przez pracowników firmy, jak i dla testów przeprowadzanych przez audytorów z zewnątrz. Należy raczej zastanowić się jak często powinny być wykonywane oba rodzaje testów.

Każda firma, w zależności od wielkości i stopnia skomplikowania infrastruktury informatycznej oraz biorąc pod uwagę analizę ryzyka związanego z bezpieczeństwem informacji, powinna ustalić częstotliwość zarówno audytów wewnętrznych jak i zewnętrznych.

Pewne jest że zawsze należy testować bezpieczeństwo we własnym zakresie. Należy jednak pamiętać że testy związane z bezpieczeństwem wymagają sporego doświadczenia i zasobu wiedzy. Dla firmy nie zawsze jest opłacalne zatrudnianie wysokiej klasy specjalistów. Bardziej ekonomicznym rozwiązaniem może być częściowy lub nawet całkowity outsourcing zarządzania bezpieczeństwem.

Nawet jeżeli korporacja posiada najwyższej klasy specjalistów, to i tak warto raz na jakiś czas poddać się testom prowadzonym przez audytorów z zewnątrz, ponieważ daje to możliwość spojrzenia na problem „świeżym okiem”, z innej perspektywy. W większości poradników na temat konstruowania polityki bezpieczeństwa, jest ściśle zalecenie, iż ten kto zabezpiecza nie może potem testować własnych zabezpieczeń.

3.2. Metodologie

Istnieje wiele metodologii testowania bezpieczeństwa. Żadna z nich nie jest uniwersalna i zawsze przyjętą metodologię należy dostosować do testowanego problemu. Inaczej wykonuje się test bezpieczeństwa sieci a inaczej test aplikacji. Jednakże zawsze przed przystąpieniem do

testowania należy ustalić jaka metodologia będzie stosowana. Szczególnie ważne jest to przy zamawianiu audytu w firmie zewnętrznej. Zamawiając audyt starajmy się nie kupować „kota w worku”. Musimy wiedzieć co dana firma audytorska chce robić, jak chce zabrać się do problemu. Często pozwala to już na wstępie uniknąć nieporozumień i odrzucić oferty niezbyt profesjonalne.

Najczęściej spotykane typy testów to:

Test penetracyjny typu „black-box”. Istotą testów penetracyjnych jest jak najbliższe naśladowanie działań hakerów. Można powiedzieć że są to autoryzowane przez klienta próby włamania do sieci. Przy zamawianiu tego typu testu trzeba ustalić jak wiele informacji o infrastrukturze sieciowej będzie posiadał wykonujący test. Należy pamiętać że testy zmierzają do oceny stopnia bezpieczeństwa sieci. Ocena ta jest ściśle związana z przyjętymi założeniami, a najważniejszym założeniem powinna być ilość informacji, którą na wstępie posiada potencjalny intruz.

W testach penetracyjnych typu „black-box” testowana sieć na początku stanowi dla audytora tajemnicę. Powinien on posiadać minimum informacji ograniczające się do tego co może wiedzieć potencjalny intruz, nie związane bliżej z firmą (np. zakres adresów IP).

Zaletą tych testów jest spojrzenie na testowany problem z perspektywy potencjalnego intruza, a więc weryfikacja odporności na rzeczywiste ataki. Testy te mają charakter „żywiolowy” nie można z góry ustalić kolejności działań. Działania te zależą od znalezionych potencjalnych luk i są ustalane podczas trwania audytu. Dzięki temu można często wykryć nowe lub nietypowe problemy, czasami nie związane bezpośrednio z celem testu.

Wadą testów „black-box” jest wyjątkowa czasochłonność. Test polega na zdobywaniu coraz większych ilości informacji o audytowanym obiekcie. Zdobyta informacja służy jako punkt wyjścia do kolejnych testów i analiz. Np. w wyniku skanowania listy adresów IP uzyskujemy listę aktywnych hostów. Adresy z tej listy skanujemy w poszukiwaniu otwartych portów TCP i UDP. Zidentyfikowane aktywne usługi służą jako materiał wejściowy do prób penetracji.

Test penetracyjny z ograniczonym dostępem do informacji. Często korzystnie jest już na początku przyjąć że potencjalny intruz tak czy inaczej zdobędzie pewną wiedzę o atakowanym obiekcie. W związku z tym już na początku testu audytor posiada pewną (ograniczoną) informację o testowanej sieci, programie, instalacji. Może to być np. informacja o stosowanych systemach, zasadach działania instalacji, procedurach, stosowanych systemach zabezpieczających. Tego typu informacja z reguły jest tak czy inaczej w miarę łatwo osiągalna dla atakującego.

