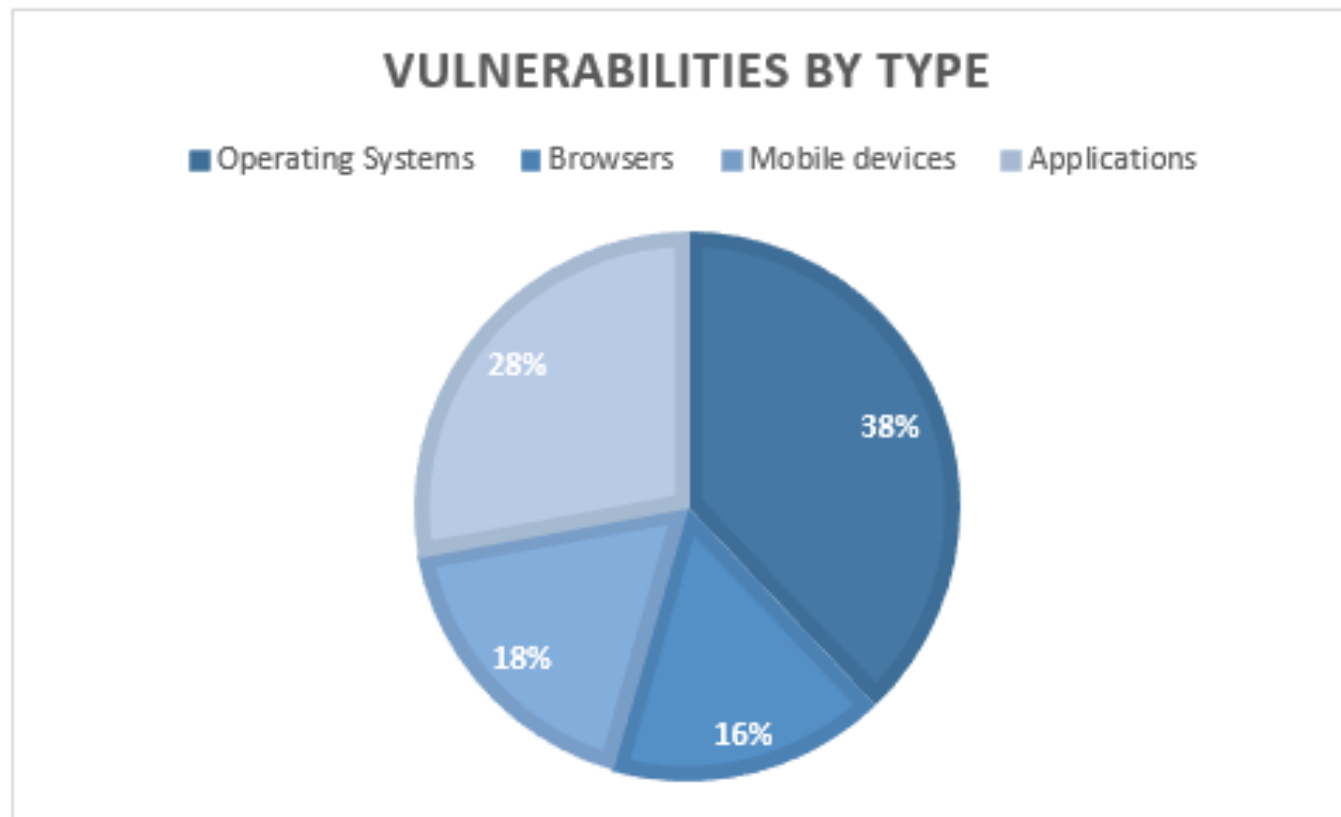


# Bezpieczeństwo aplikacji i usług sieciowych



# Statystyki CVE

## Common Vulnerabilities and Exposures database



# Zagadnienia

## 1. Bezpieczne środowisko aplikacyjne

- piaskownica: chroot/jail, Windows AppContainer
- wirtualizacja: Trusted Execution Environment (TEE), Virtualization Based Security (VBS)

## 2. Usługa WWW

- uwierzytelnianie HTTP
- tunele SSL/TSL
- aplikacje webowe

## 3. Poczta elektroniczna

- spam i phishing

# Bezpieczne środowisko

## Ograniczanie środowiska wykonania

### Idea – minimalizowanie szkód:

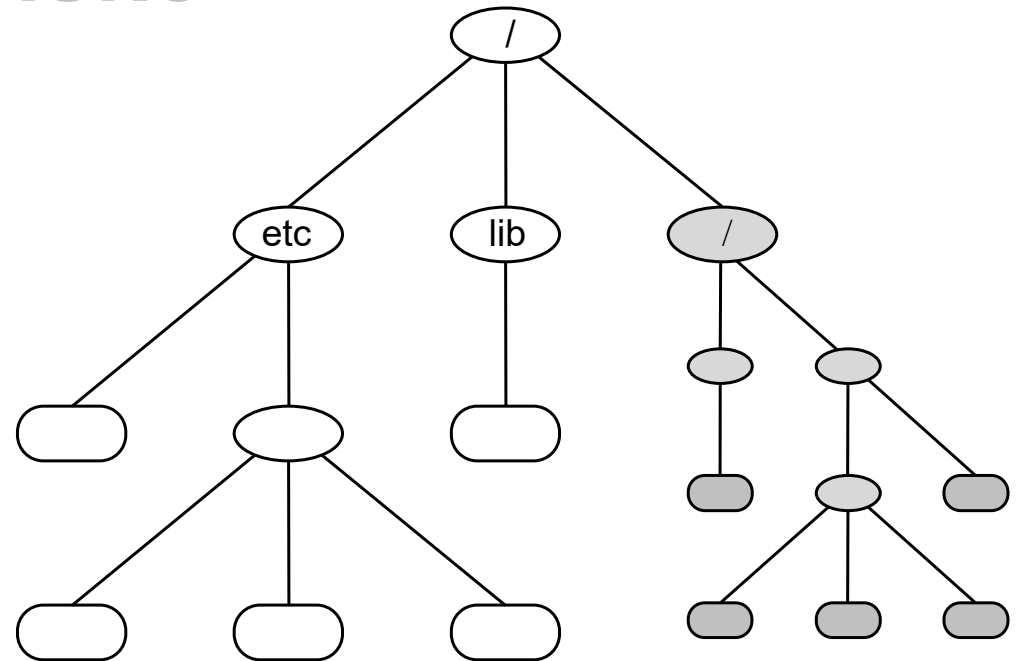
- zawsze uruchamiamy proces z najmniejszymi wystarczającymi uprawnieniami
- ograniczamy przestrzeń aktywności procesu (dozwolonych modyfikacji) do wybranego zawężonego fragmentu systemu
  - piaskownica (ang. *sandbox*)



# Bezpieczne środowisko

## Funkcja chroot() – Unix:

- uprzywilejowana funkcja systemowa ograniczająca proces do wydzielonego poddrzewa systemu plików
- blokuje jedynie dostęp do plików
- proces nie może otworzyć (w tym utworzyć) pliku poza ograniczonym obszarem
- chociaż może dziedziczyć deskryptory wskazujące na pliki spoza tego obszaru
- należy zainstalować odpowiednie pliki i katalogi potrzebne programowi i używanym przez niego bibliotekom (na ogół bardzo ograniczone fragmenty /etc, /lib czy /usr/lib)



# Bezpieczne środowisko

## Funkcja chroot() – problemy:

### Wciąż pozostają problemy – większość dotyczy DoS:

- dysk może się przepełnić (np. zrzutami obrazu pamięci, plikami raportów) – możemy ograniczać programy do oddzielnej partycji
- przepełnienie pamięci – proces może zagarnąć tyle pamięci, że zablokuje to urządzenie z plikiem wymiany – przeważnie możemy ograniczać użycie pamięci
- zużywanie czasu procesora – mamy do dyspozycji polecenie *nice*
- brak kontroli nad komunikacją sieciową

# Bezpieczne środowisko

## Funkcja chroot() – problemy:

### Polecenie chroot:

- polecenie chroot musi znajdować się w piaskownicy
- i wymaga uprawnień administracyjnych
- uruchomiony proces wykonuje się z uprawnieniami root-a
- potencjalne luki umożliwią ucieczkę z piaskownicy (uprawnienia root-a)
- inne narzędzia – np. chrootuid, jail (FreeBSD) – przed wywołaniem funkcji chroot() pozwalają zmienić UID oraz GID:

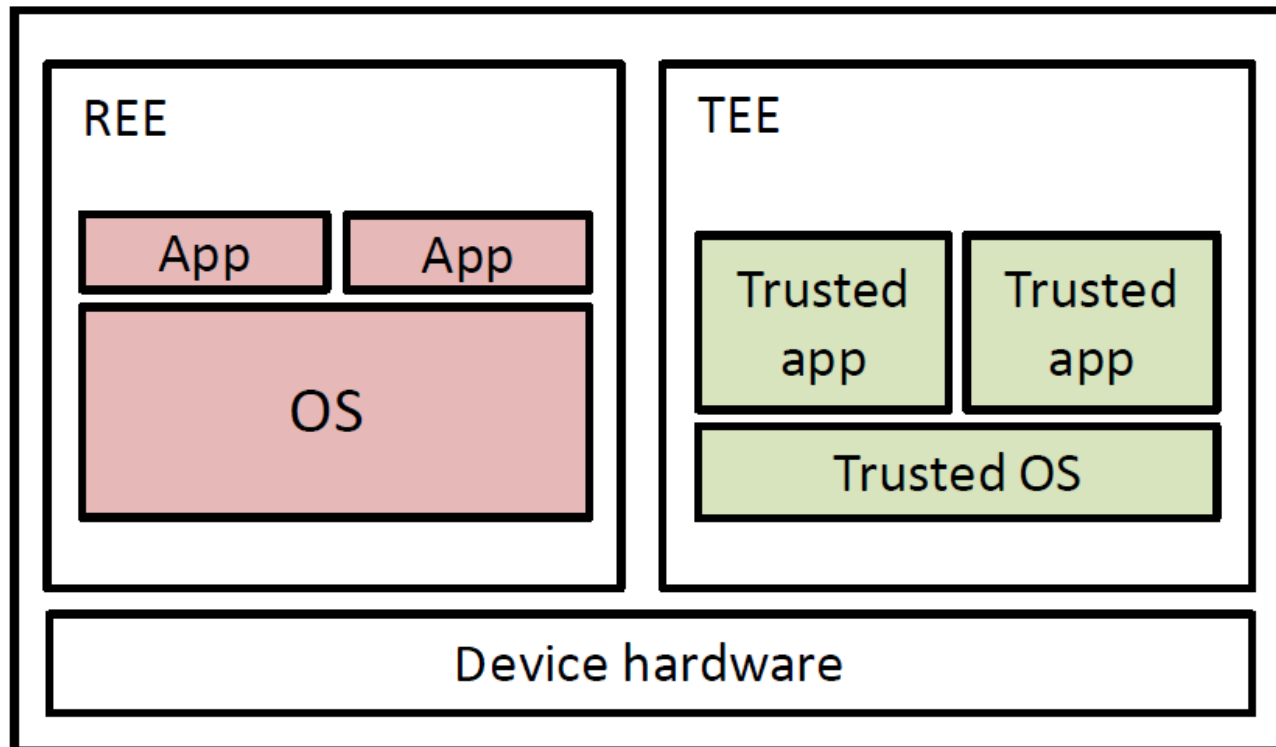
```
jail -u nobody -g www -l /tmp/jail.log -d / /usr/apache /bin/httpd
```

# Bezpieczne środowisko



## Trusted Execution Environment (TEE)

Intel TXT (VT-x), AMD Secure Processor (AMD-v), ARM TrustZone, ...

