Studia i Materiały Informatyki Stosowanej, Tom 2, Nr 2, 2010
str. 23-30

NIEINWAZYJNE METODY OCENY I POPRAWY BEZPIECZEŃSTWA SYSTEMÓW KOMPUTEROWYCH BAZUJĄCE NA STANDARDACH DISA

Marcin Kołodziejczyk

Akademia-Górniczo-Hutnicza
wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki
Al. Mickiewicza 30
30-059 Kraków
e-mail: marcink@agh.edu.pl

Artykuł jest próbą analizy dostępnych, nieinwazyjnych metod pozwalających na całościową analizę i ocenę bezpieczeństwa systemów komputerowych. Opisane metody nie ingerują bezpośrednio w działanie gotowych systemów, skupiając się w dużej mierze na statycznej analizie systemu. Analiza taka opiera się na stwierdzeniu zgodności systemu z poszczególnymi wymaganiami bezpieczeństwa, tworzonymi przez organizację DISA. Organizacja ta swoje wymagania przedstawia w trojaki sposób: tworząc statyczne wymagania tzw. STIG'i, manualne procedury, mające pomóc w analizie bezpieczeństwa oraz skrypty automatyczne dla części systemów (tzw. SRR). Przekrój systemów objętych wymaganiami jest ogromny, zaczynając od systemów operacyjnych, idąc przez systemy bazodanowe, sieci a skończywszy na tworzonych przez programistów aplikacjach. Artykuł zawiera również krótką analizę pewnych braków, niedogodności i wad wyżej wymienionych metod oraz jest próbą odpowiedzi w jakim kierunku powinien iść ich rozwój.

Słowa kluczowe: *Autentykacja (uwierzytelnianie), autoryzacja, aplikacja, baza danych, bezpieczeństwo, cheklista (procedura manualna), GoldDisk, hasła, interfejs, konta użytkowników, Linux, logi, logowanie zdarzeń systemowych, Microsoft Windows, PKI (infrastruktura klucza publicznego), SRR, STIG, system operacyjny, Unix, usługi sieciowe, wymagania*

Non-intrusive assessment approach and improving security of computer systems based on DISA standards

Abstarct: *This article is an attempt to analyze the available, non-intrusive methods of analyzing and assessing if the overall security of computer systems. These methods do not interfere directly in the working systems, focusing largely on static analysis of the system. The analysis base on the finding of compliance with the various safety requirements, formed by the DISA organization. This organization presents its requirements in three ways: by creating the static requirements, so-called STIGs, manual procedures (checklists), helped in the analysis of the security and automated scripts for system components (SRR). The scope of the systems covered by the requirements is very big, including operating systems, database systems, the network and applications created by software developers. This article also contains a brief analysis of some shortcomings, disadvantages and drawbacks of the above-mentioned methods and is an attempt to answer for the question: what should be done in the nearest future for static security analysis.*

Keywords: *authentication, authorization, application, checklist (manual procedure), database, GoldDisk, passwords, interface, Linux, logs, Microsoft Windows, network services, operating system, PKI (Public Key Infrastructure), requirements, security, SRR, STIG, system event logging, Unix, user accounts*

1. WSTĘP

Celem niniejszej publikacji jest prezentacja metod weryfikacji bezpieczeństwa istniejących, bądź tworzonych systemów informatycznych i telekomunikacyjnych w oparciu o wytyczne organizacji DISA (ang. Defence Information Systems Agency) dostarczającej rozwiązania informatyczne m.in. dla Departamentu Obrony Stanów Zjednoczonych (ang. Departament of Defense). Istnieje wiele innych metod audytu jak COBIT, CRAMM, które nie są tematem tego

artykułu. DISA zdecydowanie się wyróżnia najbardziej precyzyjnymi wymaganiami dotyczącymi technicznych aspektów bezpieczeństwa systemu komputerowego. Przez to, że większość wymagań tworzonych przez organizację DISA jest przeznaczona dla Departamentu Obrony USA, można zakładać, że poziom bezpieczeństwa opisany w takich wymaganiach jest wysoki. Wymagania te pozwalają nie tylko ocenić aktualny stan bezpieczeństwa systemu, ale również identyfikują słabe punkty systemu, a także pozwalają identyfikować zagrożenia już na etapie tworzenia danej wersji systemu. Metody oparte na wytycznych Departamentu Obrony są w dużej mierze bezinwazyjne w związku z czym istnieje znikome ryzyko uszkodzenia działającego systemu. Metody te zakładają również bardzo dobrą znajomość badanego systemu przez osobę, która przeprowadza audyt bezpieczeństwa. Wymagania stawiane przez DISA są podzielone na różne kategorie, względem rodzaju systemu jak również zagrożenia stwarzanego przez potencjalne dziury w bezpieczeństwie poszczególnych składowych systemu.

