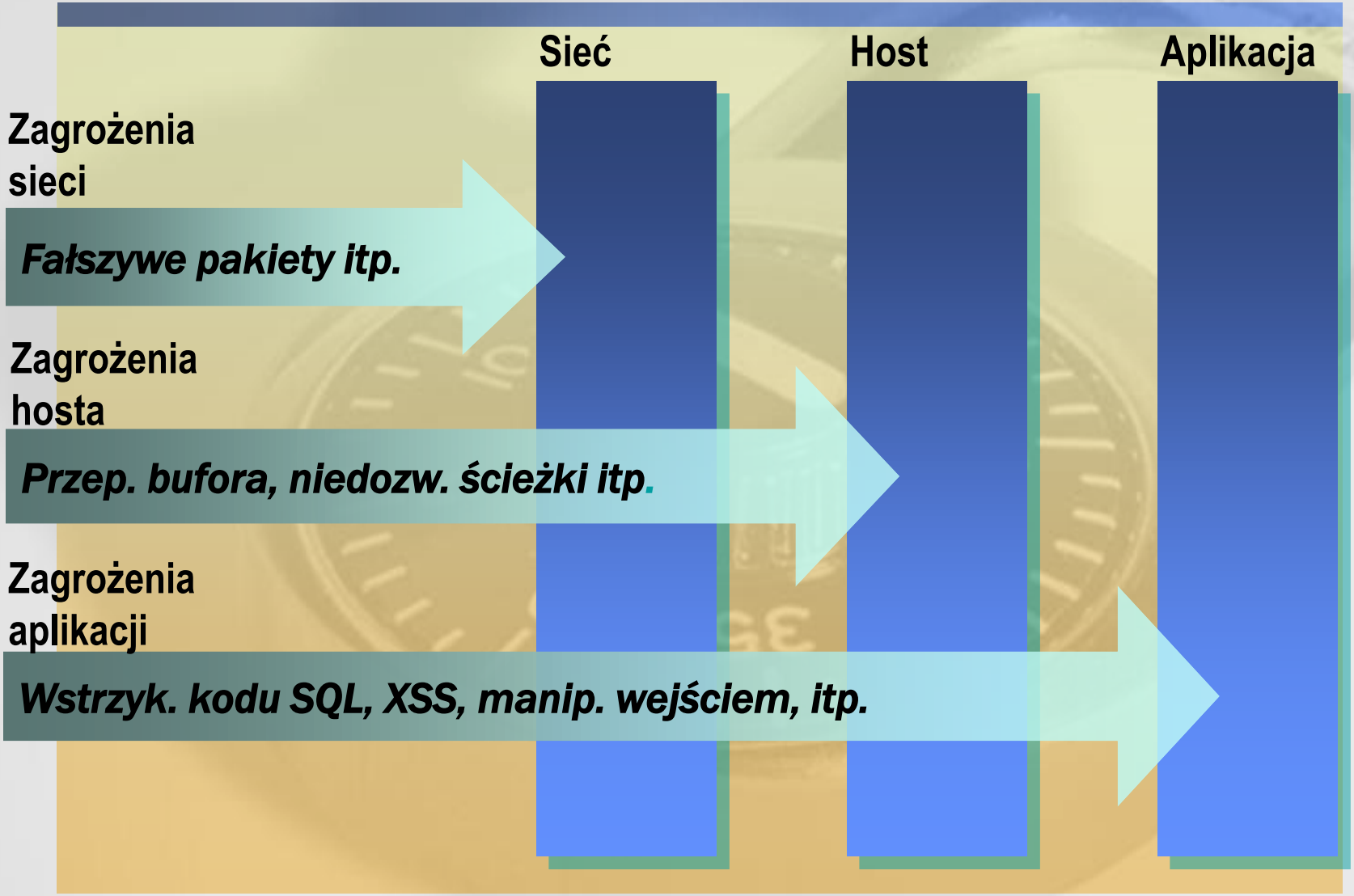
Aplikacje

Marcin Szeliga
marcin@wss.pl

Agenda

- **Typowe zagrożenia**
- **Bezpieczeństwo programów**
  - Zmiana paradygmatu
  - Aplikacje WWW
- **Kontrolowanie uruchamianych programów**
  - Zasady ograniczeń oprogramowania
- **Bezpieczeństwo baz danych**
  - Model bezpieczeństwa serwera SQL 2005