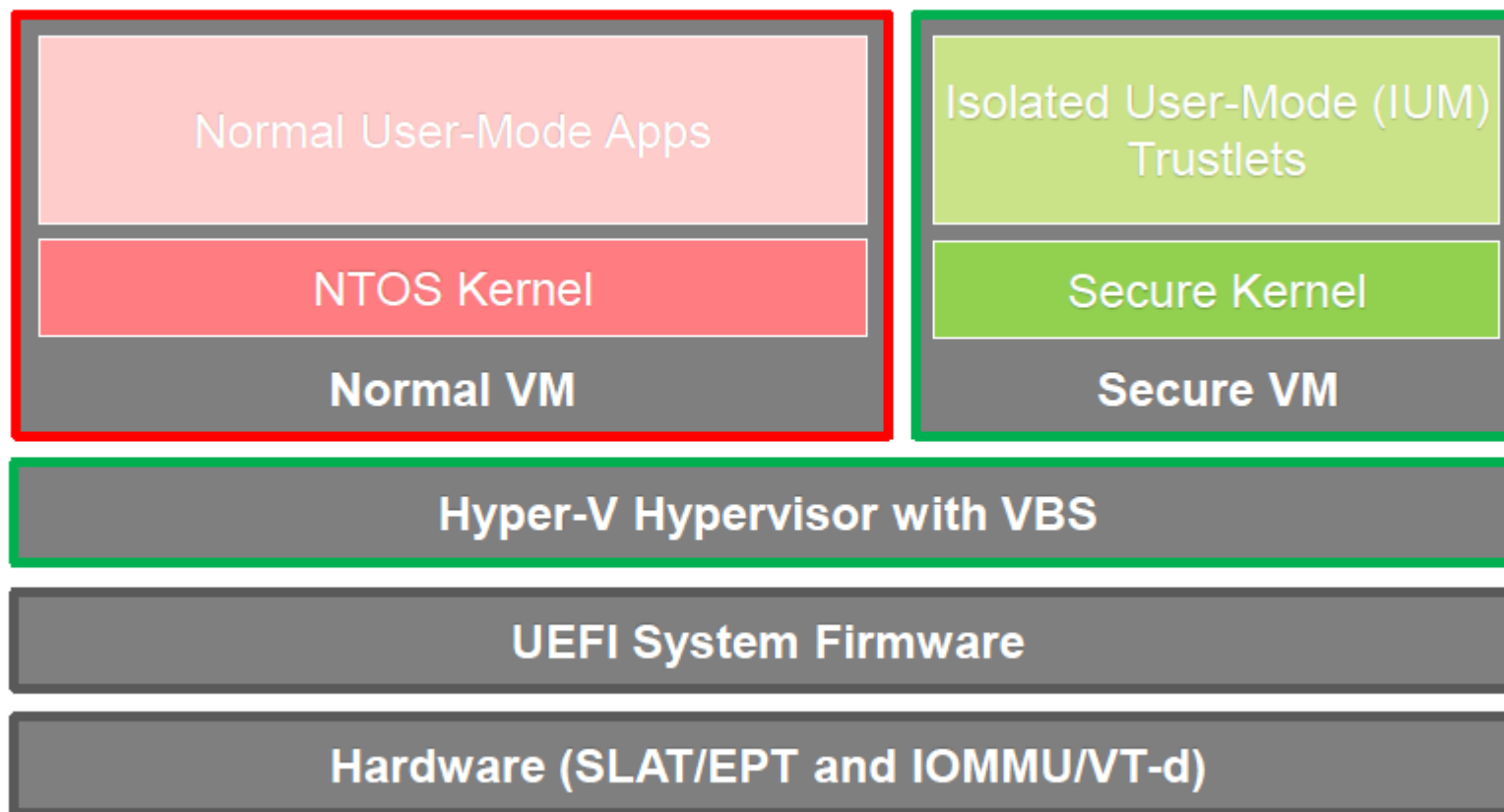




# Bezpieczne środowisko

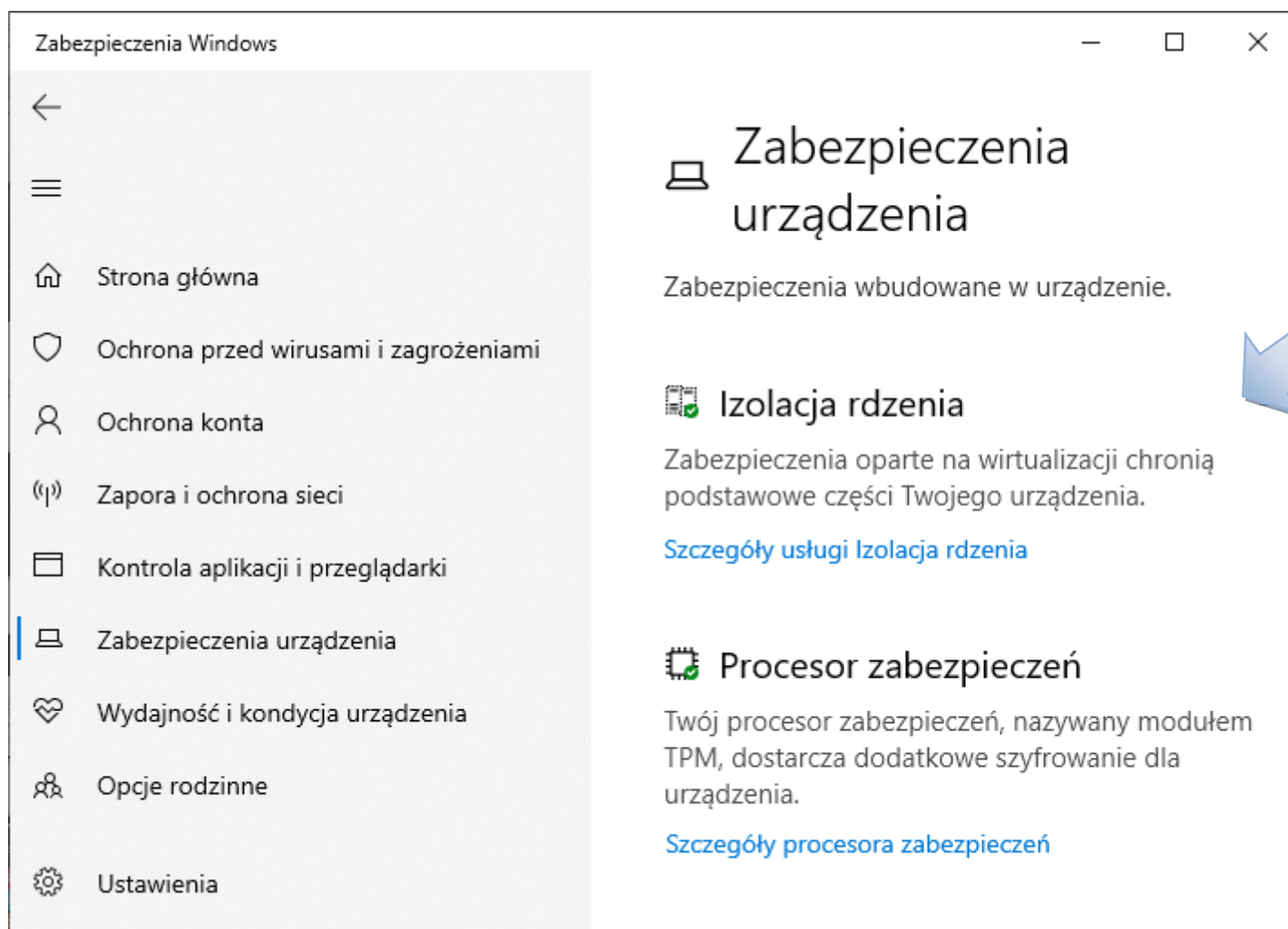
## Windows Virtualization-Based Security (VBS)

<https://docs.microsoft.com/en-us/windows-hardware/design/device-experiences/oem-vbs>