2. PODZIAŁ WYMAGAŃ [1]

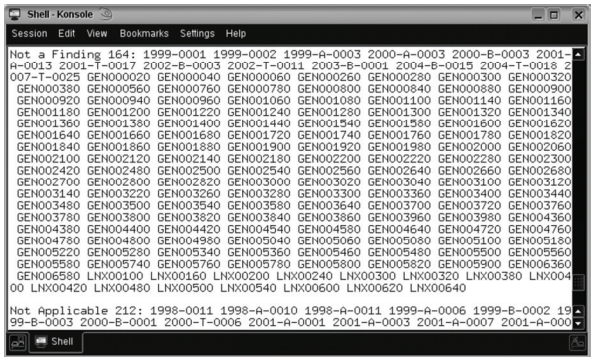
Duże, złożone systemy często są budowane w oparciu o środowiska przynajmniej częściowo heterogeniczne (np. systemy Unix i Windows). Z kolei ilość małych systemów, korzystających nawet tylko z jednego konkretnego systemu operacyjnego i pisanych w jednym konkretnym środowisku programistycznym jest ogromna. Nie da się, więc stworzyć jednego dokumentu, określającego niezbędne wymagania bezpieczeństwa dla wszystkich systemów. Należy, podzielić ogólne wymagania systemowe na mniejsze i dokładniejsze, wydzielając przy tym konkretne wymagania stawiane przed poszczególnymi składowymi systemu. Tak, więc wymagania dla systemu Linux będą nieco inne niż wymagania dla systemu Microsoft Windows. Podział taki ma jeszcze jedną zaletę, mianowicie wraz z wprowadzeniem nowej wersji produktu na rynek należy zmienić tylko jedną linię wymagań. Wytyczne tworzone przez organizację DISA obejmują systemy operacyjne, serwery baz danych, tworzone aplikacje użytkownika, konfigurację sieci, urządzeń USB, Web-serverów, maszyn wirtualnych itp. Zabierając się za analizę bezpieczeństwa konkretnego systemu należy najpierw wyodrębnić jego składowe, dla których DISA dostarcza konkretne wymagania.

Kolejnym podziałem jest podział wytycznych na trzy kategorie: wymagania, zwane dalej STIG'ami (ang. Security Technical Implementation Guides), procedury manualne (tzw. checklisty) i skrypty (ang. SRR - Security Readiness Review evaluation scripts). Należy tutaj zwrócić uwagę, że niestety te 3 kategorie bardzo często różnią się wieloma szczegółami i nie są sobie równoważne. Wszystkie trzy kategorie wymagań się częściowo uzupełniają (np. czasami procedura manualna jest bardziej restrykcyjna niż skrypt lub odwrotnie), chociaż niestety w wielu przypadkach są sprzeczne w szczegółach. To samo wymaganie w kilku miejscach może mieć inne znaczenie. Przykładem może być chociażby wymaganie GEN000800 dla systemów UNIX. Wymaganie (STIG) mówi, że ostatnie 10 haseł nie może być ponownie użyte, natomiast procedura manualna mówi o 5 hasłach. Dla przeciętnego programisty i administratora nie są to, więc duże różnice. Chcąc jednak spełnić jak najwięcej wytycznych organizacji DISA, wymagania takie jak przytoczone wyżej mogą sprawiać kłopot. Na szczególną uwagę zasługują skrypty, które można błyskawicznie uruchomić na badanym systemie, nie zmieniając przy tym żadnych z jego ustawień. Skrypty takie niestety dostępne są tylko na ograniczoną liczbę systemów. Przykładem mogą być skrypty SRR dla systemów klasy UNIX oraz tzw. GoldDisk dla Microsoft Windows. Brakuje np. skryptów dla wielu baz danych, przez co cała analiza bezpieczeństwa musi być przeprowadzana ręcznie.

Każde wymaganie przynależy do jednej z kilku kategorii. Kategoria I jest najbardziej krytyczną kategorią z punktu widzenia bezpieczeństwa. Luki w implementacji wymagań kategorii I mogą prowadzić do największych zniszczeń w systemie i wiążą się ze stosunkowo dużym ryzykiem ataku bądź przypadkowego uszkodzenia systemu. Najmniej ważne wymagania znajdują się w kategorii III lub IV (zależnie od systemu). Niespełnienie części z tych wymagań nie pociąga za sobą dużego ryzyka dla bezpieczeństwa systemu.