Można też z góry założyć, że atakujący może uzyskać dostęp do którejś ze stacji roboczych w atakowanej sieci i sprawdzić jak dużo informacji będzie mógł dzięki temu zdobyć. W takim wypadku audytorzy powinni działać nie z zewnątrz (z Internetu) ale z wewnątrz sieci (z Intranetu).

W obu typach testów penetracyjnych należy również ustalić jaki jest cel testów. To znaczy, należy ustalić granicę - jak daleko może sięgnąć testujący w przypadku znalezienia luki. Czy powinien on tylko to zanotować w raporcie? Czy może spróbować wejść w posiadanie jakiejś ważnej informacji? Czy może spróbować coś zmienić w systemie? A może w całości skasować system? Ustalenie tego na początku audytu pozwoli na uniknięcie wielu przykrych niespodzianek, takich jak: wyciek poufnych informacji lub ich całkowita utrata. W testach wykonywanych przez nasz zespół, przyjęliśmy, że przy znalezieniu luki w systemie zabezpieczeń, rejestrujemy ją w raporcie, a na wyraźne zlecenie klienta próbujemy wykorzystać określone luki do zdobycia chronionej informacji.

Weryfikacja tez polityki bezpieczeństwa. Ten typ audytu polega na testowaniu tylko ściśle wyznaczonego fragmentu infrastruktury, a nie całości instalacji. Jest to sprawdzenie pojedynczego szczegółu w instalacji. Np.:

- systemu firewall,
- protokołu transmisji,

- pojedynczej aplikacji,
- wykrywanie modemów za pomocą których można się dostać do sieci (*wardialing*).

Testy takie powinny polegać na sprawdzaniu tez sformułowanych w polityce bezpieczeństwa. Np.:

„Do sieci nie można się dostać przez bezpośrednie wdzwonienie się na podłączony do niej modem”

„Stacje w Internecie nie mają dostępu do zasobu X”

Zaletą takich wybiórczych testów, jest możliwość skupienia się na pojedynczym problemie, bardzo uporządkowana metodologia działań, oraz ściśle wyznaczony cel testu.

Należy jednak pamiętać, że audytujący stwierdzi tylko określony fakt. Do właściciela sieci należeć będzie natomiast interpretacja tego faktu. Jaki ma on wpływ na całość infrastruktury?

Inne typy testów. Pseudotesty. Często do testowania sieci stosuje się skanery zintegrowane - automaty testujące bezpieczeństwo sieci. Część z firm oferujących usługi audytorskie w zakresie bezpieczeństwa, opiera wyniki swoich analiz tylko i wyłącznie na wynikach działania skanerów bezpieczeństwa. Należy wyraźnie podkreślić, że narzędzia te nie powinny być stosowane jako podstawa do wyciągania całościowych wniosków. Są one zaprojektowane, jedynie jako narzędzia wspomagające pracę autytora. Test skanerem zintegrowanym nie może być uważany za test penetracyjny.

Tego typu test można natomiast stosować jako:

- porównanie wyników z wynikami uzyskanymi inną drogą,
- okresowe, automatyczne sprawdzenie, w ograniczonym zakresie, stopnia bezpieczeństwa sieci.

Często stosuje się politykę testowania polegającą na okresowym (np. raz na rok) zamówieniu audytu w firmie zewnętrznej i częstym (np. raz na miesiąc) przeglądaniu sieci skanerem zintegrowanym.

4. Narzędzia

Przy testowaniu bezpieczeństwa systemów, stosuje się skanery zintegrowane, o których mówiliśmy przed chwilą, oraz narzędzia wspomagające testowanie ręczne. Obie grupy mają swoje plusy i minusy, jednakże nie należy ich porównywać bezpośrednio, ponieważ są one przeznaczone do różnych celów.

4.1. Testowanie ręczne

Klasyczny test penetracyjny przeprowadza się przy użyciu drobnych narzędzi niskopoziomowych, które pozwalają na bezpośrednią interakcję z daną usługą lub protokołem. Do takich narzędzi można zaliczyć:

- klient TCP - np. netcat, telnet
- klient danego protokołu - np. przeglądarka WWW
- analizatory ruchu - np. tcpdump, ethereal
- narzędzia do przetwarzania danych

Stosuje się również narzędzia automatyzujące pewne czynności, wspomagające proces testowania, np:

- Skanery portów - np. nmap

- Narzędzia pozwalające na zdalne rozpoznanie wersji systemu - np. queso, nmap
- Narzędzia realizujące spoofing - np. hunt, arp spoof
- Snifery haseł - np. dsnifi
- Crackery - np. l0phtcrack, Crack

Stosowanie narzędzi niskopoziomowych, pozwala bardzo szczegółowo kontrolować system informatyczny. Za pomocą nich można wykryć nie tylko błędy popełnione przez twórców oprogramowania, ale także błędy proceduralne, popełnione przez osoby stosujące te oprogramowanie, czyli projektantów i administratorów. Doświadczenie uczy, że tego typu błędy są najczęstsze.