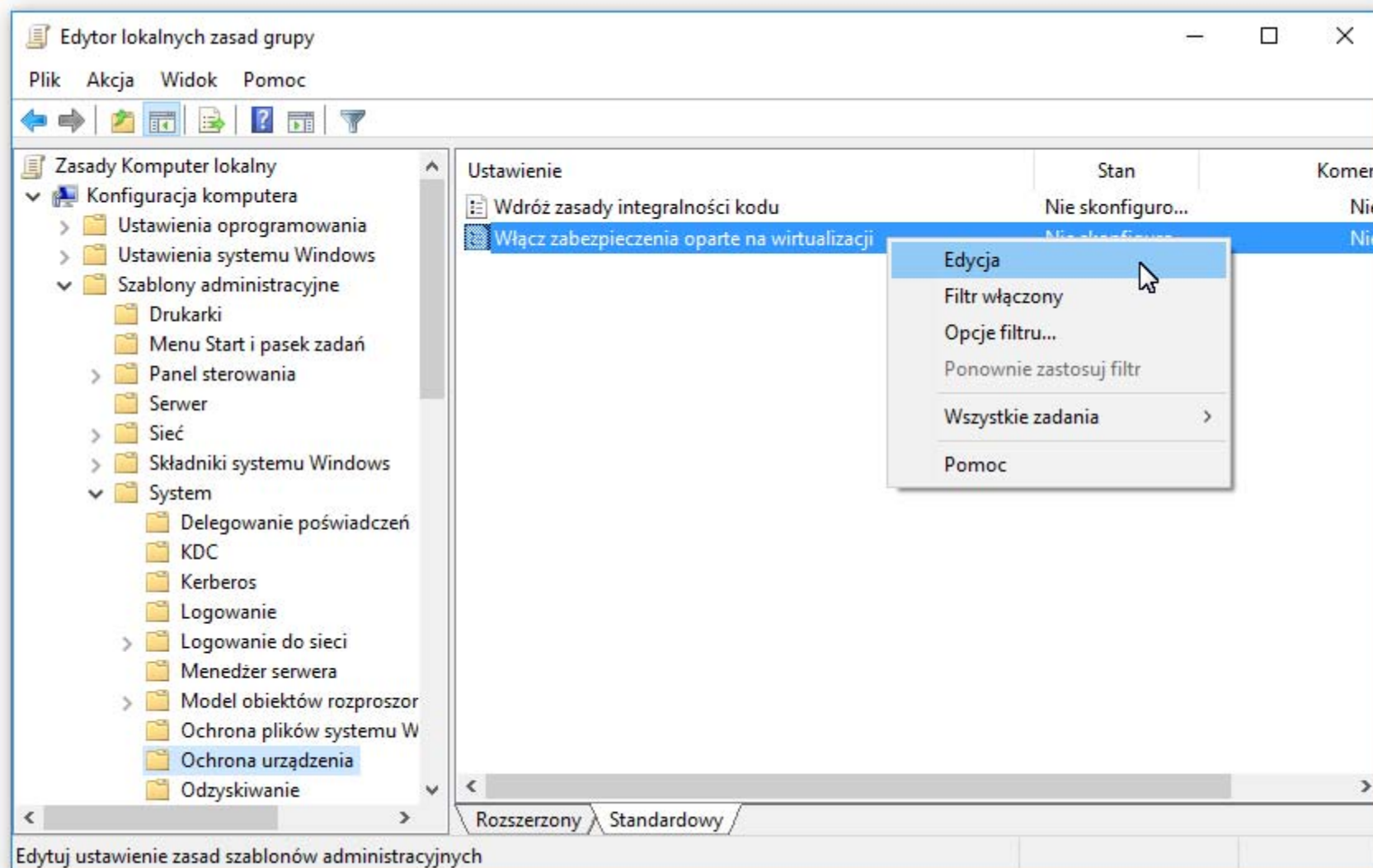
# Bezpieczne środowisko

## Windows Virtualization-Based Security (VBS)

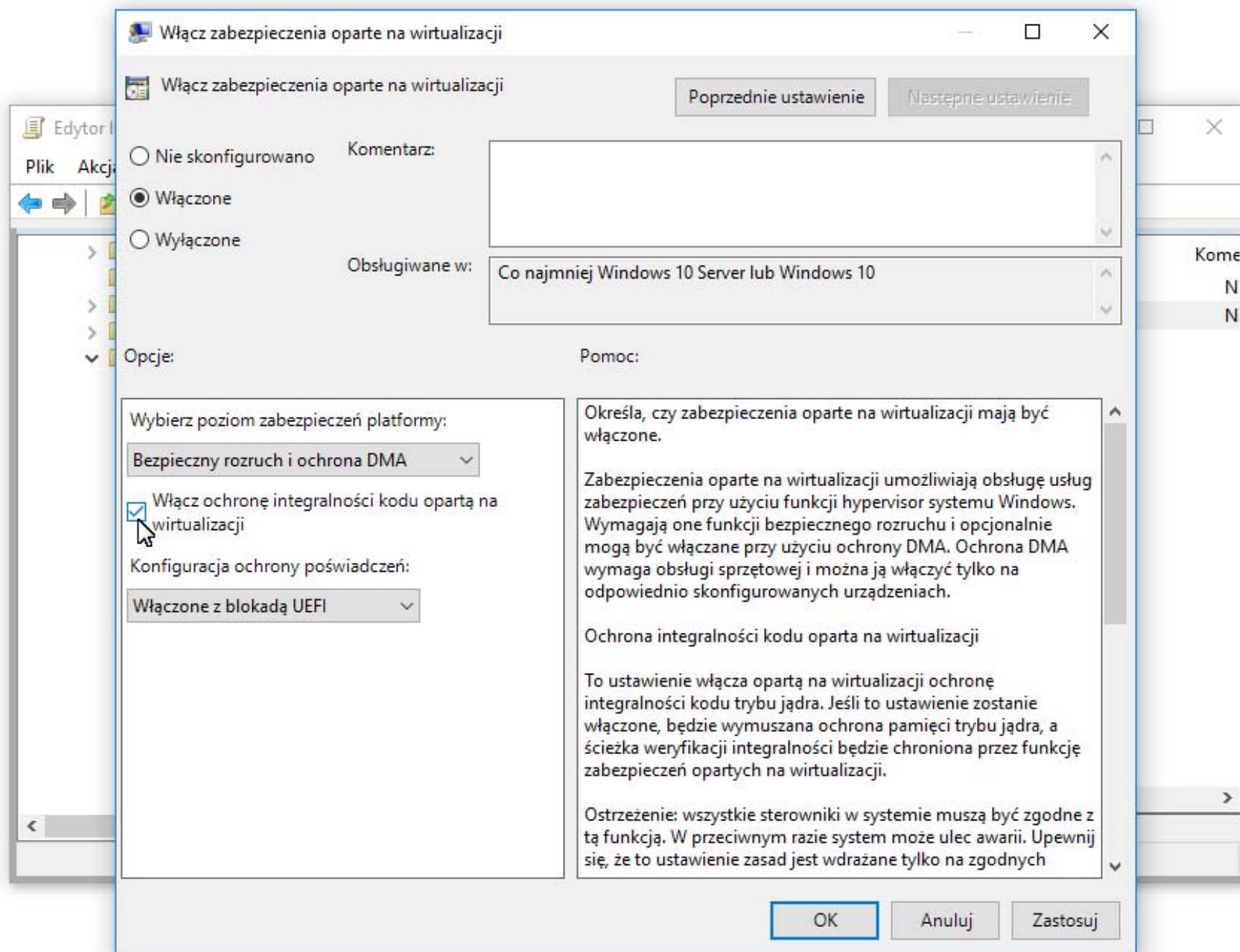


# Bezpieczne środowisko

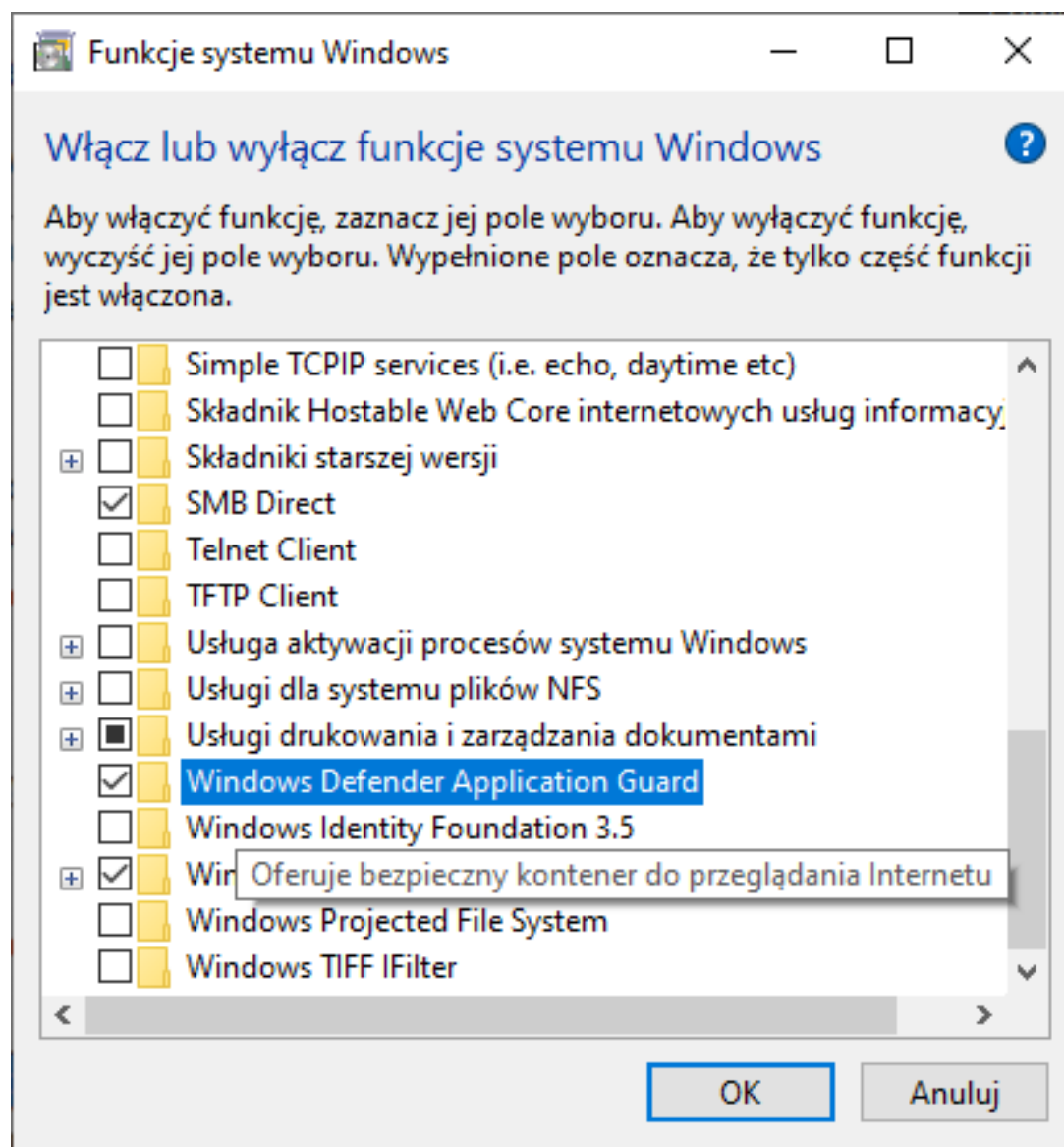
## Windows Virtualization-Based Security (VBS)