3. SYSTEMY UNIX

Wymagania dla systemów Unix obejmują wiele odmian tego systemu. Przykładami mogą być systemy takie jak: Solaris, HP-UX, AIX, IRIX oraz Linux. Problem sprawdzenia pewnych ustawień systemowych nie jest trywialny, gdyż niektóre, specyficzne pliki konfiguracyjne na Solarisie mogą być umiejscowione w innym katalogu niż np. w systemie HP-UX czy Linux. Warto tutaj wspomnieć, że skrypty automatyczne, tzw. SRR'y, dostępne dla Unix'a obsługują wszystkie wymienione systemy, uwzględniając różnice między nimi. Niestety nie są one jednak wolne od błędów.



Rysunek 1: Fragment wyniku skryptów SRR uruchomionych na systemie Linux. Widoczna tutaj jest lista spełnionych wymagań (Not a Finding) oraz fragment specyficznych wymagań, które się nie aplikują do danej wersji systemu (Not Applicable).

Pierwszą ważną rzeczą, na jaką zwraca się uwagę w wymaganiach jest integralność systemu. Chodzi tutaj zarówno o integralność sprzętową (np. nie wpinanie dodatkowych urządzeń do sieci) jak również integralność programową, która może być zapewniona przez okresowe sprawdzanie sum kontrolnych plików (tzw. hash'y), które są niezbędne do prawidłowej pracy systemu. W ten sposób można weryfikować czy pliki systemowe nie zostały zmienione (zmianę taką może powodować np. wirus, koń trojański czy inne dowolne, szkodliwe oprogramowanie). Watro zwrócić uwagę, że większość wymagań dot. integralności jest kategorii II. Tylko dwa z nich są kategorii I.

Kolejnym ważnym aspektem bezpieczeństwa systemów Unix, jest kontrola dostępu do wszelkiego rodzaju plików i zasobów. DISA w swoich wymaganiach precyzuje tutaj konfiguracje kont użytkowników, włącznie z określeniem warunków integralności między plikami takimi jak: /etc/passwd, /etc/shadow oraz /etc/group. Wymagania te specyfikują również zakres dostępnych identyfikatorów dla użytkownika i grupy, precyzując, które z nich są systemowe lub zarezerwowane i których nie powinno się używać. Ważną rzeczą, z prawnego punktu widzenia wielu państw (m.in. USA) jest wypisywanie odpowiedniego komunikatu w trakcie interakcyjnego procesu logowania użytkownika. Aby można było wyciągnąć prawne konsekwencje od intruza, należy go po prostu przedtem o takiej możliwości poinformować i wyświetlić stosowny komunikat (np. że ma do czynienia z systemem federalnym i nieautoryzowany dostęp jest karalny). Konta użytkowników powinny być obłożone restrykcjami czasowymi, zarówno pod względem długości ważności konta, jak również mechanizmu wylogowywania lub zamrażania sesji użytkownika po 15 minutach nieaktywności. Konta powinny być również zamrażane, jeśli użytkownik nie skorzysta z niego dłużej niż np. przez 35 dni. Wymagania dostarczane przez STIGi są bardzo restrykcyjne względem haseł użytkowników. Należy tutaj zauważyć jednak, że obecna wersja STIG'ów powstała w 2006 roku i wymagania dot. haseł zostały zmodyfikowane w checklistach oraz SRR'ach na bardziej restrykcyjne. Wymagania te precyzują długość haseł a także to, że każde hasło powinno zawierać małe i duże litery, cyfry oraz znaki specjalne. Szczególną uwagę przywiązuje się do konta administratora zwanego „root". Konto to powinno być szczególnie chronione chociażby przez sprawdzanie zmiennej środowiskowej $PATH (tak, aby administrator przypadkiem nie uruchomił innego programu niż zamierzony) czy w przypadku zdalnego dostępu do tego konta, zapewnienie tylko i wyłącznie szyfrowanego połączenia SSH. Użytkownik „root" posiada specjalne wymagania dotyczące praw dostępu do swoich plików, które różnią się od wymagań stawianych przed resztą użytkowników.