Typowe zagrożenia - klasyfikacja



Typowe zagrożenia - Problem złodzieja i strażnika

- **Zasada #1** Strażnik musi pilnować wszystkiego; złodziej może wybrać najsłabszy punkt według uznania
- **Zasada #2** Strażnik musi czuwać bez przerwy; złodziej podejmie próbę kradzieży w dogodnym dla siebie momencie
- **Zasada #3** Strażnik pilnuje tych zasobów, o których wie; złodziej będzie starał się znaleźć ukryte furtki i słabe miejsca w systemie

Typowe zagrożenia

Zagrożenie	Przykłady
Wstrzyknięcie kodu SQL	Dołączenie polecenia DROP TABLE do tekstu wpisywanego do pola wejściowego
Skrypty między witrynami	Używanie złośliwych skryptów po stronie klienta do wykradania plików cookie
Manipulowanie ukrytym polem	Złośliwa zmiana wartości ukrytego pola
Podsłuch	Używanie snifferów do wykradania haseł i plików cookie z ruchu na nieszyfrowanych łączach
Przejęcie sesji	Używanie skradzionego pliku cookie identyfikatora sesji do uzyskania dostępu do stanu czyjejś sesji
Fałszowanie tożsamości	Używanie skradzionych plików cookie formularzy uwierzytelniania w celu udawania innego użytkownika
Ujawnianie informacji	Umożliwianie klientowi obejrzenia stosu, gdy wystąpi nieobsługiwany wyjątek

Typowe zagrożenia – niesprawdzanie danych wejściowych

- **Przepełnienie bufora (ok. 30% wszystkich błędów)**

10101101101101011011	10101101101101011011101000101001010

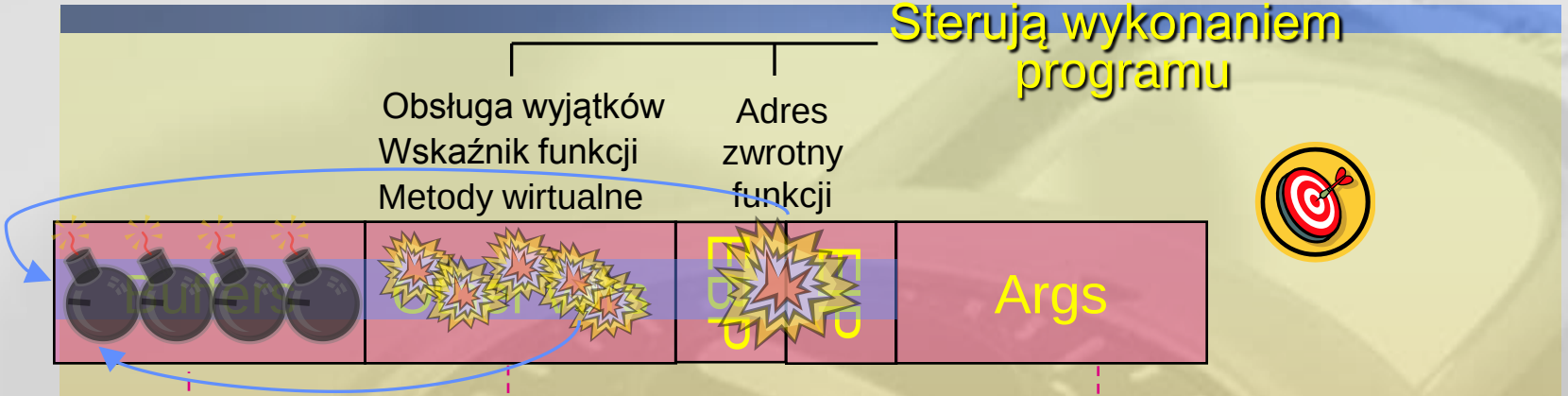
- **Manipulacja zapytaniami w SQL**

Blake	Blake’ drop table foo --

- **Cross-Site Scripting**

Blake	<script>var i=document.cookie</script>

Typowe zagrożenia – Przepełnienie bufora



Own3d!

```
void func(char *p, int i) {
  int j = 0;
  CFoo foo;
  int (*fp)(int) = &func;
  char b[128];
  strcpy(b,p);
}
```

Problem pojawi się gdy *p wskaże na więcej danych niż b a program działa z podwyższonymi uprawnieniami

Typowe zagrożenia – iniekcja SQL

```
SqlConnection conn = new SqlConnection();
conn.ConnectionString =
                        "data source=PYN-SQL;" +
                        "initial catalog=pubs;" +
                        "user id=sa;" +
                        "password=password;" +
                        "persist security info=True;“+
                        "packet size=4096";

conn.Open();

string strQuery;
strQuery = "select * from Users where UserName = '" +
            username.Text +
           "' and Password ='" +
            password.Text + "';";
```

Nie używana nazwa DNS

Uprzywilejowane konto sa

…z fatalnym hasłem

Brak kontroli danych wejściowych !