# Bezpieczne środowisko

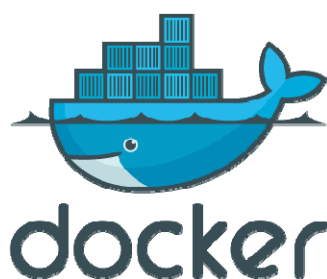


# Bezpieczne środowisko



# Bezpieczne środowisko

## Wirtualizacja systemu



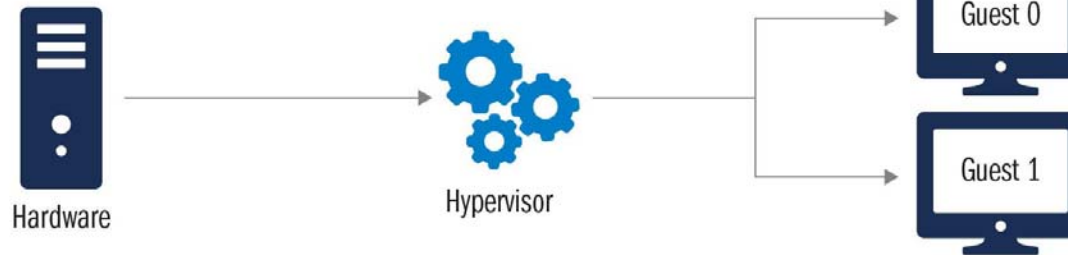


# Bezpieczne środowisko

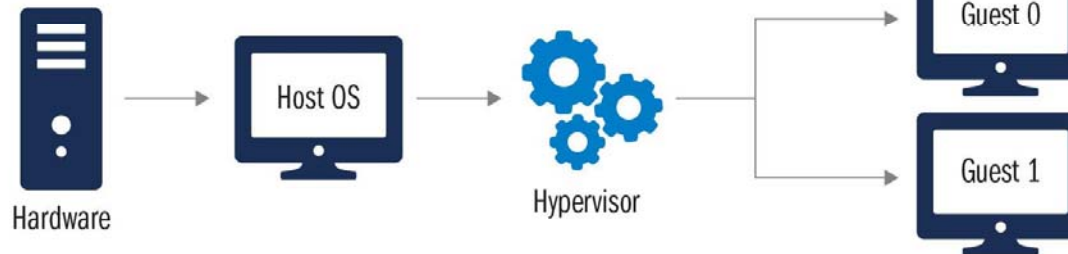
## Wirtualizacja systemu

### HYPERVISOR TYPES

#### TYPE 1 HYPERVISOR

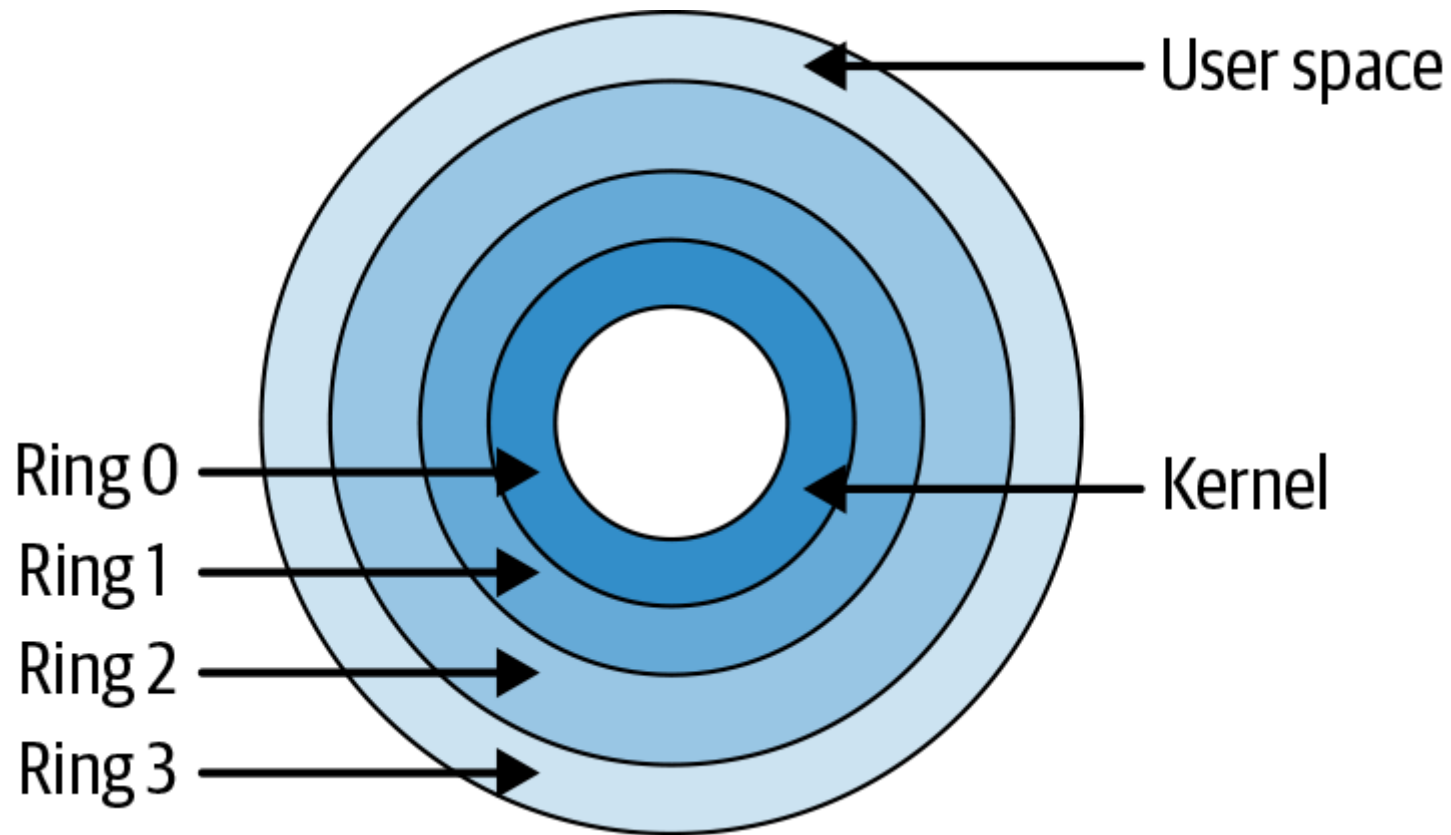


#### TYPE 2 HYPERVISOR



# Bezpieczne środowisko

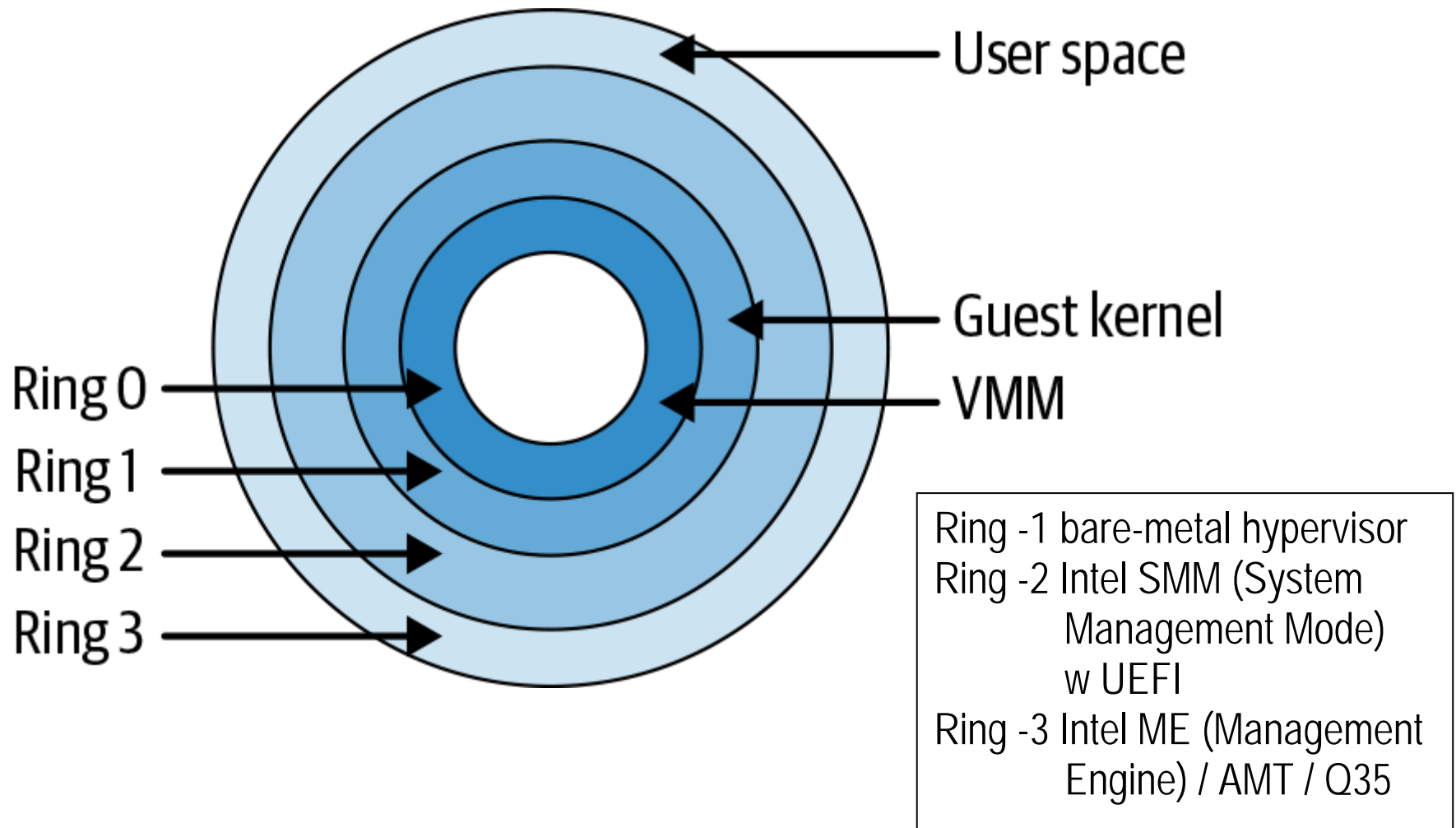
## Wirtualizacja systemu





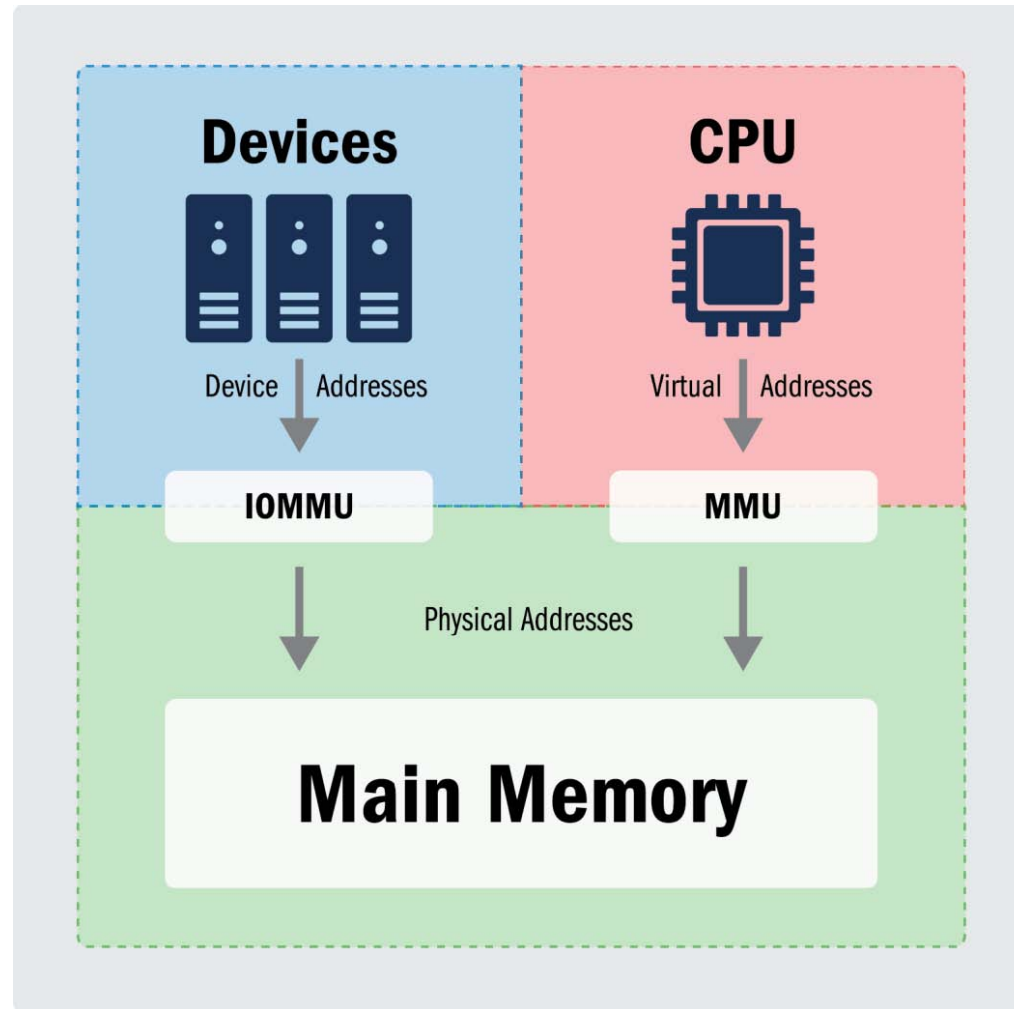
# Bezpieczne środowisko

## Wirtualizacja systemu



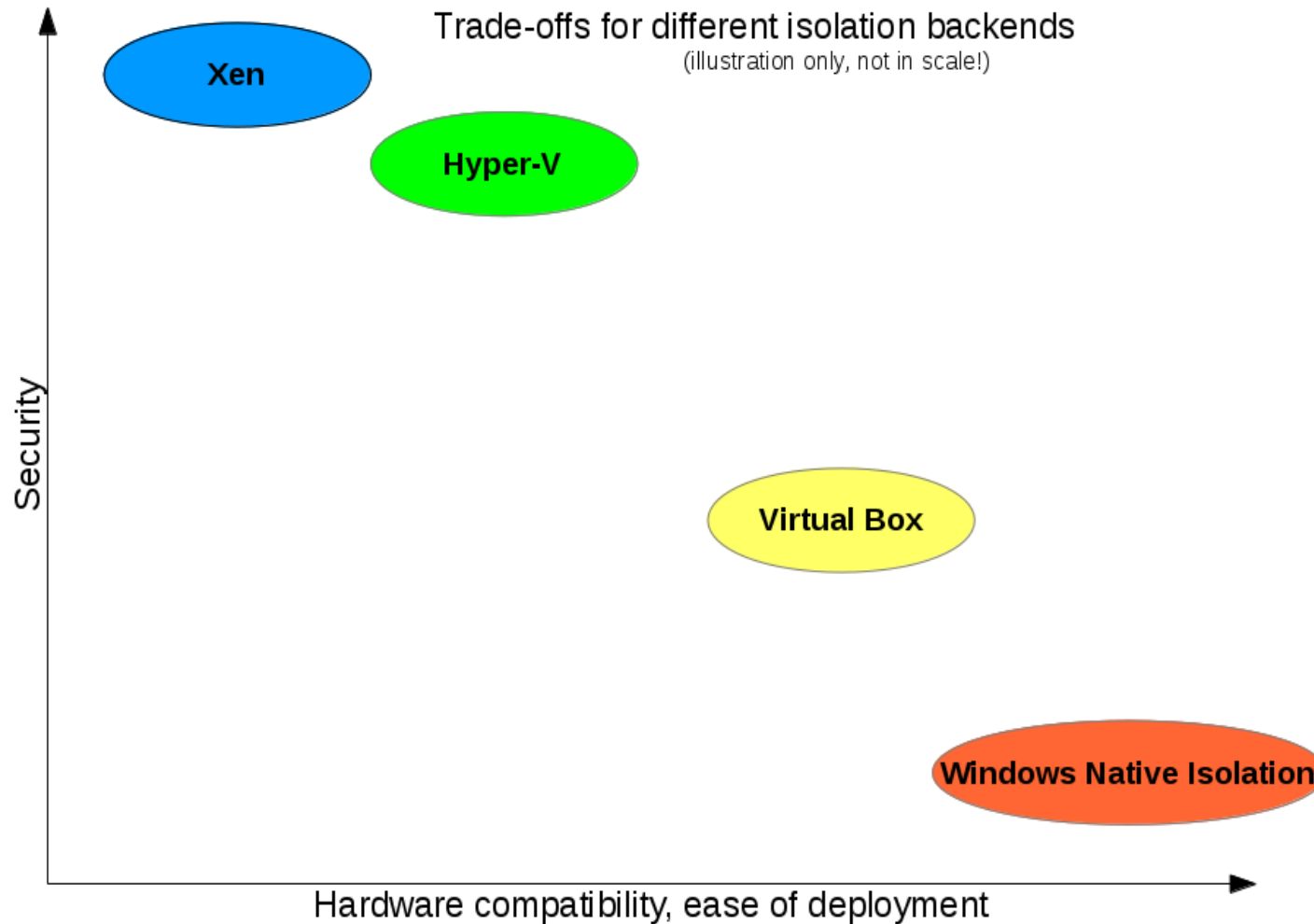
# Bezpieczne środowisko

## Wirtualizacja systemu



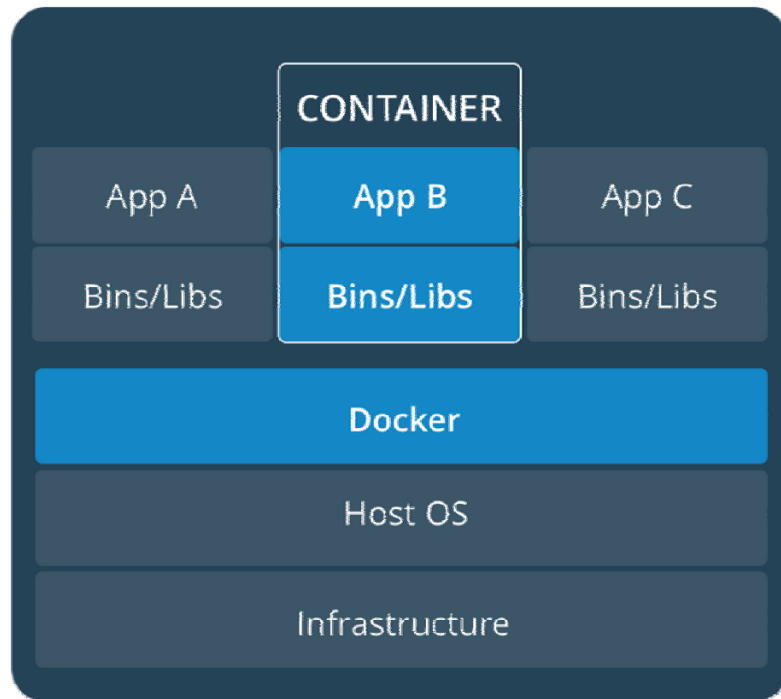
# Bezpieczne środowisko

## Wirtualizacja systemu

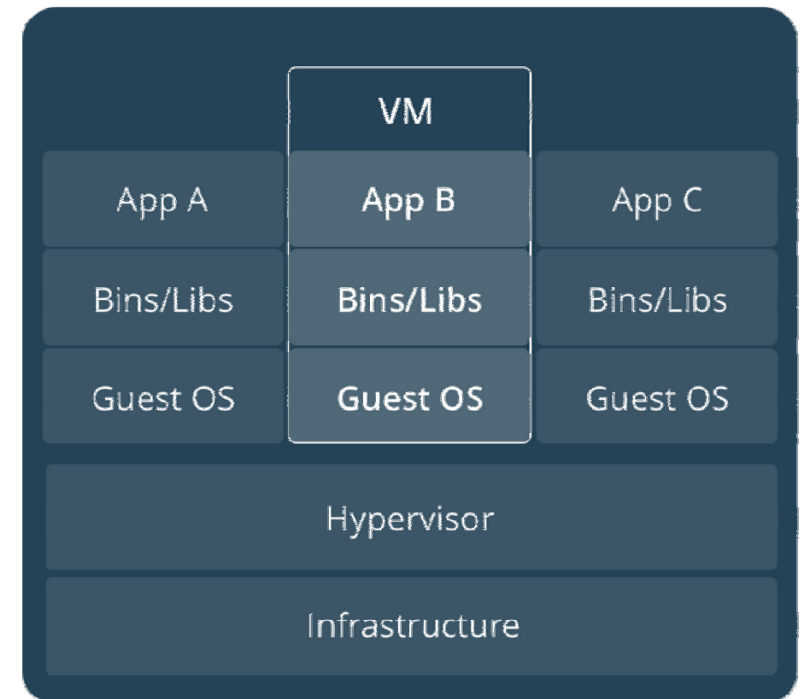


# Bezpieczne środowisko

## Kontenery



CONTAINERS

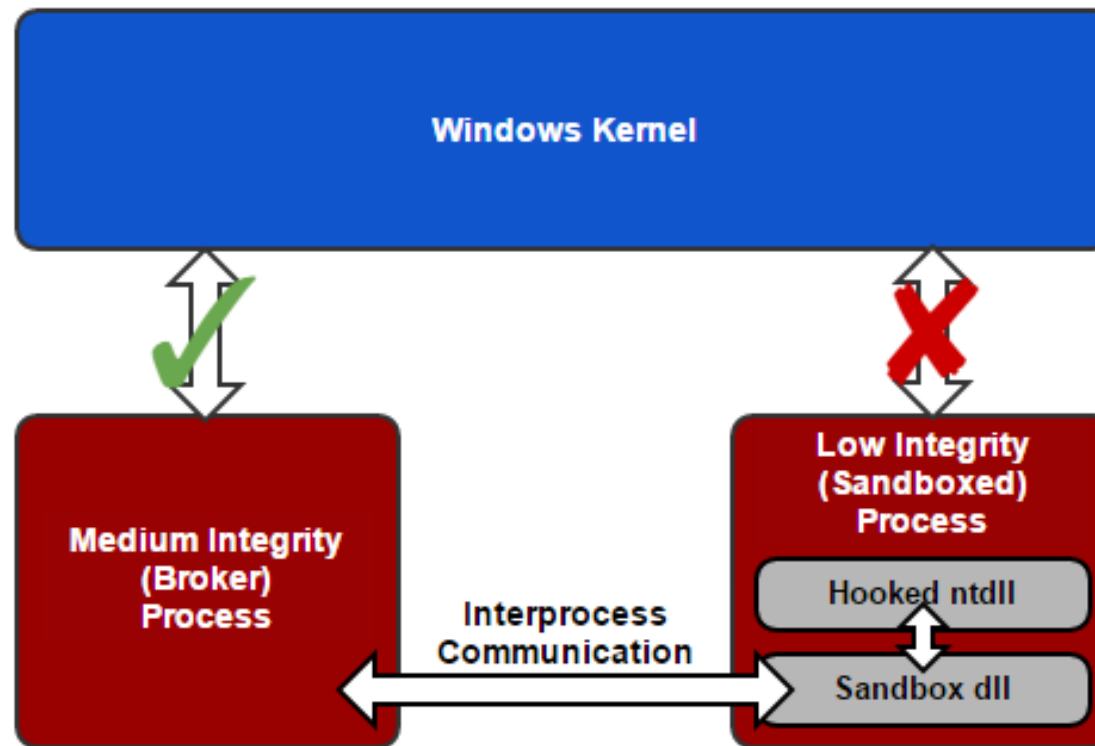


VIRTUAL MACHINES

# Bezpieczne środowisko

## Windows AppContainer

- piaskownica aplikacji UWP (\*.Appx)