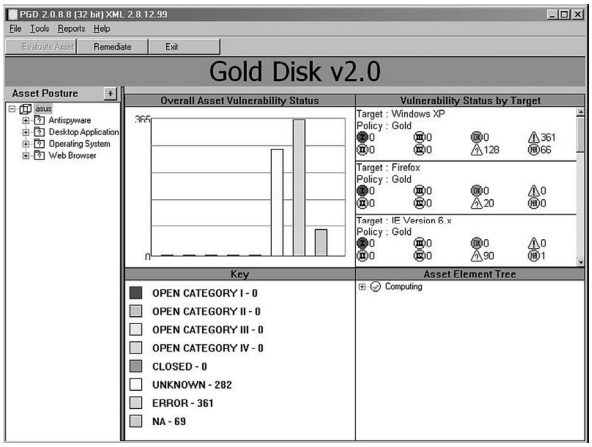
Ważną rzeczą w przypadku awarii lub włamania do systemu jest posiadanie stosownych logów. Wymagania Unix'owe kładą szczególny nacisk na procesy syslog'a oraz audit'a. Pierwszy z nich powinien działać na osobnej maszynie tak, aby w przypadku zapchania się logów nie odcinać dostępu do części funkcjonalnej systemu. Wyjątkiem, który pozwala na użycie syslog'a lokalnie jest sytuacja, w której syslog loguje tylko lokalne zdarzenia i nie akceptuje żadnych komunikatów z zewnątrz. O ile syslog nie jest zbyt uciążliwy dla systemu o tyle proces audyt może generować ogromne ilości informacji (włącznie z logowaniem udanych/nieudanych prób dostępu do pliku, czytania/pisania itd.). Może to powodować błyskawiczne zapychanie dysku, jednak DISA mimo to wymaga, aby pewne zdarzenia systemowe niezależnie od tego były logowane. Sposobem na bezpieczną implementację tych wymagań może być wprowadzenie rotacji logów, tak aby najstarsze logi były pakowane a następnie przy braku miejsca usuwane z dysku lub archiwizowane na zewnętrznych nośnikach danych. Dziwnym wydaje się tutaj fakt, że o ile istnieją wymagania dotyczące rotacji logów audytowych o tyle brak jest takich wymagań dla sysloga.

W przeciwieństwie do systemów Windows, wiele dystrybucji Linux'a zawiera dużą liczbę różnych serwerów i usług sieciowych. Można tutaj wymienić serwery www, poczty elektronicznej, ftp, tftp, telnet, DNS, samba, SSH, SNMP i inne. Wiele z nich jest opisanych i posiadają swoje, precyzyjnie zdefiniowane wymagania w STIG'ach.

4. SYSTEMY MICROSOFT WINDOWS

Obecna wersja wymagań dla systemów Microsoft Windows powstała w maju 2007 roku i obejmuje systemy Windows 2000, 2003, XP, 2008 oraz Vista. W chwili pisania artykułu brak jest wymagań dla specyficznych dla Windowsa 7. Podobnie jak w przypadku systemów UNIX, pierwszą rzeczą na jaką zwraca się uwagę w wymaganiach jest spójność systemu i cotygodniowe sprawdzanie czy pliki systemowe nie zostały zmienione. W tym przypadku podany jest nawet przykładowy skrypt, który może być uruchamiany co tydzień.

STIG'i dla Microsoft Windows przywiązują dużą wagę do bootowania systemu operacyjnego. Uruchomienie innego systemu z płyty bootowalnej lub pen drive'a znacząco ułatwia wiele rodzajów ataków. W przypadku systemów Windows znika częściowo problem niespójności wymagań z ręczną „checklistą", gdyż w przypadku systemu plików, wymagania w większości powołują się na checklistę, same nie precyzując żadnych konkretnych praw dostępu. Podobnie rzecz ma się z rejestrem systemowym. STIG'i dla systemu Windows również odnoszą się do logowania zdarzeń systemowych, konta administratora, automatycznego wylogowywania użytkownika po określonym czasie bezczynności oraz do haseł i usług sieciowych.



Rysunek 2: Program GoldDisk dla systemu MS Windows, służący do analizy aktualnego stanu bezpieczeństwa systemu operacyjnego.

Warto zwrócić uwagę, że w przypadku systemu MS Windows pojawiają się nowe wymagania dotyczące na przykład kosza. Ze względów bezpieczeństwa, wszystkie pliki powinny być usuwane natychmiast, nie powinny w ogóle się znajdować w koszu. Jest to poniekąd zablokowanie funkcjonalności dostarczanej przez kosz. Wymagania dotyczące kosza, nie istnieją jak na razie dla systemów Linux mimo, że wiele środowych graficznych posiada zaimplementowaną taką funkcjonalność. Wymagania dla Windows'a obejmują dodatkowo szeroki zestaw oprogramowania dostarczanych z systemem operacyjnym jak chociażby: .NET Framework, MSN Explorer, Windows Messenger, Media Player czy Internet Explorer. Microsoft dostarcza dla nowych systemów narzędzie zwane „Security Compliance Management Toolkit". Jest to owoc współpracy Microsoftu i National Security Agency, który pozwala w łatwiejszy sposób zabezpieczyć system w sposób wymagany przez STIGi.