Niestety, żeby używać narzędzi niskopoziomowych, trzeba posiadać szeroką wiedzę na temat protokołów, systemów, itd. Trzeba wiedzieć jak ich używać, opracować własne procedury. Często, trzeba dostosować narzędzie do swoich potrzeb.

Narzędzia te są tworzone dla zaawansowanych administratorów, osób zajmujących się audytami oraz... intruzów. Właśnie dlatego testy wykonywane niskopoziomowo, przy użyciu tych narzędzi, są najbardziej zbliżone do prawdziwych ataków.

4.2. Skanery automatyczne

Skanery automatyczne, jest to grupa produktów, przeznaczonych do okresowego zautomatyzowanego przeglądu bezpieczeństwa instalacji. W żadnym wypadku nie służą one do prowadzenia testów penetracyjnych. Przede wszystkim dlatego, że są one nastawione na testowanie znanych „dziur” w oprogramowaniu, a nie na wykrywanie błędnego stosowania tego oprogramowania. Narzędzia te mogą też wspomagać i przyspieszać jeden z etapów testu - identyfikowanie „dziurawego” oprogramowania.

Produkty te są relatywnie proste w użyciu, do ich obsługi wystarczy średnia wiedza techniczna. Generują one przejrzyste i jasne raporty. Jednakże należy pamiętać że mogą one wykryć tylko te błędy, o których wiedzą, które są w ich bazie błędów.

Najważniejszą cechą decydującą o jakości skanera jest właśnie kompletność bazy błędów, oraz dostępność uaktualnień tej bazy, szybkość reakcji na nowe problemy. Inne cechy jakie należy wziąć pod uwagę przy wyborze skanera to: wydajność skanowania, poprawność wnioskowania (liczba fałszywych alarmów), unikanie niepotrzebnych testów (np. przeprowadzanie testów charakterystycznych dla produktów Windows na hostach które zostały zidentyfikowane jako Unixy) i (*last but not least*) cena.

W tej chwili na rynku pojawiło się sporo tego typu produktów. Do najbardziej znanych należą:

- ISS Internet Scanner - dosyć kompletne lecz bardzo drogie narzędzie
- Axent NetRecon - narzędzie tanie, lecz posiadające sporo niedostatków (duża ilość fałszywych alarmów)
- eEye Retina - stosunkowo nowy, obiecujący produkt
- Nessus - narzędzie *Open Source*, całkowicie darmowe, bardzo kompletna i na bieżąco uaktualniana baza

5. Najczęściej spotykane błędy

Postaram się przedstawić przegląd i klasyfikację najczęściej spotykanych błędów w testowanych przez nas instalacjach. O dziwo okazuje się że znacznie więcej zagrożeń wynika z braku wiedzy technicznej lub właściwych procedur (polityki bezpieczeństwa), niż z tego że stosowane oprogramowanie posiada „dziury”.

5.1. Błędy w fazie projektowania infrastruktury

Najczęstszym, spotykanym niemal powszechnie tego typu błędem jest zabezpieczenie sieci od frontu, lecz pozostawienie „tylnych drzwi” przez które można ominąć wszelkie zabezpieczenia. Często spotykamy się z sieciami w których zabezpieczenie od strony Internetu jest przeinwestowane - są to drogie systemy zaporowe, które mają zapewnić (wg ulotek) kompletne bezpieczeństwo. Jednocześnie w tych samych sieciach można znaleźć modemy podłączone do stacji, za pomocą których można się wdzwonić do sieci omijając wszelkie zapory.

Często spotykaną praktyką jest umieszczanie niezabezpieczonych serwerów i stacji roboczych przechowujących mało istotne informacje (np. serwerów testowych) w sieci zabezpieczonej drogimi systemami zaporowymi. Stacje te po opanowaniu mogą służyć jako przyczółki dla intruza w sieci wewnętrznej.