# Bezpieczne środowisko

## Windows AppContainer

Process Explorer - Sysinternals: www.sysinternals.com

File Options View Process Find Handle Users Help

Process	PID	CPU	Private Bytes	Working Set	Description	Company Name	Integrity
csrss.exe	1820	< 0.01	3,124 K	4,260 K	Client Server Runtime Process	Microsoft Corporation	System
wininit.exe	1904		936 K	676 K	Windows Start-Up Application	Microsoft Corporation	System
services.exe	1956	< 0.01	5,728 K	6,872 K	Services and Controller app	Microsoft Corporation	System
svchost.exe	1296	< 0.01	3,200 K	5,116 K	Host Process for Windows Services	Microsoft Corporation	System
dllhost.exe	5568		1,316 K	2,332 K	COM Surrogate	Microsoft Corporation	System
BabylonHelper64.exe	8052	< 0.01	1,592 K	1,268 K	Support for 64-bit OS	Babylon	Medium
dllhost.exe	2164		1,768 K	1,692 K	COM Surrogate	Microsoft Corporation	High
winamp.exe	5192	0.36	39,668 K	50,792 K	Winamp	Nullsoft, Inc.	Medium
WWAHost.exe	804	3.30	189,184 K	249,304 K	Microsoft WWA Host	Microsoft Corporation	AppContainer
RuntimeBroker.exe	7404	0.01	10,004 K	20,744 K	Runtime Broker	Microsoft Corporation	Medium
svchost.exe	1228	< 0.01	7,152 K	8,284 K	Host Process for Windows Services	Microsoft Corporation	System

WWAHost.exe:804 Properties

Environment Job .NET Assemblies .NET Performance Strings

Image Performance Performance Graph Disk and Network GPU Graph Threads TCP/IP Security

Image File

Microsoft WWA Host  
Microsoft Corporation

Version: 6.2.9200.16420  
Build Time: Thu Sep 20 05:45:22 2012  
Path: C:\Windows\System32\WWAHost.exe

Command line:  
"C:\Windows\system32\wwahost.exe" -ServerName:AppexNews.wwa

Current directory:  
C:\Program Files\WindowsApps\Microsoft.BingNews\_2.0.0.273\_x64\_\_8wekyb3d8bbwe\

Autostart Location:

<https://news.saferbytes.it/analisi/2013/07/securing-microsoft-windows-8-appcontainers/>



# Bezpieczne środowisko

## Windows AppContainer

- AppContainer SIDs, oddzielny poziom izolacji MIC (untrusted = 0)

The screenshot illustrates the Windows AppContainer security model. It shows a Windows Explorer window with the 'Sessions' folder expanded, highlighting a specific AppContainer SID: `S-1-15-2-508114518-3340871649-811464485-52...`. A red box highlights this SID in the file list. Another red box highlights the same SID in the 'Security' tab of the 'WinObj' dialog, under 'Group or user names'. The 'Permissions for Account' table shows that the account has full control over the object, including 'List', 'Add Object', 'Add Subdirectory', 'Read', and 'Write'.

Permissions for Account	Allow	Deny
List	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Add Object	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Add Subdirectory	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Read	<input type="checkbox"/>	<input type="checkbox"/>
Write	<input type="checkbox"/>	<input type="checkbox"/>

Below the dialog, a table shows the symbolic links for the object:

SymbolicLink	Section
Global	SymbolicLink
Local	SymbolicLink
Session	SymbolicLink

The bottom of the Explorer window shows the full path: `\Sessions\3\AppDataContainerNamedObjects\S-1-15-2-508114518-3340871649-811464485-526616082-4258465299-1774086546-1865468257`.

# Bezpieczne środowisko

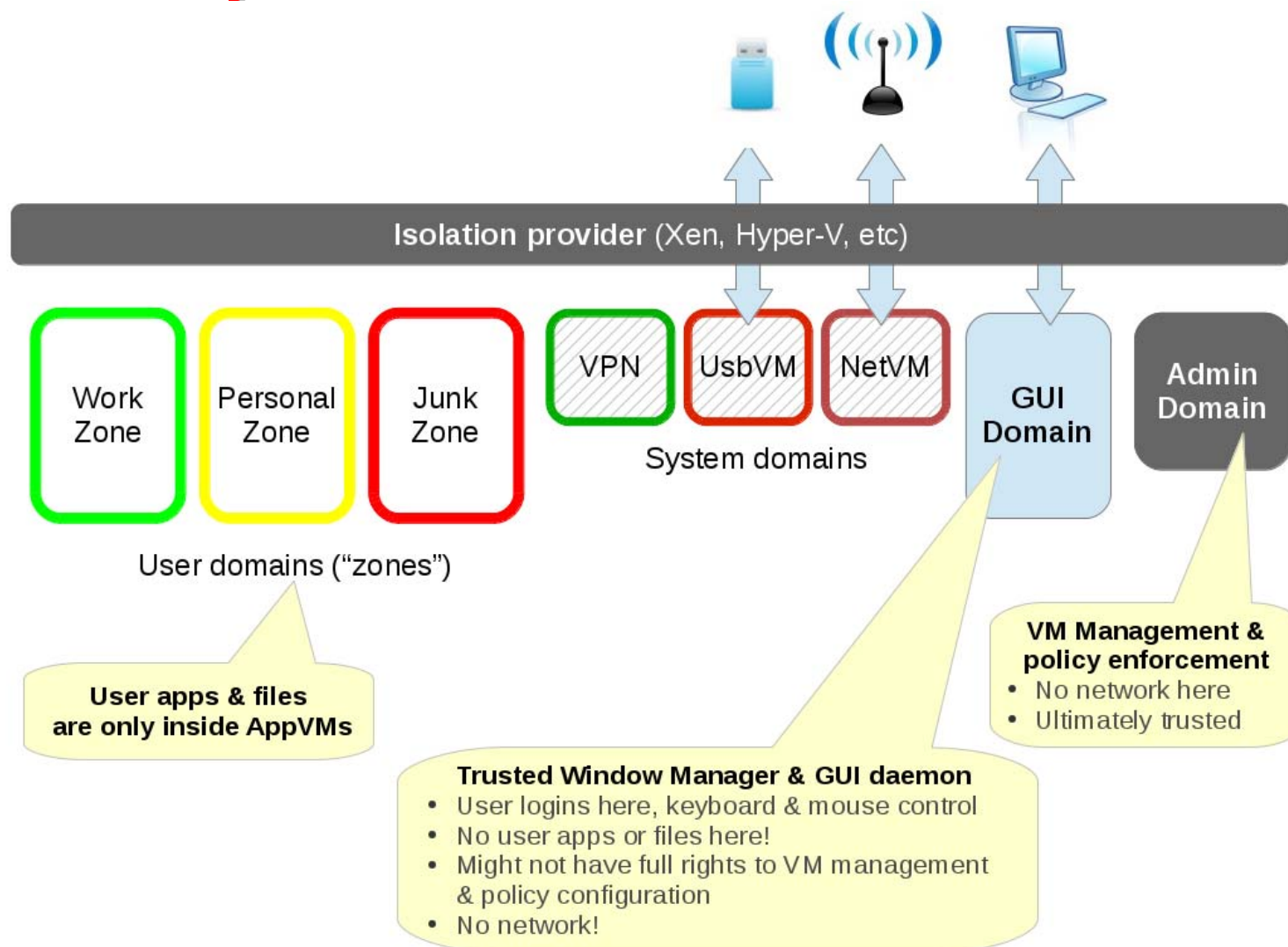
## Windows AppContainer

- oddzielne *namespace* obiektów jądra
- dostęp domyślnie tylko do swojego katalogu instalacyjnego
- dostęp do innych zasobów wymaga odpowiednich CAP (Capability SIDs)
- dodatkowa wirtualizacja:  
    %LocalAppData%\ → %LocalAppData%\Packages\<Container Name%\  
    → HKCU\Software\Classes\Local Settings\Software\Microsoft\Windows\CurrentVersion\AppContainer\Storage



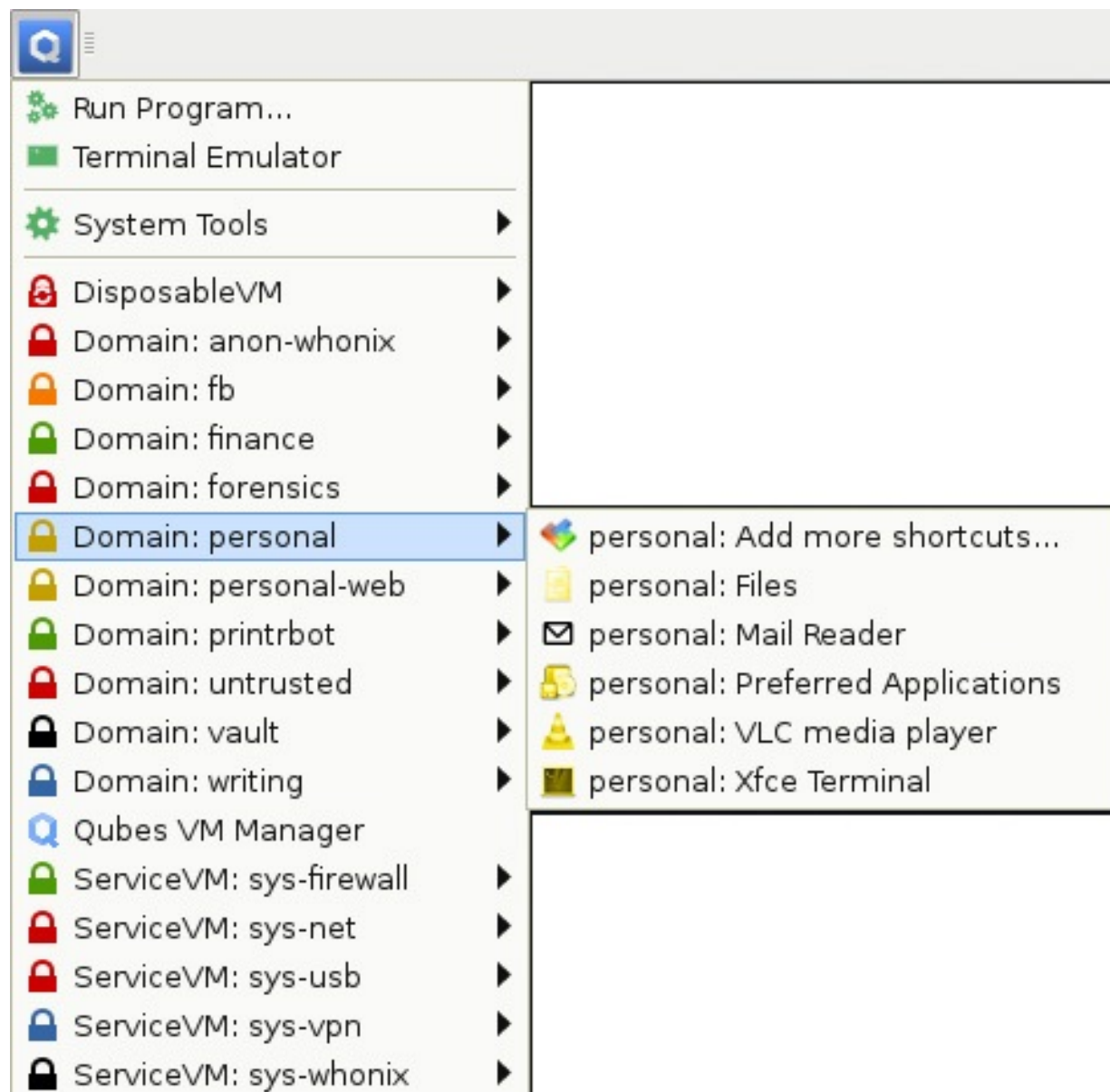
# Bezpieczne środowisko

## Wirtualne domeny



# Bezpieczne środowisko

## Wirtualne domeny



# Bezpieczne środowisko

## Cloud

<https://cloudsecurityalliance.org>

### Wirtualizacja:

1. IaaS (Infrastructure as a Service): CPU, net, mass memory
  2. PaaS (Platform as a Service): processing, networking, storage
  3. SaaS (Software as a Service): run applications & store data
- ...

# Bezpieczne środowisko

## Cloud



- CipherCloud gateway
- CryptDB proxy server

**Usługa WWW**

# Usługa WWW

## Uwierzytelnianie

### Uwierzytelnianie w protokole HTTP (RFC 1945)

- gdy klient żąda dostępu do zasobu podlegającego autoryzacji, serwer zwraca w odpowiedzi status 401 `Authorization Required`
- i w nagłówku `WWW-Authenticate`: wskazuje kontekst uwierzytelnia dla danego zasobu (domenę – *realm*)

```
HTTP/1.0 401 Authorization Required
WWW-Authenticate: Basic realm="Sesame"
Content-Type: text/html

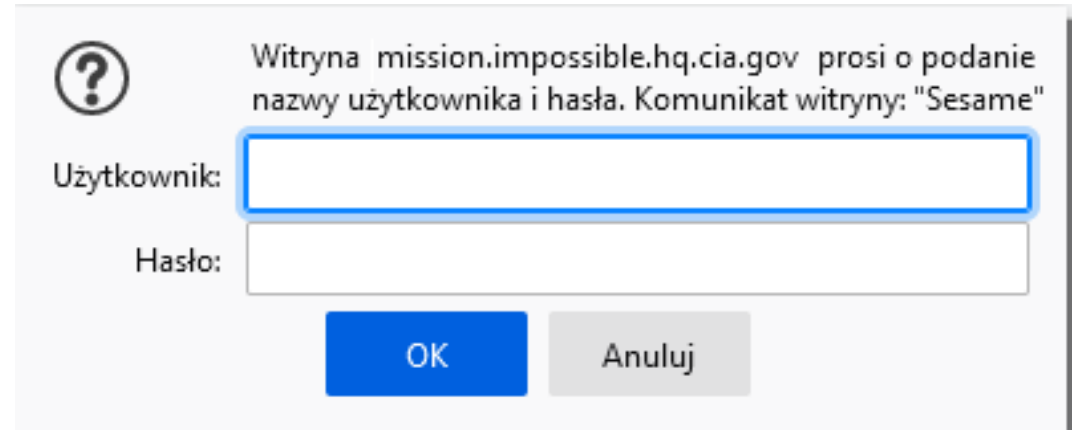
<HTML>
  <HEAD><TITLE>Error</TITLE></HEAD>
  <BODY><H1>Access Unauthorized.</H1></BODY>
</HTML>
```

# Usługa WWW

## Uwierzytelnianie

### Uwierzytelnianie w protokole HTTP (RFC 1945)

- przeglądarka wyświetla stosowne okno dialogowe, które pozwoli użytkownikowi na wprowadzenie danych uwierzytelniających
- po ich podaniu przeglądarka zapamięta je i automatycznie prześle do serwera na każde następne żądanie
- dane uwierzytelniające zostaną usunięte z pamięci z chwilą zamknięcia okna przeglądarki



A screenshot of a web authentication dialog box. At the top left is a question mark icon. To its right, the text reads: "Witryna mission.impossible.hq.cia.gov prosi o podanie nazwy użytkownika i hasła. Komunikat witryny: 'Sesame'". Below this, there are two input fields. The first is labeled "Użytkownik:" and the second is labeled "Hasło:". At the bottom, there are two buttons: "OK" (blue) and "Anuluj" (grey).