5. BEZPIECZEŃSTWO BAZ DANYCH

Bazy danych są integralną częścią wielu systemów informatycznych. Niestety wymagania tworzone przez DISA dla baz danych mają postać płaską, czyli nie posiadającą żadnej hierarchii jak to ma miejsce w przypadku systemów operacyjnych. Wymagania bazodanowe nie są przez to w żaden sposób pogrupowane. Pierwszą ważną rzeczą poruszaną przez STIG'i jest to, aby używana baza danych była w wersji wspieranej przez producenta i aby były na niej zainstalowane wszystkie aktualne poprawki (ang. patch), zwłaszcza te dotyczące bezpieczeństwa (tj. security patche) [1,8]. Podobnie jak w przypadku systemów operacyjnych, należy monitorować wykonywalne pliki baz danych oraz wszystkie biblioteki używane przez bazę. Zmiana pliku wykonywalnego z silnikiem bazy danych może umożliwić intruzowi nieautoryzowany dostęp do danych. Również prawa dostępu do wszystkich plików skojarzonych z bazą powinny być jak najbardziej restrykcyjne. Ciężko jest tutaj podać konkretne wartości, gdyż liczba dostępnych na rynku silników baz danych jest ogromna. Ponadto bazy te można instalować na różnych platformach systemowych i sprzętowych.

Aby zapobiec utracie dużej części informacji zaleca się tworzenie kopii zapasowej wszystkich baz danych dostępnych w systemie. Wymagania nie precyzują jednak rodzaju kopii, czy ma to być tzw. backup pełny, przyrostowy czy różnicowy, ani jak często ma być taki backup wykonywany. Tworzenie kopii w przypadku baz danych różni się nieco od zwykłego kopiowania plików. Każda baza do tego celu udostępnia swoje narzędzia, które w poprawny sposób zrzucają zawartość bazy do pliku. Zwykłe kopiowanie może np. naruszyć więzy integralności. Kopiując plik nie mamy pewności czy akurat nie ma rozpoczętych transakcji w bazie lub czy baza nie aktualizuje jakichś danych. Specjalne narzędzia dostarczane przez producentów baz danych rozwiązują te

problemy. Warto zwrócić uwagę, że również kopia zapasowa powinna być ściśle chroniona odpowiednimi prawami dostępu, gdyż wydostanie się takiego backupu na zewnątrz może być równoznaczne z wyciekiem wszystkich danych z bazy.

Każdy dodatkowy, opcjonalny komponent, który nie jest należycie utrzymywany jest potencjalną luką w bezpieczeństwie systemu. Dlatego w systemach akredytowanych przez DISA wymaga się, aby wszystkie nieużywane komponenty bazy danych były usunięte. Przykłady, dostępne często w katalogach „examples" również pochodzą pod te wymagania a co za tym idzie, na systemie produkcyjnym nie powinny być dostępne.

O ile to możliwe zaleca się szyfrowanie wszystkich połączeń do bazy danych oraz samych danych w niej zawartych. Wymagania obejmują również tzw. infrastrukturę klucz publicznego (ang. Public Key Infrastructure – PKI) [6,7]. Niestety nie wszystkie dostępne na rynku silniki baz danych implementują wspomnianą funkcjonalność. Podobnie rzecz ma się z audytami bazodanowymi oraz logami. Mimo wielu wymagań, w większości systemów tylko część z nich może być spełniona, gdyż wiele z wymaganej funkcjonalności jest nie zaimplementowane.

Każdy użytkownik czy aplikacja korzystająca z bazy danych powinna mieć swoje osobne konto. Do konta tego powinny być przydzielone role, które uprawniają danego użytkownika do wykonania tylko niezbędnych dla niego operacji. Wszelkie inne operacje powinny być zabronione. W przypadku aplikacji mającej dostęp do bazy, hasła nie mogą być przechowywane tekstem jawnym, w żadnym pliku, skrypcie czy kodzie źródłowym. W przypadku narzędzi interaktywnych, w których użytkownik proszony jest o podanie swojej nazwy i hasła, hasło to nie powinno pojawiać się na ekranie jako tekst jawny podczas jego wpisywania. Wymagania dotyczące blokowania kont czy komplikacji haseł są podobne jak dla systemów operacyjnych Unix i Windows.