Tego typu przypadki są wynikiem błędnego projektu sieci lub błędnej bądź nie przestrzeganej polityki bezpieczeństwa. Trzeba pamiętać o podstawowej zasadzie bezpieczeństwa systemów: „łańcuch jest tak mocny jak jego najsłabsze ogniwo”.

Przykład: Duże przedsiębiorstwo, 600 audytowanych numerów telefonicznych, 27 wykrytych modemów, 4 systemy zgłaszały się loginem.

Co zrobić? Okresowe audyty sieci telefonicznej, wykrywanie modemów.

5.2. Polityka bezpieczeństwa

Wiele testowanych przez nas instalacji, nawet w bardzo dużych instytucjach sprawia wrażenie, że nie obowiązuje w nich żadna polityka bezpieczeństwa, bądź przyjęte zasady nie są w praktyce stosowane.

Bardzo często spotykamy np. serwery które sprawiają wrażenie braku opieki administracyjnej.

Przykład: Sieć 1500 adresów IP, 5 serwerów WindowsNT, 3 z nich z Service Pack < 4

Co zrobić? Śledzenie na bieżąco list *bugtraq* lub *Security Advisories* producenta systemu. Zlecenie opieki nad bezpieczeństwem systemu firmie zewnętrznej. Częste kontrolowanie sieci.

5.3. Błędy w konfiguracji

Instalowane systemy i urządzenia często są zainstalowane „z pudełka”. Konfiguracja standardowa narzucona przez producenta, przeważnie nie jest dostosowana pod względem bezpieczeństwa. W tego typu konfiguracjach często są włączone wszystkie usługi udostępniane przez dany system, niezależnie od tego czy są one potrzebne w danej instalacji, czy też nie. Zbędne usługi mogą stanowić zagrożenie bezpieczeństwa systemu. Musimy pamiętać o tym że twórcy oprogramowania i urządzeń często przedkładają wygodę użytkownika nad bezpieczeństwo.

Przykład 1: Routery i urządzenia sieciowe, zazwyczaj mają włączony protokół zarządzania zdalnego - SNMP i standardowe community-strings („private”, „public”). Umożliwia to zdalne odczytanie a czasami również zmianę konfiguracji urządzenia.

Przykład 2: W serwerze WWW - Microsoft IIS, standardowo jest instalowany skrypt do odpluskwiania (debuggingu) kodu, który pokazuje źródła dowolnego skryptu ASP.

Co zrobić? Sprawdzanie systemów we własnym zakresie. Uruchamianie tylko niezbędnych usług. Poddanie konfiguracji systemu, audytowi firmy zewnętrznej.

5.4. Niewłaściwe stosowanie technologii

Często bezgranicznie ufamy technologii zapominając, że działa ona pod warunkiem spełnienia określonych założeń. Nie ma technologii uniwersalnej zabezpieczającej wszystko. Np. często spotykanym stwierdzeniem jest: „Mam firewall, więc moja sieć jest bezpieczna”. Owszem firewall

zabezpiecza sieć, ale pod warunkiem że jest dobrze skonfigurowany. Najczęściej firewalle są skonfigurowane w ten sposób, że ograniczają tylko i wyłącznie ruch wchodzący do sieci, a nie ingerują w ruch wychodzący. W takiej konfiguracji, atakujący może uzyskać łączność z systemem, zarażając go koniem trojańskim (np. za pomocą listu elektronicznego), który nawiąże sesję z wewnątrz sieci, do komputera atakującego.

Innym przykładem mogą być technologie szyfrujące. W szyfrowaniu z kluczem publicznym, przeważnie stosuje się tylko szyfrowanie informacji bez podpisywania. Należy pamiętać że szyfrowanie nie weryfikuje pochodzenia listu, za to odpowiada podpis.

Ponad to przy stosowaniu technologii szyfrujących często zapomina się o podstawowej zasadzie: Szyfrowana informacja jest na tyle bezpieczna, na ile zabezpieczony jest klucz, którym szyfrujemy.

Co robić? Kształcić pracowników odpowiedzialnych za utrzymanie sieci. Korzystać z usług konsultantów. Audytować sieć.

5.5. Błędy w oprogramowaniu

Ten rodzaj błędów nie jest niestety zależny od nas - użytkowników, lecz od twórców oprogramowania. Należy przyjąć założenie, że nie ma oprogramowania bez błędów. Jedyne co możemy zrobić to śledzić na bieżąco poprawki związane z bezpieczeństwem, oraz chronić się na innych poziomach - separować potencjalnie groźne oprogramowanie przez systemy zewnętrzne (firewalle, filtry proxy), oraz przez ograniczenia na poziomie systemów operacyjnych (uruchamianie oprogramowania z minimalnymi uprawnieniami w systemie).