Typowe zagrożenia – iniekcja SQL i XSS

```
int rowCount = ds.Tables["Users"].Rows.Count;
if(rowCount > 0) // If we get back something...
{
        //...we must be succesfully logged in
        Session["LoggedIn"] = true;
        // Store the username in a session variable
        Session["username"] = username.Text;
...

<body>
        <asp:Label id="Username" runat="server">
                Label
        </asp:Label>
...
private void Page_Load(object sender,
{
...
        Username.Text = Session["username"].ToString();
}
```

Nazwa użytkownika z formularza

Przesyłana (cross-site script) na stronę WWW!

Typowe zagrożenia – Hasła zaszyte w kodzie

Uwaga

Veritas used a hard-coded password to authenticate to the agent. Patch is available at:

http://securityresponse.symantec.com/avcenter/security/Content/2005.08.12b.html

Niebezpieczne

Typowe zagrożenia – dobrze znane hasła

- Proszę wpisać nazwę użytkownika SA i hasło, a następnie kliknąć OK

UWAGA! Hasło znajduje się w dołączonej do programu instrukcji użytkownika

- Po zainstalowaniu programu hasłem administratora jest:
  - Admin, Administrator, Root, …
  - Puste

Niebezpieczne

Typowe zagrożenia – Brak kontroli dostępu

- Dostęp do każdego obiektu (pliku, klucza rejestru, semafora, itp.) kontroluje lista DACL
- Brak wpisów ACE na liście = brak dostępu do obiektu
- Pusta lista DACL = pełny dostęp do obiektu

Niebezpieczne

Typowe zagrożenia – wymaganie uprawnień administracyjnych

- Wiele programów wymaga dodatkowych uprawnień
- Winni są i producenci i Microsoft
- Przykład:

```
HANDLE hFile;
hFile = CreateFile("MYFILE.TXT",        // open MYFILE.TXT
                   FILE_ALL_ACCESS,     // full control
                   FILE_SHARE_READ,     // share for reading
                   NULL,                // no inheritance
                   OPEN_EXISTING,       // existing file
                   FILE_ATTRIBUTE_NORMAL,  // normal file
                   NULL);               // no attr. template
```

- W tym wypadku posłużenie się dodatkowymi uprawnieniami jest niczym nie uzasadnione

Typowe zagrożenia – niewspieranie aktualizacji zabezpieczeń systemu operacyjnego

- Nasi inżynierowie nie mieli okazji dokładnie przetestować działania programu z ta aktualizacją. Radzimy zainstalować ją w środowisku testowym i samodzielnie sprawdzić poprawność działania….
- Znane są problemy z działaniem programu po zainstalowaniu aktualizacji zabezpieczeń opisanej w artykule KB XYZ
- Użycie funkcji X jest niemożliwe po zainstalowaniu pakietu SP Y

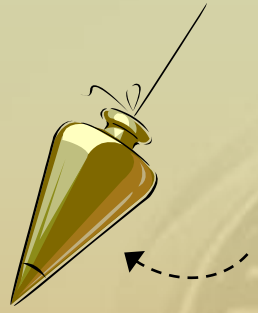
Bezpieczeństwo programów

- **Dlaczego zabezpieczenia na poziomie aplikacji są tak ważne?**
  - Ochrona granic zapewnia ograniczone zabezpieczenia
  - >70% ataków pochodzi z lokalnej sieci
  - Wiele obron opartych na zabezpieczeniach systemu nie jest związanych z konkretną aplikacją
  - Większość współczesnych ataków ma miejsce na poziomie aplikacji

Bezpieczeństwo programów

- **Z operacyjnego punktu winni są programiści ☺**
- **Co możemy zrobić?**
  - Korzystać z zaleceń producenta dotyczących zabezpieczeń jego programów
  - Opracować zestaw wskazówek dotyczących zabezpieczeń dla wszystkich używanych aplikacji
  - Włączyć je do polityki bezpieczeństwa firmy
  - Regularnie sprawdzać przestrzeganie tej polityki