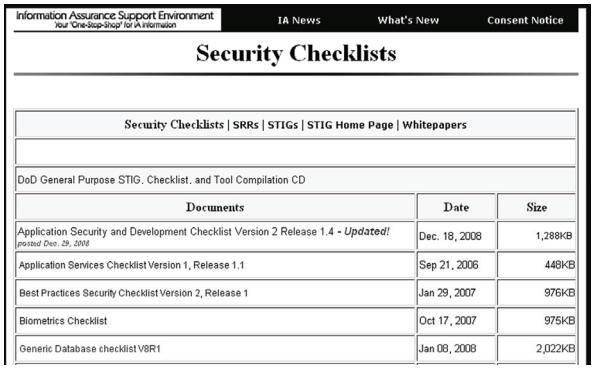
STIGi bazodanowe zalecają również blokowanie dostępu z bazy do zewnętrznych obiektów. Jeśli taki dostęp istnieje, należy odpowiednio nim zarządzać i powinien być stosownie udokumentowany. O ile to możliwe warto również zainstalować serwer bazy danych na osobnej maszynie, przeznaczonej tylko dla konkretnej bazy. Zmniejszy to ryzyko zablokowania dostępu do bazy przez inną usługę działającą niewłaściwie. Liczba równoległych połączeń do bazy powinna być również limitowana. Na szczególną uwagę zasługuje liczba równolegle wykonujących się transakcji. Może ona mieć duży wpływ na wydajność bazy i w skrajnych przypadkach może być niebezpieczna i służyć do przeprowadzenia tzw. ataku DoS (ang. Denial of Service).

Ostatnią rzeczą wartą uwagi w przypadku baz danych są skrypty automatyczne. Niestety dostępne są one tylko dla dwóch baz: MS SQL oraz Oracle (w tym drugim przypadku dostępne są skrypty na platformy UNIX'owe i Windowsowe). Autor testował tylko skrypty dla bazy Oracle. W przypadku wymagań specyficznych dla Oracle'a skrypty spisują się bardzo dobrze. Sytuacja ma się gorzej w przypadku wymagań ogólnych, gdyż tylko ok. 10% wymagań jest sprawdzanych przez skrypt. Powodem braku skryptów dla reszty silników bazodanowych jest ich różnorodność i specyficzność. Należałoby bowiem stworzyć osobne skrypty dla każdej bazy a ponadto w wielu przypadkach skrypty te byłyby zależne od konkretnej platformy.

6. INNE STIG'I

Oprócz opisanych wyżej STIG'ów dla systemów Unix, Windows oraz baz danych istnieje całe mnóstwo innych. Czasami zdarza się, że mimo, iż sam dokument ze STIG'ami nie istnieje, to istnieją procedury manulane (tzw. checklisty) lub skrypty automatyczne. Niestety w przypadku STIG'ów jedynymi systemami operacyjnymi dla których istnieją precyzyjne wytyczne są systemy Windows i UNIX/Linux. Nie zostały tutaj opisane pozostałe wymagania takie jak np. wymagania dot. sieci bezprzewodowych [4], maszyn wirtualnych czy urządzeń biometrycznych. Oprócz wyżej wymienionych istnieją jeszcze wymagania aplikacyjne, które skupiają się na bezpiecznym ich tworzeniu przez programistów. Wymaga się np. aby aplikacja sprawdzała na swoim interfejsie dane wejściowe jeśli np. obsługuje formatujące ciągi znaków, aby była odporna np. na taki typu „buffer overflow". Duży nacisk położony jest również na uwierzytelnianie użytkowników aplikacji oraz na autoryzację zadań przez nią wykonywanych. Ostatnie wersje STIG'ów aplikacyjnych zawierają kilkadziesiąt nowych wymagań w większości dotyczących samego procesu wytwarzania oprogramowania, np. używania kontroli wersji czy monitorowania wszelkiego rodzaju błędów.

Wszystkie, opisane i nieopisane tutaj wymagania, skrypty i procedury można znaleźć na stronach DISA [1]. Warto tylko zwrócić uwagę na ilość dokumentów. W chwili pisania artykułu było ich ok. 30. Procedur manualnych było natomiast ok. 70. Skryptów automatycznych było ok. 20.



Information Assurance Support Environment
Your "One-Stop-Shop" for IA Information

IA News | What's New | Consent Notice

Security Checklists

Security Checklists | SRRs | STIGs | STIG Home Page | Whitepapers

DoD General Purpose STIG, Checklist, and Tool Compilation CD

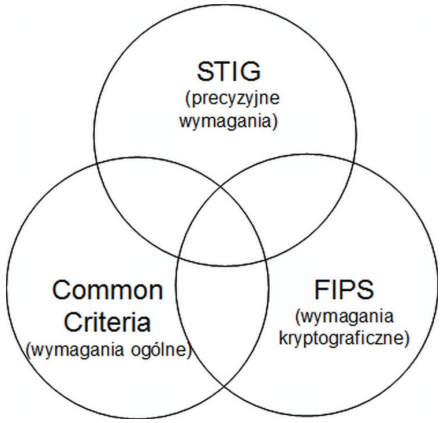
Documents	Date	Size
Application Security and Development Checklist Version 2 Release 1.4 - *Updated!* posted Dec. 29, 2008	Dec. 18, 2008	1,288KB
Application Services Checklist Version 1, Release 1.1	Sep 21, 2006	448KB
Best Practices Security Checklist Version 2, Release 1	Jan 29, 2007	976KB
Biometrics Checklist	Oct 17, 2007	975KB
Generic Database checklist V8R1	Jan 08, 2008	2,022KB