Zauważalnym trendem w ostatnich czasach jest odnajdywanie mnóstwa błędów w oprogramowaniu komercyjnym, którego analiza powinna być utrudniona, ponieważ nie ma dostępnego kodu źródłowego. Błędy (często trywialne) odnajdywane w oprogramowaniu takich firm jak (Microsoft, Oracle, IBM, itd) weryfikują tą opinie. Bardziej prawdziwa wydaje się być teza, że czym więcej osób może sprawdzić kod, tym większa szansa na wczesne odnalezienie błędu i tym bezpieczniejsze oprogramowanie.

Np. błąd związany z błędnym formatowaniem adresu URL w serwerze WWW - Apache został usunięty ponad 3 lata temu. W konkurencyjnym serwerze WWW - Microsoft IIS, został wykryty dopiero w rok temu (*Unicode Directory Traversal*).

5.6. Zaufanie do „uznanych producentów”

Podczas prac przy sprawdzaniu bezpieczeństwa instalacji i oprogramowania stosowanego w naszym kraju, zauważamy wręcz bezgraniczne zaufanie do tego co piszą ulotki reklamowe, lub co mówią przedstawiciele producenta. Jeżeli producent systemu jest znaną, dużą firmą, to niejako a priori zakładamy że przeznaczają on wystarczająco dużo środków na sprawdzenie bezpieczeństwa aplikacji. Niestety rzadko kiedy jest to prawda.

Należy pamiętać że producenci sprzętu i oprogramowania nie wymyślają technologii, a jedynie ją implementują. Implementacje te często są niesprawdzone i pełne podstawowych błędów.

Przykład 1: IBM WebSphere. Oprogramowanie do obsługi handlu elektronicznego. Teoretycznie powinno być wyjątkowo bezpieczne, ale... posiada podstawowe błędy implementacyjne. Do szyfrowania haseł został zastosowany protokół 3DES, jednak został źle zaimplementowany. Prawidłowo, hasło zakodowane, powinno być wynikiem szyfrowania określonego ciągu znaków, algorytmem 3DES za pomocą klucza, którym jest hasło. W WebSphere hasło jest szyfrowane za pomocą stałego klucza. Na dodatek zakodowane hasło da się odczytać zdalnie, przez podanie odpowiednio sformatowanego adresu URL.

Przykład 2: Checkpoint Firewall-1. Renomowany system firewall. Bardzo popularny w Polsce i na świecie. Na konferencji *Black Hat Briefings 2000*, ujawniono 4 poważne błędy w tym oprogramowaniu. M.in. w protokole zarządzającym oraz implementacji VPN. (Po prezentacji przedstawiciel Checkpointa rozdał uczestnikom CD z poprawkami).

Podobnym źródłem zagrożeń może być to, że producenci oprogramowania często polegają tylko na własnej wiedzy, zamiast korzystać z powszechnie uznanych standardów, zakładając że utajnienie kodu i algorytmów podnosi bezpieczeństwo (*security by obscurity*). Przykładem może być stosowanie własnych algorytmów szyfrujących w miejsce bezpiecznych algorytmów, które są znane od lat.

Przykład: Technologie szyfrujące uznane przez kryptologów za niewystarczająco bezpieczne:

Checkpoint Firewall-1 - własny szyfr FWZ-1.

GSM A3/A5/A8 - protokoły szyfrowania i uwierzytelniania w telefonii GSM

Równie zgubne może być zaufanie do rozwiązań wymyślonych wewnątrz, przez samą instytucję (np. we własnych aplikacjach).

Przykład: Podsystem do realizacji zdalnych transakcji w dużej instytucji finansowej. Do szyfrowania danych zastosowano własny algorytm. Algorytm został złamany w ciągu kilku godzin przez jednego konsultanta, za pomocą prostej analizy statystycznej. W ciągu kilku następnych godzin powstała aplikacja, która pozwalała na rozszyfrowywanie danych w czasie rzeczywistym. System działał przez 10 lat ! Został stworzony przez dużego, znanego integratora i był już wcześniej audytowany!

Co robić? Tylko kod poddany szerokiemu audytowi i krytyce, może zostać uznany za bezpieczny. Trzy wyjścia:

- Zakładamy że jeżeli kod jest otwarty, to na pewno go ktoś sprawdził
- Zakładamy że producent mówi prawdę
- Sprawdzamy sami
- Zlecamy audyt aplikacji firmie zewnętrznej