Bezpieczeństwo programów - Wczoraj

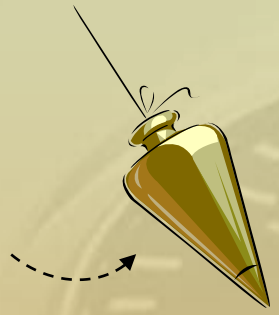


Funkcje, użyteczność

Bezpieczeństwo, ochrona inf. poufnych

- **Bardzo łatwe w użyciu**
- **Wiele sposobów ataku**
- **Funkcje „automagiczne”**
- **Tworzone „na wczoraj”**
- **Wszysko (w tym wirusy) „po prostu działa”**

Bezpieczeństwo programów - Dziś



Funkcje, użyteczność

Bezpieczeństwo, ochrona inf. poufnych

- Mniej sposobów ataku
- Często trudniejsze w użyciu
- Bezpieczne „za wszelką cenę”
- Dłuższy czas tworzenia
- Użytkownik nękany oknami dialogowymi

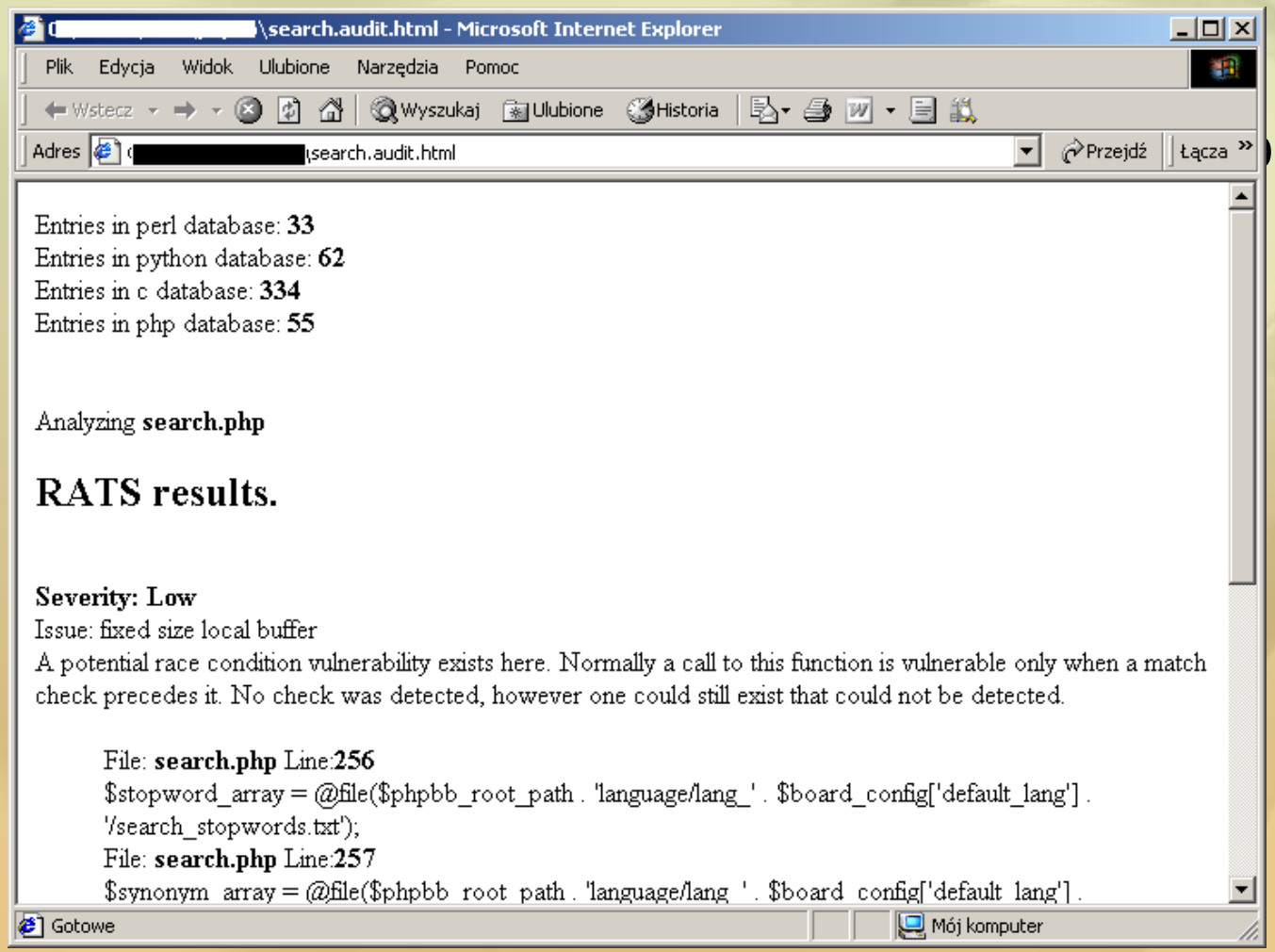
Bezpieczeństwo programów - Jutro

Funkcje, użyteczność

Bezpieczeństwo, ochrona inf. poufnych

- **Mniej sposobów ataku**
- **Bezpieczeństwo i ochrona informacji poufnych są podstawowymi wymaganiami projektowymi**
- **Mniej okien dialogowych**
  - bezpieczne wartości domyślne
  - możliwość dostosowania do własnych wymagań
- **Użyteczność *i* bezpieczeństwo!**