Rysunek 3: Fragment strony DISA [1] prezentującej dokumenty z wymaganiami bezpieczeństwa

Na rysunku 3 widać fragment strony z procedurami manualnymi (checklistami). W celu zdobycia skryptów automatycznych lub wymagań, należy wybrać z górnego menu odpowiednio „SRRs" lub „STIGs". W przypadku zastosowań w systemach komercyjnych, trzeba pamiętać, że mimo iż wszystkie wymagania są dostępne publicznie to jednak dostępne są tylko w najnowszej wersji. Dlatego tworząc konkretny projekt watro wcześniej zrobić kopię bazową wymagań na której system ma bazować.

7. PROCES COMMON CRITERIA [2]

Poruszając temat audytowania bezpieczeństwa systemów komputerowych należy wspomnieć o procesie zwanym Common Criteria. Jest to bardzo znany i rozpowszechniony proces oceny systemu komputerowego. Jest to proces zdecydowanie bardziej ogólny niż ocena zgodności ze STIG'ami. Oznacza to, że proces ten jest mniej techniczny. Jest to powód dla którego autor pominął go w dokładnych rozważaniach w artykule.



Rysunek 4: Zależności między STIG'ami, Common Criteria oraz FIPS

Common Criteria definiuje kilka podstawowych pojęć. Pierwszym z nich jest tzw. profil ochrony (ang. Protection Profile). Jest to zbiór wymagań jakie powinien spełniać konkretny produkt, np. System operacyjny, antywirus czy jakieś urządzenie sieciowe np. router. Jednym z etapów certyfikacji wg. Common Criteria jest wybranie celu, tzw. Security Target. Jest to dokładny opis właściwości jakie spełnia certyfikowany obiekt. Weryfikacja dokonywana jest na jednym z siedmiu poziomów pewności – EAL (ang. Evaluation Assurance Level). Wyższy poziom nie oznacza większego bezpieczeństwa systemu. EAL określa sposób oceny zgodności produktu z deklarowaną specyfikacją (np. testy funkcjonalne, formalna weryfikacja itd.)

8. STANDARDY FIPS [3]

FIPS (ang. Federal Information Processing Standard) jest organizacją zajmującą się głownie kryptograficznymi aspektami bezpieczeństwa. Na FIPS standardowo składają się dwie podgrupy. Pierwszą z nich jest Cryptographic Algorithm Validation Program (CAVP). Część ta odpowiedzialna jest za testowanie i ocenę algorytmicznych rozwiązań kryptograficznych. Nie ingeruje ona w konkretne implementacje a jedynie zaleca używanie wyspecyfikowanych algorytmów. Drugą gałęzią jest Cryptographic Module Validation Program. Ta część organizacji zajmuje się testowaniem konkretnych implementacji algorytmów. FIPS wypuszcza co jakiś czas listę certyfikowanych produktów. Obecna wersja tej listy to FIPS 140-2. Na liście znajdują się konkretne produkty wraz z podanie dokładnej wersji sprzętu lub oprogramowania (w zależności od rodzaju

produktu). Warto zwrócić uwagę, że nie każda wersja oprogramowania jest certyfikowana i obecność produktu z przed roku nie oznacza, że najnowszy produkt też jest bezpieczny i znajduje się na liście FIPS 140-2. W systemach o wysokich wymaganiach dot. bezpieczeństwa warto więc sprawdzać obecność używanych bibliotek i modułów na liście FIPS 140-2.