# Usługa WWW

## Uwierzytelnianie

### HTTP Basic Authentication (RFC 2616)

- klient w nagłówku `Authorization`: przekazuje token zawierający dane uwierzytelniające (*credentials*)
- token zawiera identyfikator podmiotu (np. username) skonkatenowany z hasłem  
`Aladdin:OpenSesame`
- token jest zakodowany Base64

```
GET /secret/agents_list/ HTTP/1.0
Host: mission.impossible.hq.cia.gov
Authorization: Basic QWxhZGRpbjppPcGVuU2VzYW1l
Cache-Control: max-age=0
```



# Usługa WWW

## Uwierzytelnianie

### Back-end

- Apache: *mod\_auth* (+ *mod\_auth\_mysql*, *mod\_authnz\_ldap*, *mod\_authnz\_pam*, ...)

```
<Location /secret/agents_list>  
    AuthName "Sesame"  
    AuthType Basic  
    AuthUserFile /etc/apache/passwd/users  
    AuthGroupFile /etc/apache/passwd/groups  
    Require group agents  
    Require valid-user  
</Location>
```

# Usługa WWW

## Uwierzytelnianie

### Back-end

- Apache: *mod\_auth* (+ *mod\_auth\_mysql*, *mod\_authnz\_ldap*, *mod\_authnz\_pam*, ...)

```
<Location /secret/agents_list>
  AuthName "Sesame"
  AuthType Basic
  AuthBasicProvider file ldap
  AuthUserFile /etc/apache/passwd/users
  AuthGroupFile /etc/apache/passwd/groups
  AuthLDAPURL ldap://ldap.mi6.gov/o=mi6
  Require group agents
  Require ldap-group cn=agents,o=mi6
  Require valid-user
</Location>
```

# Usługa WWW

## Uwierzytelnianie

### HTTP Basic Authentication (HTTP 1.0)



# Usługa WWW

## Uwierzytelnianie

### HTTP Digest Authentication (HTTP 1.1 → RFC 2617)

- typowo MD5 (choć teoretycznie jest to jedynie rekomendacja)

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest realm="Sesame",
                    nonce="b7102d2f0e8b11d0f600bfb0c093"
```

- klient konkatenuje identyfikator:realm:hasło i liczy skrót MD5 (=H1)
- dalej konkatenuje metodę HTTP (GET, POST,...) z URI i liczy skrót MD5 (=H2)
- ostatecznie konkatenuje H1:nonce:H2 i liczy wynikowy skrót MD5

```
Authorization: Digest username="Aladdin", realm="Sesame",
                    nonce="b7102d2f0e8b11d0f600bfb0c093",
                    uri="/secret/agents_list/",
                    response="ae49393a05397450978507c4ef1"
```

# Usługa WWW

## Uwierzytelnianie

### Inne mechanizmy uwierzytelniania

- np. Amazon S3 (Simple Storage Service):

```
Authorization: AWS AWS_AccessKeyID:Signature
```

lub OAuth (Open Authorization):

```
Authorization: OAuth
  oauth_version="1.0",
  oauth_consumer_key="a1191fd420e016432ed35d23",
  oauth_token="ad0d1c7a765c39c763177312d18e7e",
  oauth_signature_method="RSA-SHA1",
  oauth_signature="696304181e11c6eb8127ffea7e2",
  oauth_timestamp="1258328458",
  oauth_nonce="109843dea839120a"
```

# Usługa WWW

## Uwierzytelnianie

### Inne mechanizmy uwierzytelniania

- Apache + Kerberos

```
<Location /secret/agents_list>  
    AuthName "Sesame"  
    AuthType Kerberos  
    KrbMethodNegotiate On  
    KrbAuthRealms MI6  
    KrbLocalUserMapping On  
    Require valid-user  
</Location>
```

# SSL/TLS

## Tunel kryptograficzny usługi WWW

- warstwa sesji modelu OSI
- tunel kryptograficzny dla usług aplikacyjnych

https	443/tcp
nntp	563/tcp
ldaps	636/tcp
imaps	993/tcp
pop3s	995/tcp

- zintegrowany z niektórymi protokołami (ESMTP, HTTP/2)
- możliwe tunelowanie dowolnych portów (*port forwarding* → stunnel) i VPN

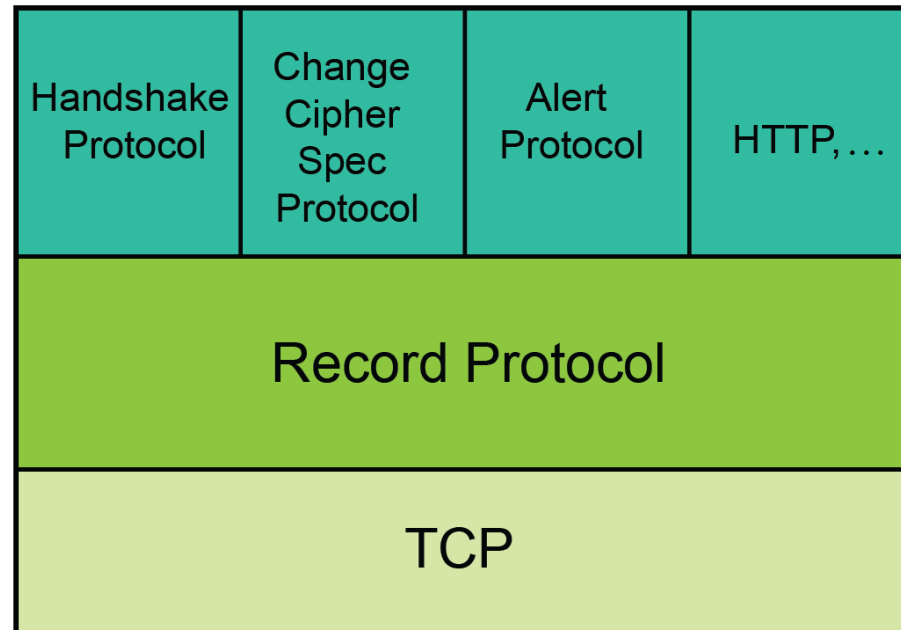
# SSL/TLS

## Protokoły składowe:

- Handshake Protocol – uzgadnianie sesji
- Change Cipher – wybór i zmiana metody szyfrowania
- Alert Protocol – sygnalizacja zdarzeń i błędów
- Record Protocol – tunelowanie PDU

aplikacyjnych:

- szyfrowanie symetryczne
- integralność
- kompresja





# SSL/TLS

## Handshake Protocol

1. klient wysyła do serwera komunikat `ClientHello` (wersja protokołu, identyfikator sesji, listę obsługiwanych szyfrów i metod kompresji)
2. serwer odsyła komunikat `ServerHello` (wersja protokołu, identyfikator sesji, wybrany szyfr i metodę kompresji oraz swój certyfiat X.509) oraz opcjonalnie ( $\rightarrow$  mTLS) żądanie certyfikatu klienta (wraz z losowym zawołaniem)
3. klient wstępnie uwierzytelnia serwer przez weryfikację autentyczności odebranego certyfikatu i w razie niepowodzenia przerywa połączenie
4. klient tworzy *pierwotny sekret główny* (*premaster secret*), który szyfruje kluczem publicznym serwera i wysyła do serwera
5. jeśli serwer żądał uwierzytelnienia klienta, to klient wysyła jednocześnie swój certyfiat oraz podpisane zawołanie odebrane wcześniej od serwera

# SSL/TLS

## Handshake Protocol

6. po ewentualnym uwierzytelnieniu klienta podpisanym zawołaniem, serwer deszyfruje pierwotny sekret główny i na jego podstawie uzyskuje *sekret główny* (*master secret*), podobnie czyni w tym czasie klient
7. z wygenerowanego sekretu głównego obie strony tworzą (zależny od ustalonego algorytmu szyfrującego) klucz sesji (lub klucze sesji – do szyfrowania i podpisywania)
8. klient i serwer wysyłają do siebie nawzajem zaszyfrowany kluczem sesji komunikat o zakończeniu fazy uzgadniania
9. protokół uzgadniania kończy się i (o ile weryfikacja komunikatów przebiegła pomyślnie) rozpoczyna się sesja TLS

# SSL/TLS

## Uwierzytelnianie

- jeśli serwer nie posiadałby klucza prywatnego odpowiadającego kluczowi publicznemu z certyfikatu zweryfikowanego przez klienta:
  - nie rozszyfruje poprawnie sekretu i nie wygeneruje tego samego klucza sesji co klient (→ krok 7.)
  - wówczas połączenie zostanie przerwane w kroku 9.
  - stąd klient ma pewność, że serwer jest tym, którego tożsamość poświadcza certyfikat (po weryfikacji jego autentyczności)
- jeśli klient nie posiadałby klucza prywatnego odpowiadającego kluczowi publicznemu z certyfikatu zweryfikowanego przez serwer:
  - serwer pobierze jego klucz publiczny z certyfikatu i rozszyfruje podpisane kluczem prywatnym klienta zawołanie (→ krok 6.)
  - nie otrzyma tego, które sam wysłał
  - zatem klient nie jest tym, czyją autentyczność poświadcza certyfikat

# SSL/TLS



# SSL/TLS

## Problemy

- newralgiczna weryfikacja certyfikatów – atak man-in-the-middle

→ <https://secure-resumption.com/>

w przypadku HTTPS można się w pewnym stopniu bronić mechanizmami

*HTTP Strict-Transport-Security* oraz *HTTP Public Key Pinning*

- istnieją też inne problemy implementacyjne dot. protokołu:

→ Breach (<http://breachattack.com>),

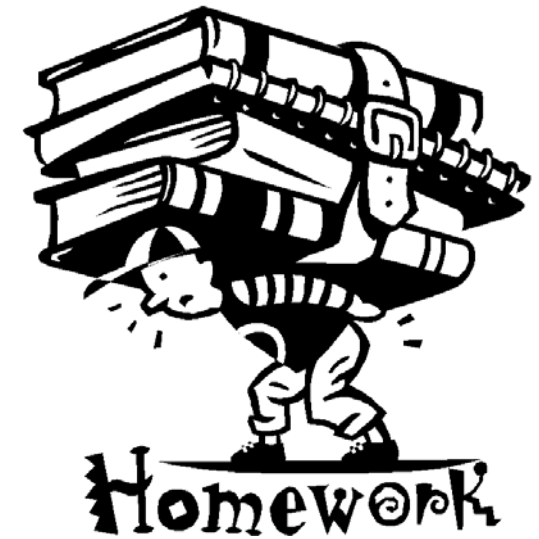
czy implementacji trybu CBC szyfrowania, np. Beast, Poodle

[https://en.wikipedia.org/wiki/Transport\\_Layer\\_Security#Attacks\\_against\\_TLS.2FSSL](https://en.wikipedia.org/wiki/Transport_Layer_Security#Attacks_against_TLS.2FSSL)

- problemy z implementacjami rozszerzeń (np. Heartbeat → Heartbleed)

# Usługa WWW

## Protokół HTTP/2 (RFC 7540, 2015 r.)



- bazujące na SPDY usprawnienia efektywności
- multipleksacja żądań/odpowiedzi HTTP (por. pipelining), priorytetyzacja
- format binarny
- kompresja nagłówek
- ładowanie z wyprzedzeniem inicjowane przez serwer (*server push*)
- wsparcie dla modelu subskrypcji (usługi *publish/subscribe*)
- ➔ ○ opcje bezpieczeństwa (min. TLS 1.2, minimum key size, itp.)
- ... HTTP/3 bazujący na QUIC (draft – listopad 2020)

# Usługa WWW

## Oprogramowanie

### Serwery WWW

- błędy programowe pozwalające na uruchomienie kodu w kontekście procesu serwera z jego uprawnieniami
- niedoskonała konfiguracja (również rozszerzeń) wprowadzająca luki bezpieczeństwa
- błędy implementacji SSL, S/MIME, np. problemy z losową generacją kluczy

# Usługa WWW

## Oprogramowanie

### Przeglądarki i aplikacje WWW

- luki (przeoczenia) w obsłudze zabezpieczeń, takich jak np. SOP (*Same Origin Policy*)
- potencjalnie niebezpieczne rozszerzenia i wtyczki
- WebView itp. silniki HTML/JavaScript
- pokaźny zbiór pseudoprotokołów (często źle udokumentowanych), np.: XHTML, MHTML, WebDAV, SOAP, ITS, MS-ITS, MS-HELP, MK, HCP,...
- mnogość technologii i frameworków

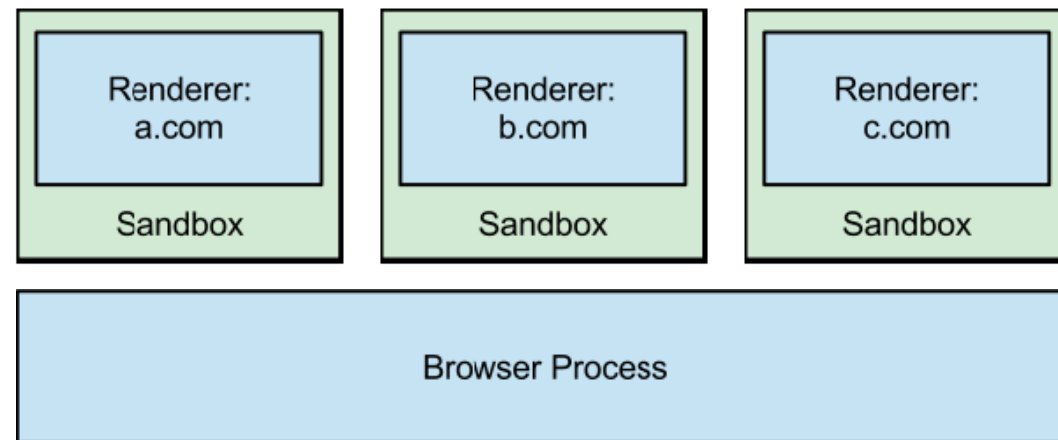


# Usługa WWW

## Man-in-the-Browser (MitB)

### Złośliwy kod

- odseparowane środowiska uruchomieniowe (*sandbox*)
- zaufane serwery źródłowe



### Języki skryptowe (ECMA Script)

- JavaScript
- ActionScript
- TypeScript

# JavaScript

## Zagrożenia

### Typowe luki bezpieczeństwa:

- *Cross-Site Scripting* (XSS) / *Cross Site Script Inclusion* (XSSI)
- *Cross-Site Request Forgery* (CSRF)

→ wykład Bezpieczne programowanie

- powszechność JavaScript:
  - HTML: www, e-mail, WebView, PDF, ...
  - XML / XHTML
  - Universal Windows Platform (UWP): C# + VB + JavaScript + AppContainer

# JavaScript

## Powszechność JavaScript:

### Pliki graficzne SVG:

```
<svg xmlns="http://www.w3.org/2000/svg">  
  <script>  
    <![CDATA[ alert("Hello!"); ]]>  
  </script>  
</svg>
```

# Cookies

## Zagrożenia

### Typowe luki bezpieczeństwa:

- naruszenia prywatności, uwierzytelniania i autoryzacji (→SessionID)

### **Cross-Site Cooking (XSC):**

- wymuszenie akceptacji cookies pochodzących z niezaufanych źródeł, a wykorzystujących podatności serwera/aplikacji www, np. luki w obsłudze uogólnień nazw domenowych w przeglądarkach
- pozyskiwanie cookies ważnych dla \*.domain.com, np. przez zatrucie DNS i przekierowanie pod sfałszowany adres <http://victim.domain.com>

# Cookies

## Ochrona

### Proste zabiegi często pomagają:

- flaga *Secure* – ciasteczko będzie wysyłane tylko przez HTTPS
- flaga *HTTPOnly* – do ciasteczka nie dobierze się JavaScript
- flaga *SameSite* – ogranicza XSS i CSRF

# Click-jacking i in.

## User Interface redress attack (UI redressing)

- skłonienie do kliknięcia w inne miejsce niż „wycelowane”
- HTML oferuje współcześnie bogaty asortyment wspierających technik
- np. rzeczywisty obiekt jest umieszczany w warstwie o atrybucie niewidzialna
- okienka pop-up, ramki, iFrame
- relatywnie niewinna odmiana: like-jacking (Facebook)
- cursor-jacking: zmiana współrzędnych wizualizowanego kursora
- podobne: keystrokes hijacking (np. wpisywanie hasła w nałożone niewidzialne pole)
- obrona na poziomie HTTP – nagłówek X-Frame-Options: deny



# OWASP

The Open Web Application Security Project

<https://www.owasp.org>



## browsersec

Browser Security Handbook

<https://code.google.com/p/browsersec>



## Gruyere

<https://google-gruyere.appspot.com>

# Poczta elektroniczna

## Podstawowe problemy

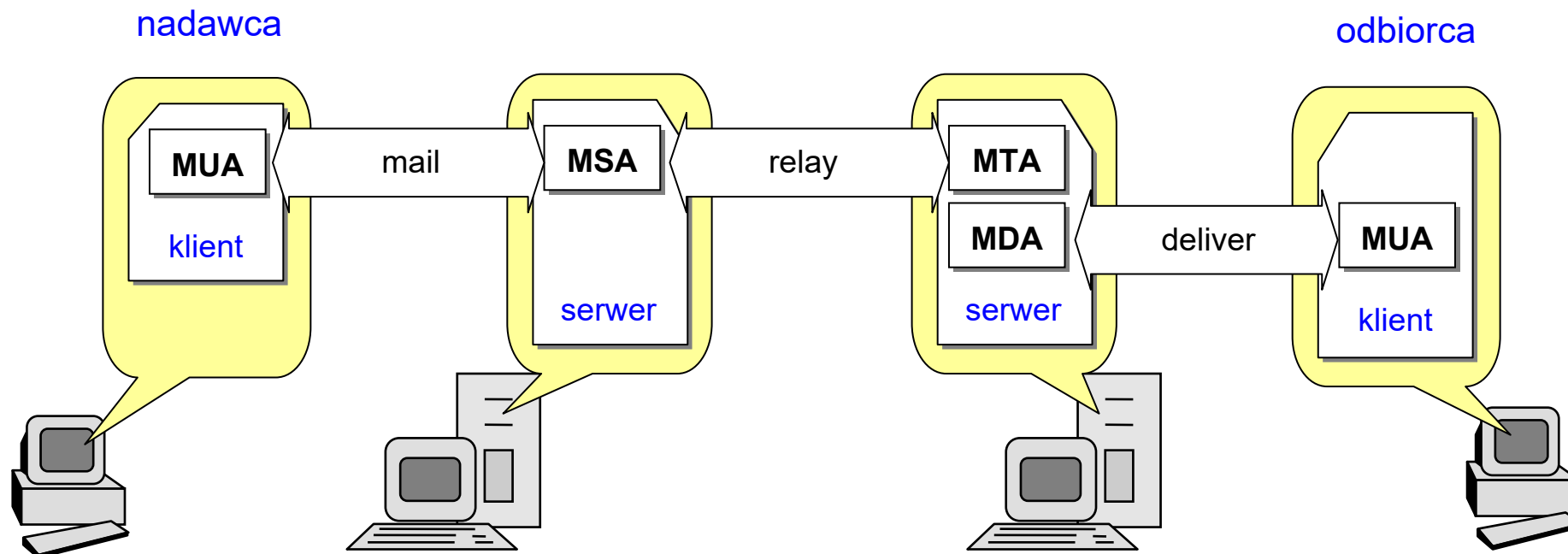
- niepożądane przesyłki (*spam*)
- niebezpieczne załączniki (wirusy)
- potwierdzanie dostarczenia
- naruszenie poufności / integralności / autentyczności





# Poczta elektroniczna

## SMTP (RFC 821)



**MUA** = Mail User Agent

**MSA** = Mail Submission Agent

**MTA** = Mail Transfer Agent

**MDA** = Mail Delivery Agent

# Poczta elektroniczna

## SMTP (RFC 821)

### Naiwna weryfikacja tożsamości MTA

- SMTP zawiera komendę **HELO**, której celem jest wymiana adresów domenowych serwerów:
- serwery naiwnie domniemają prawdziwość podawanych adresów

```
> HELO mi6.mil.uk  
< 250 secret-service.mil.uk
```

# Poczta elektroniczna

## ESMTP: Simple Authentication and Security Layer (SASL, RFC 2222)

- komenda **AUTH** uwierzytelnia metodą *challenge-response* (RFC 2554)

```
> EHLO mi6.mil.uk
< 250 secret-service.mil.uk says hello
< 250 AUTH CRAM-MD5 DIGEST-MD5 XOAUTH2

> AUTH CRAM-MD5
< 334 PeUxFRJoU0NnbmhWx3b29k9zb2Z0LmNvbT4=

> ZnJlZCA5ZTk1YWVlMDljNDBhhMGMyYjNiYmFlNzg2ZQ==
< 235 Authentication successful
```

- SASL może uwierzytelniać poszczególne przesyłki z osobna

```
> MAIL FROM:<e=m@hq.cia.mil> AUTH=e+3Dm@hq.cia.mil
< 250 OK
```

# Poczta elektroniczna

## ESMTP: współpraca z protokołem TLS (RFC 2487, RFC 3207)

```
> EHLO mi6.mil.uk
< 250 secret-service.mil.uk says hello
< 250 STARTTLS

> STARTTLS
< 220 Go ahead

> rozpoczęcie negocjacji parametrów sesji TLS

    dalsza komunikacja odbywa się w postaci zaszyfrowanej:

> .J...F..@.t..DOi.U.%2
< LU.....0.*.H%2.n.Ol..<..)B.$..
> f..0.*.H..0..,.....F.K.Kz...|E...
< !0...*.H.....po
> ...gai.....5.^.....Me:.....~.k...%+.Q.m...5..
  ...~.}.o.$.....}#.p.....b.....m.....0...*.H.
  .....Xu...,.....8.....'.m#.u
  ..MNA....V.....bS..~..3C.A.L...P.....H|.!.
  !_...Y.&k.N...\..d`U.Z.....s.....
```

# Poczta elektroniczna

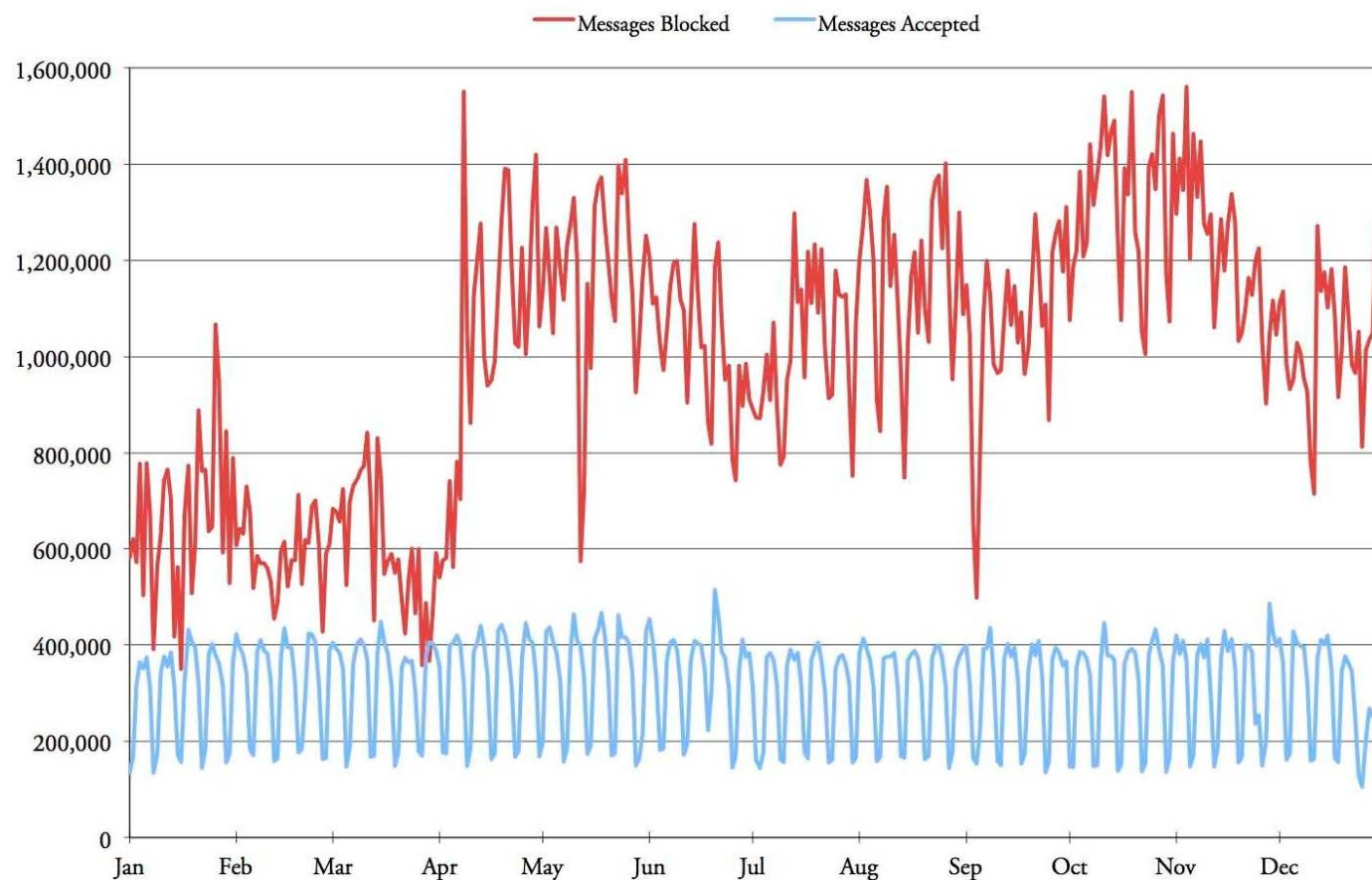




# Poczta elektroniczna

## Spam

natężenie spamu: 60% – 90% ruchu e-mail



Note: The number of messages accepted fluctuates with fewer messages on weekends and holidays.

Source: Office of Information Technology, 1/3/12



# Poczta elektroniczna

## Spam

*While the most important cost involved with spam is in human time—time spent reading, deleting, and devising ways to fight it—there's actually a huge environmental cost as well: to filter out the estimated 62 trillion junk emails sent in 2008, computers burned through enough electricity to power 2.4 million homes for the year, leading to 17 million metric tons of CO<sub>2</sub> emissions.*

# Poczta elektroniczna

## Ochrona anty-spamowa

### Filtracja

- czarne listy: adresy (konta i/lub domeny) nadsyłające rozpoznany spam
- konfiguracja dynamiczna (np. Spamhaus Black List [www.spamhaus.org/sbl/](http://www.spamhaus.org/sbl/))
- + białe listy: adresy jawnie wskazane za *bezpieczne* (zaufane domeny)

### Podstawowa klasyfikacja

- wiadomość *pożądana*  $\leftrightarrow$  *niepożądana*

### Klasyfikacja 3-wartościowa

- wiadomość *pożądana*; *niepożądana*; *nierozpoznana*



# Poczta elektroniczna

## Ochrona anty-spamowa

### Poziom MTA

- zaleta: oszczędność zasobów – odrzucamy spam na pierwszej linii obrony
- wada: mało informacji do dyspozycji – duże prawdopodobieństwo pomyłki
- analiza nagłówka SMTP, np.
  - adresów: czy są weryfikowalne w DNS, czy odpowiadają rekordom MX (→ SPF), czy nie są na czarnej liście (zwykle brak białych list)
  - weryfikacja konta nadawcy (komenda VRFY protokołu SMTP)
  - szare listy (*greylisting*) – duża skuteczność, małe efekty uboczne (opóźnienie)

# Poczta elektroniczna

## Ochrona anty-spamowa

### Poziom MDA

- analiza heurystyczna (na podstawie przygotowanej bazy danych charakterystycznych)
- analiza statystyczna (samouczące się filtry Bayesa)
- *challenge-response* (kontrowersyjne i rzadko stosowane)
  - nieskuteczny wobec scamu

# Poczta elektroniczna

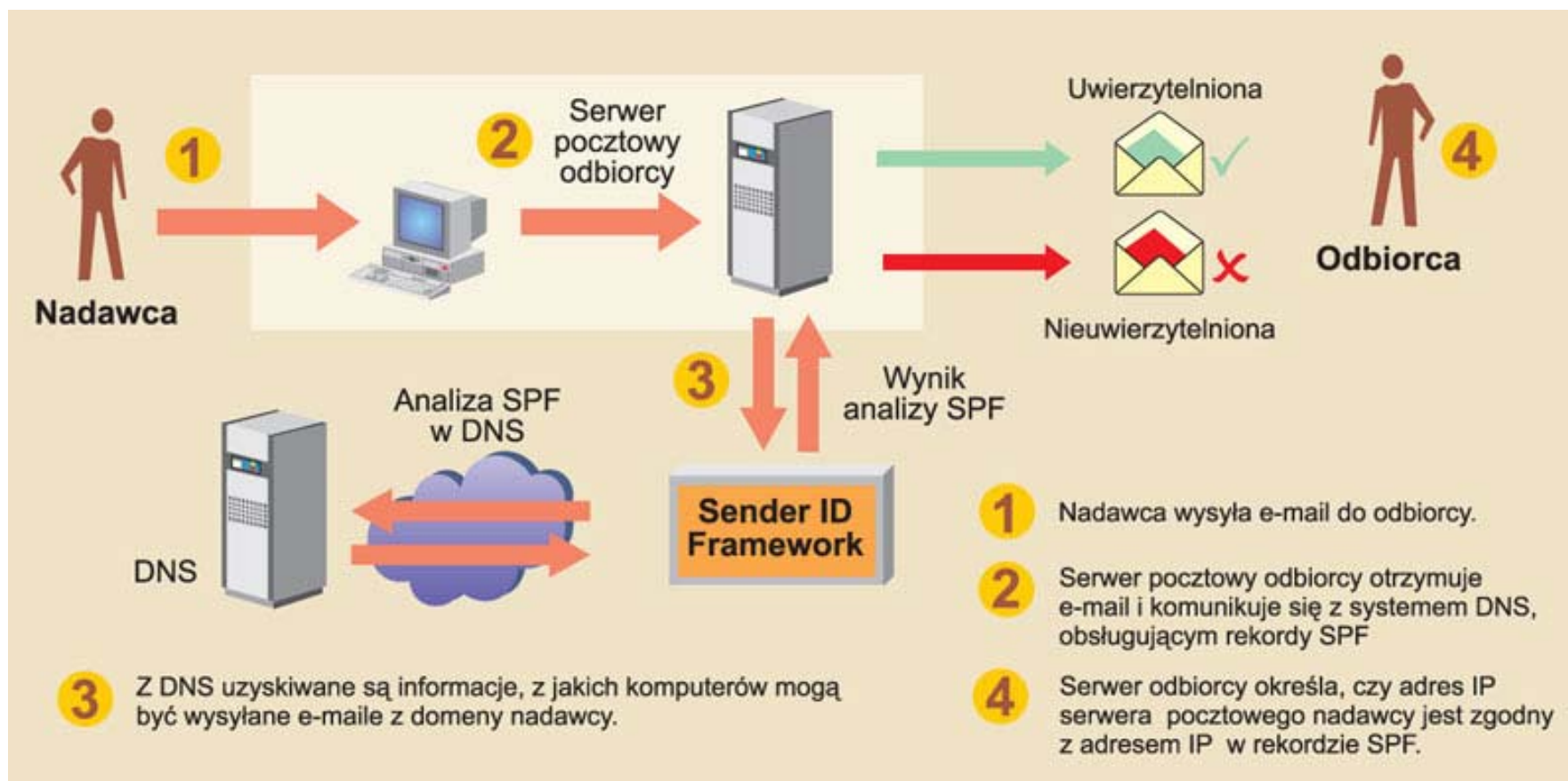
## Uwierzytelnianie przesyłek

- standardy SenderID, DKIM (*Domain-Keys Identified Mail*), DMARC
- uwierzytelnione wiadomości mogą omijać filtry (unikając problemu błędnej klasyfikacji)
- ale ochrona nie zawsze jest skuteczna

# Poczta elektroniczna

## Uwierzytelnianie przesyłek

### SenderID (RFC 4408) + Sender Policy Framework (SPF)



# Poczta elektroniczna

## Uwierzytelnianie przesyłek

### Przykładowy rekord SPF dla microsoft.com

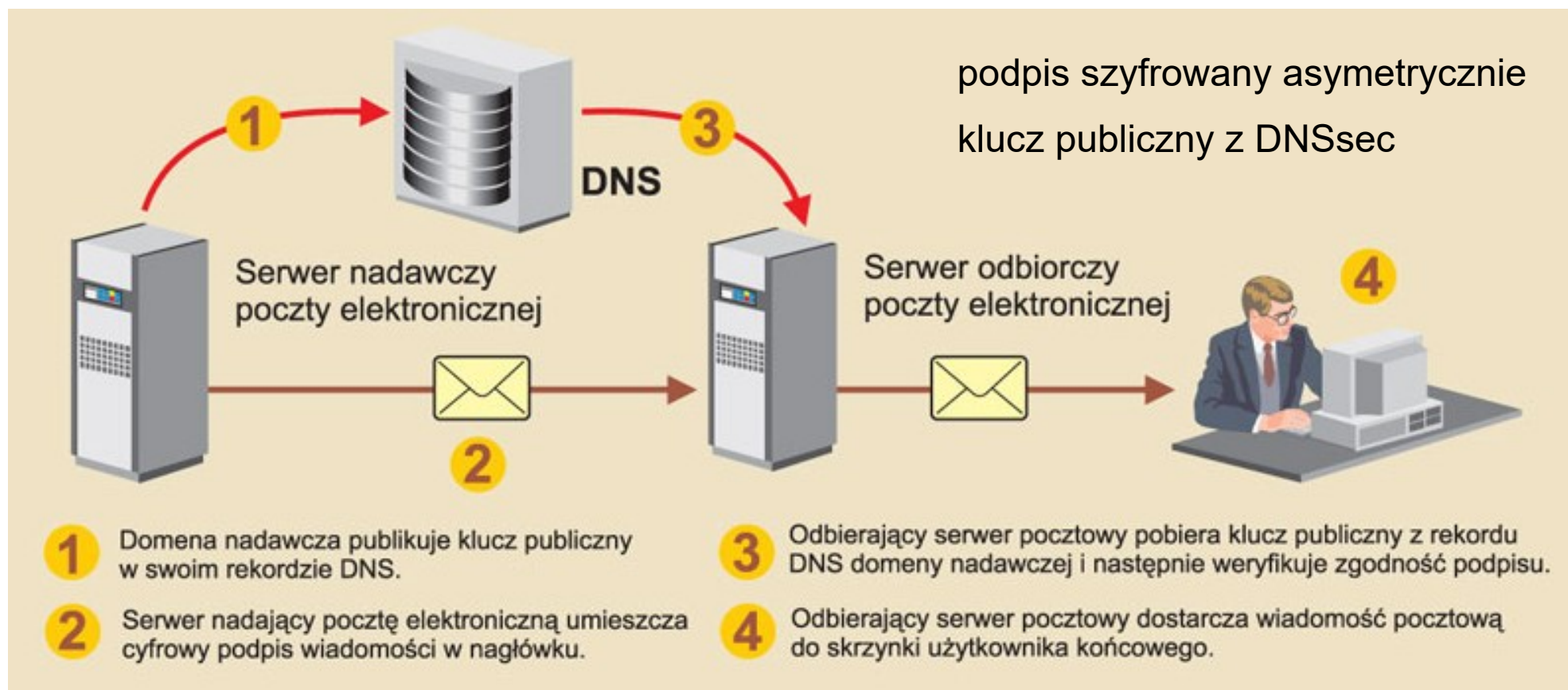
```
spf1 include:_spf-a.microsoft.com include:_spf-b.microsoft.com include:_spf-c.microsoft.com  
include:_spf-ssg-a.microsoft.com ip4:131.107.115.215 ip4:131.107.115.214 ip4:205.248.106.64  
ip4:205.248.106.30 ip4:205.248.106.32 ~all
```

- SPF v.1
- include = odpytaj dodatkowe DNS
- ~all = SOFTFAIL all other IP  
(+ = PASS, ? = NEUTRAL, - = FAIL, ~ = SOFTFAIL)

# Poczta elektroniczna

## Uwierzytelnianie przesyłek

### DKIM (*Domain Keys Identified Mail*)



# Poczta elektroniczna

## Uwierzytelnianie przesyłek

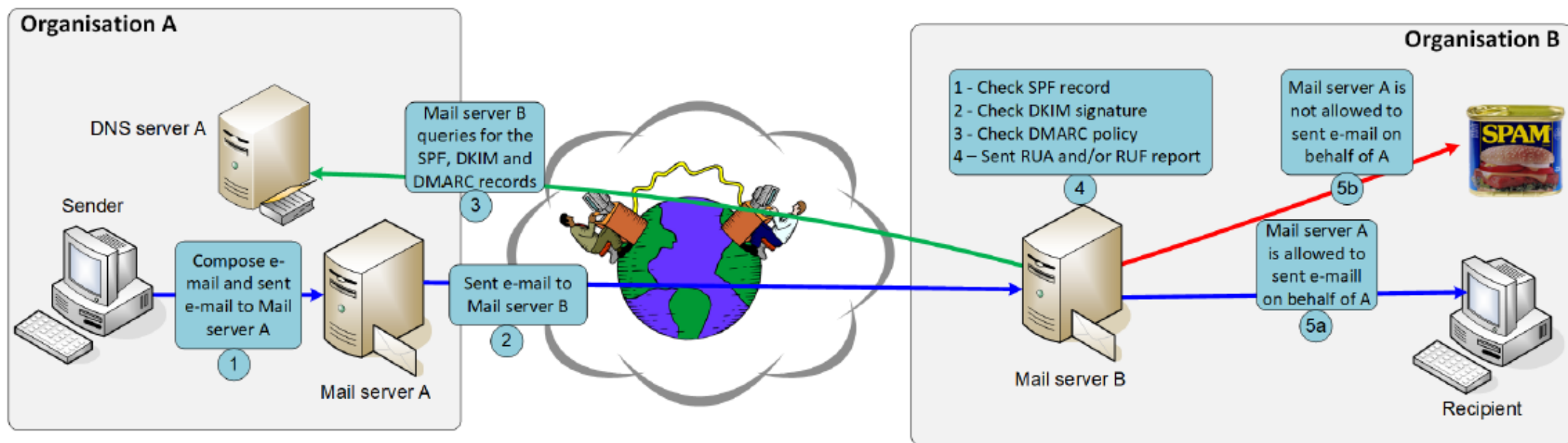
### DKIM TXT resource record:

```
201707._domainkey.belastingdienst.nl IN TXT "v=DKIM1;  
p=MIIBIjANBggqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAYzXWCOzeB5qswe  
y69WrHNeqdgNNUiFJkT/EMjm78h1zMXkrd6t0VtTB4rAe39/BlwNFC0jKskE  
3u1n16whfQX3fT/68xr2SdcOp6j/DTtS6rC1EWFXYawX6NfxM/Pt8DV5CLDF  
GHMht63LetGyiQYv+TrBBiATPjfLPgrArx7jaAoPv0Az/ec86rl+Q9jXA0QO  
7zR6Ih""0TIJYwnzVf/7Dsl4GpsmZsN1oEaXhauuDuynQsHm9iptzKC8IKHa  
Gr9g8qPnh8PDAm0QJSWAq5j1h12j7qjMLwOMEwPKwCE9HnWzeUpzxaJDHL2K  
4dHYkXF6ErRjLhtTU2Mx6/F+7Ku4wQIDAQAB;"
```

# Poczta elektroniczna

## Uwierzytelnianie przesyłek

DMARC (*Domain-based Message Authentication, Reporting and Conformance*)





# Poczta elektroniczna

## Uwierzytelnianie przesyłek

### DMARC TXT resource record:

```
_dmarc.belastingdienst.nl IN TXT "v=DMARC1; p=reject;  
    rua=mailto:dmarc.rua@belastingdienst.nl; sp=reject;"
```

rua: Reporting URI(s) for aggregate data

ruf: Reporting URI(s) for failure data

- raporty w formacie XML są dostarczane przez SMTP

# Poczta elektroniczna

## Standardy ochrony dla SMTP

- PGP – Pretty Good Privacy
- S/MIME – *Secure* MIME  
(v. 3.2 RFC 5750-51, 2010 r.)



<https://blog.hboeck.de/archives/893-efail-Outdated-Crypto-Standards-are-to-blame.html>

<https://latacora.micro.blog/2019/07/16/the-pgp-problem.html>

## Protokoły IM wykorzystujące kryptografię

- OTR – Off-the-Record Messaging
- Signal Protocol – np. Signal, WhatsApp i in.