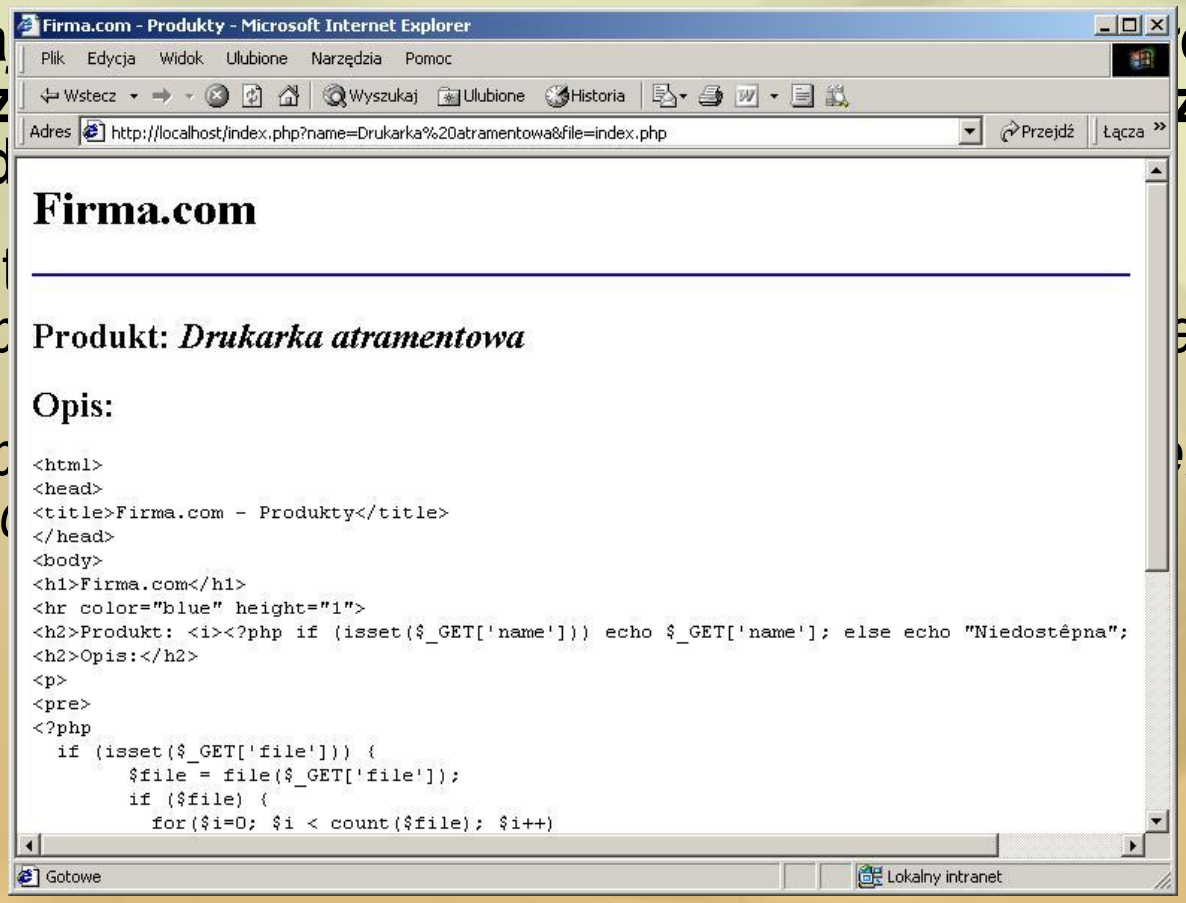
Aplikacje WWW



Aplikacje WWW

- **Nie ufa… …ędne (niebez… …zez Ciebie sprawd…**