9. DALSZE KIERUNKI ROZWOJU

Niestety ciężko jest przewidzieć zmiany w wymaganiach tworzonych przez DISA oraz inne organizacje. Po analizie aktualnego stanu oraz rozwoju jaki nastąpił przez ostatnie dwa lata, można jednak spróbować wyciągnąć pewne wnioski. Wydaje się, że na pewno wraz z pojawieniem się nowego systemu operacyjnego Microsoftu, Windowsa 7, muszą pojawić się dla niego nowe, specyficzne wymagania dotyczące jego bezpieczeństwa. W przypadku systemów UNIX widać niewielki zastój w wymaganiach, które nie zmieniły się od marca 2006 roku. Jednak mimo to co kilka miesięcy powstają nowe wersje checklist, czyli manualnych procedur, które czasami są bardziej restrykcyjne niż same STIG'i. Co jakiś czas pojawiają się również skrypty automatyczne, które zawierają mniej błędów i testują coraz więcej wymagań. Widać wyraźnie, że sporym problemem są bazy danych, do których – w wielu przypadkach – brak jest oprogramowania testującego oraz wymagania nie są uporządkowane i nierzadko można je interpretować na kilka sposobów. Niektóre wymagania osiągnęły już swój docelowy poziom dojrzałości, niektóre zaś nadal ewoluują i co kilka miesięcy pojawiają się w nowych wersjach. Wydaje się, że DISA w dalszym ciągu będzie utrzymywała i aktualizowała swoje wymagania z czasem uzupełniając je skryptami automatycznymi lub coraz ściślejszymi procedurami manualnymi. Aplikacje co prawda mają swoje wymagania, jednak problemem wydaje się stwierdzenie czy dana aplikacja je spełnia. Jak zagwarantować, że tworzona aplikacja jest odporna np. na atak buffer overflow albo nie ma w ogóle wycieków pamięci? [5] Można użyć wielu komercyjnych narzędzi do wychwytywania tego typu błędów programistycznych, jednak o wiele łatwiej jest znaleźć błąd niż udowodnić, że go nie ma.

10. PODSUMOWANIE I WNIOSKI

Niniejsza publikacja miała na celu przegląd i krótkie wprowadzenie do metod nieinwazyjnego audytu bezpieczeństwa systemów opartych na wytycznych organizacji DISA oraz krótkie porównanie ich z innymi najbardziej znanymi metodami. Autor zastanawia się nad dalszymi kierunkami badań i rozwoju metod oceny bezpieczeństwa systemów. Istniejące metody pozwalają w znacznym stopniu zweryfikować poziom zabezpieczeń, jak również w wielu przypadkach pokazują pewne spojrzenie na kwestie bezpieczeństwa widziane od strony organizacji mających wysokie wymagania dotyczące tej kwestii. Zaletą opisanych wymagań i narzędzi jest fakt, że w trakcie testowania nie powinny uszkodzić systemu, w przeciwieństwie do innych, często inwazyjnych narzędzi takich jak np. skanery sieciowe takie jak Nessus, Nmap czy Foundstone. DISA jednak zaleca mimo to tworzenie kopii zapasowych przed uruchomieniem swoich skryptów. Opisane metody nie mogą zagwarantować, że system jest bezpieczny. W celu dokładnej weryfikacji należy zastosować przynajmniej kilka różnych narzędzi. Stosując jednak tylko checklisty lub SRR'y można przynajmniej dostać pogląd na to, co konkretnie w systemie stanowi zagrożenie, co z kolei pozwala stosunkowo niskim kosztem zwiększyć bezpieczeństwo poszczególnych jego komponentów.

Otwartą sprawą wydają się protokoły sieciowe. W jaki sposób je weryfikować, jeśli nie ma do nich wymagań? Jak zweryfikować wyższe warstwy korzystające ze znanych protokołów? Te pytania pozostają nadal otwarte, gdyż nie da się na nie udzielić prostej odpowiedzi. Wydaje się, że mogą one wyznaczyć pewne trendy i kierunki badań w przyszłości. Zagadnienie bezpieczeństwa protokołów sieciowych jest szczególnie bliskie autorowi. Wydaje się, że właśnie w tym kierunku należy prowadzić pracę i badania mające na celu wprowadzenie pewnych standardów wymaganych przez nowoczesne protokoły sieciowe.

Literatura

1. Dokumenty i skrypty ze strony organizacji IASE oraz DISA:
   http://iase.disa.mil/stigs/stig/index.html
   http://iase.disa.mil/stigs/checklist/index.html
   http://iase.disa.mil/stigs/SRR/index.html
2. Common Criteria portal:
   http://www.commoncriteriaportal.org/
3. Informacje o FIPS 140:
   http://csrc.nist.gov/publications/fips/
4. Marcin Kołodziejczyk - Applying of security mechanisms to low layers of OSI/ISO network model - Automatyka, wyd. AGH, Kraków 2010 (w druku, w języku angielskim)

5. Michael Howard, David LeBlanc, John Viega - The 19 Deadly Sins of Software Security – McGraw-Hill/Osborne, California 2005
6. Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone - Handbook of Applied Cryptography - CRC Press 1996 (wersja online)
7. Marek R.Ogiela - Security of computer systems - Wydawnictwa AGH, Kraków 2002
8. The Top 10 Most Critical Internet Security Threats - (2000-2001 Archive) - http://www.sans.org/top20/2000/
9. Marcin Kołodziejczyk - Tablice tęczowe jako skuteczna optymalizacja algorytmu brute-force - Elektrotechnika i Elektronika, wyd. AGH, Kraków 2009/2010 (w druku)