  - Met… *http… …egoly*

    http… …ntowa&file= / ind…

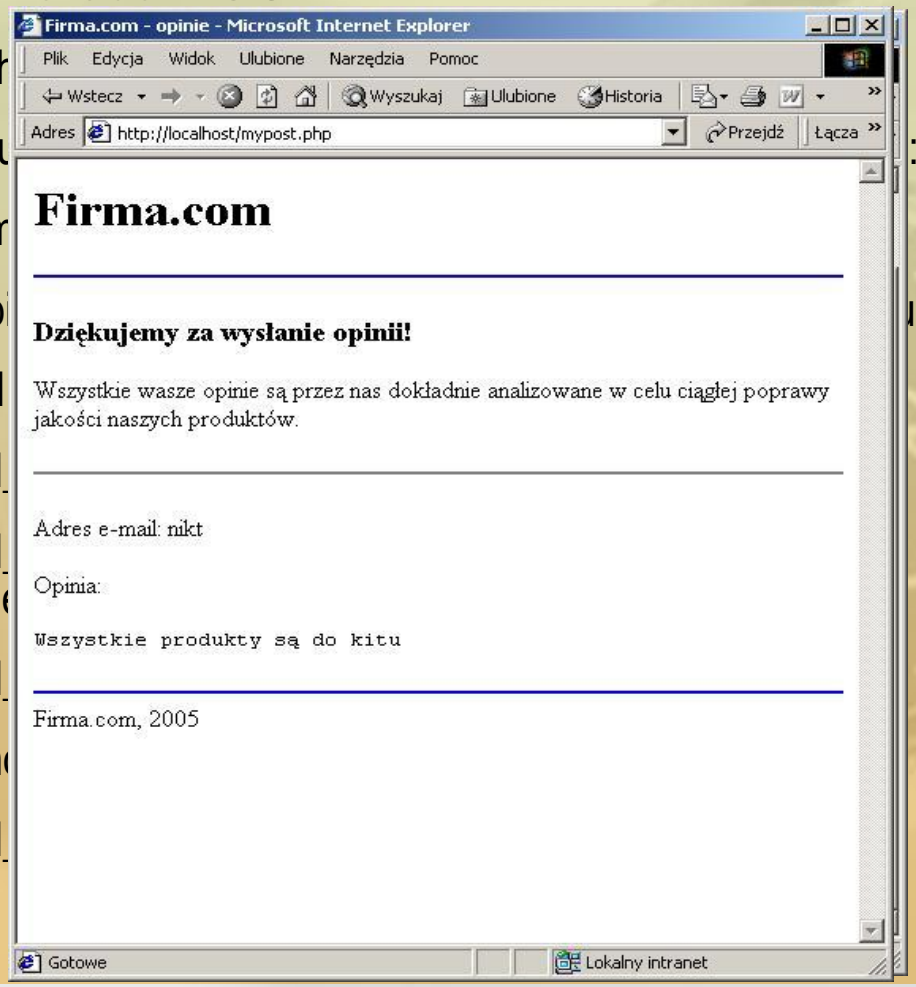


  - A gdyby wpisać *file= /etc/passwd* ?

Aplikacje WWW

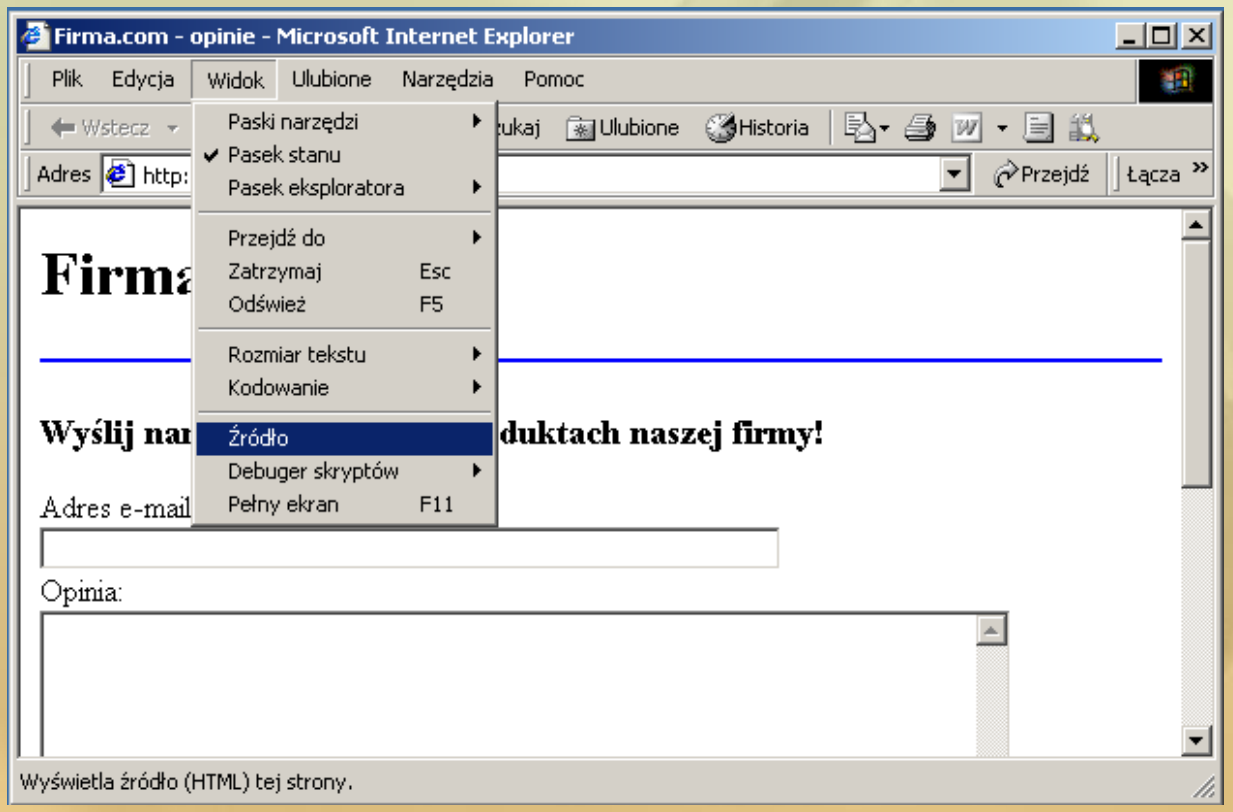
- Metoda POST:

```
<?ph
// Tu                                                    :
$em
$op                                                      ";
$url
curl_
curl_
    "
curl_
echo
curl_
?>
```



Aplikacje WWW

- „Ukrywanie” poufnych danych w kodzie HTML



Kontrolowanie uruchamianych programów

- **Wszystkie aplikacje do czasu wykazania, że są bezpieczne, należy traktować jako niebezpieczne**
  - Zasada minimalnych uprawnień (czy prog. jest niezbędny?)
  - Ograniczać dostęp zdalny
  - Izolować na dedykowanych serwerach aplikacji
    - Ograniczać serwery za pomocą zasad IPSec, tak aby zezwalały jedynie na taką komunikację, której aplikacja wymaga w jawny sposób
  - Monitorować sposób używania, i alarmować, gdy nastąpi zmiana użycia programu
    - Włączyć szczegółowe śledzenie
  - Żądać formalnej analizy zagrożeń, jeśli powyższe ograniczenia są zbyt surowe

Zasady ograniczeń oprogramowania

- Reguły identyfikujące pogram:
  - Certyfikatu
  - Mieszania
  - Ścieżki
  - Strefy Internetowej
- Akcja zezwalająca lub blokująca uruchamianie zidentyfikowanych programów
- Mechanizm niezależny od uprawnień użytkowników !

Bezpieczeństwo baz danych

- **Dopiero od kilku lat są de facto ogólnodostępne**
- **Przechowują kluczowe dla firm informacje**
  - Ich ujawnienie bywa gorsze od ich utraty
- **SQL jest językiem interpretowanym**
  - Serwer wykona każdą poprawną instrukcję +
  - Instrukcje mogą być dynamicznie konstruowane =
  - Użytkownik może zmienić oryginalną instrukcję …
  - o ile poprawność danych wejściowych nie zostanie sprawdzona

Model bezpieczeństwa serwera SQL 2005