Bezpieczny komunikator w sieci heterogenicznej

Autor:

Piotr Ignał

Opiekun:

prof. dr hab. inż. Zbigniew Kotulski

Plan prezentacji

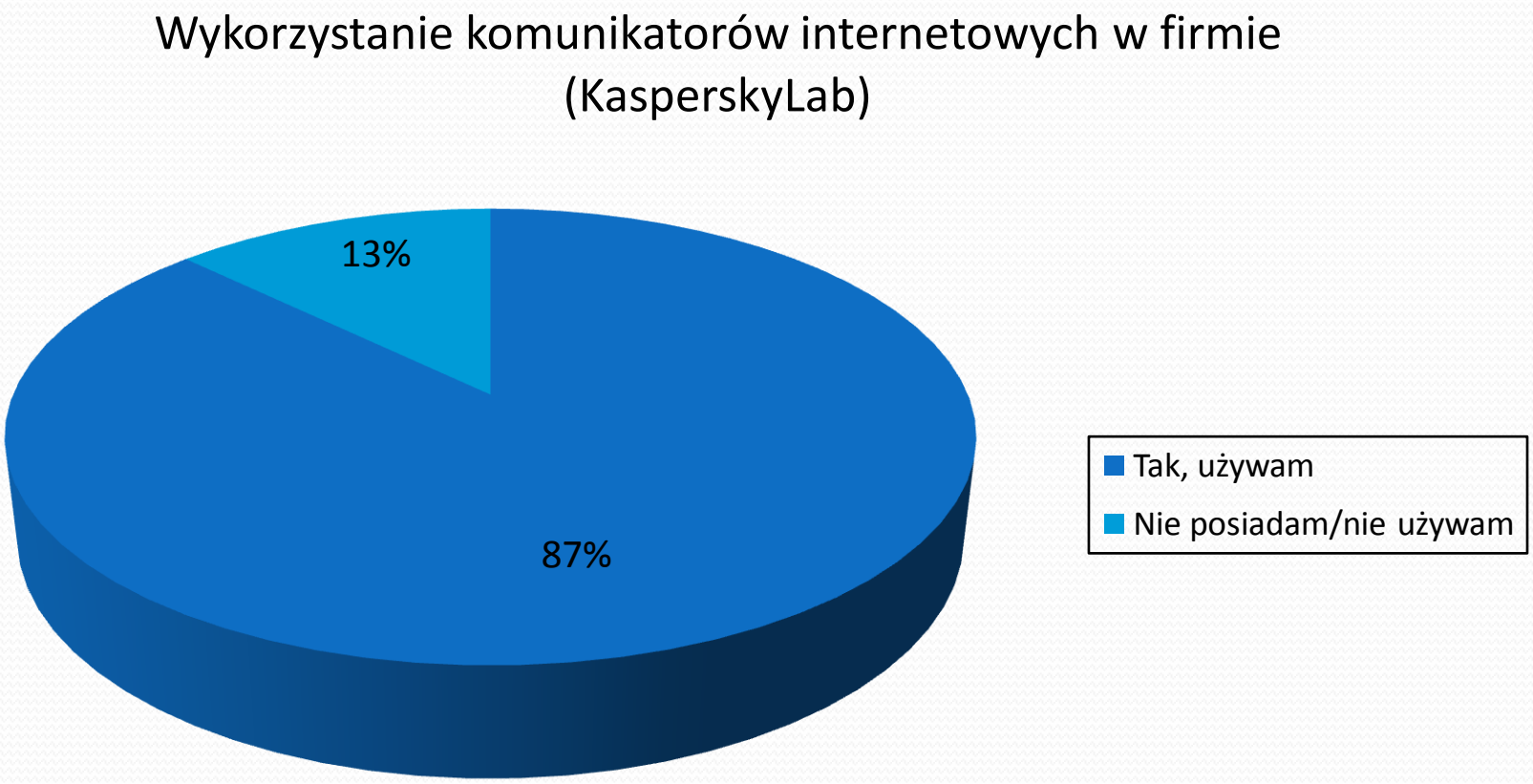
1. Wprowadzenie – przedstawienie zagrożeń
2. Cel pracy
3. Opis istniejących rozwiązań
4. Opis użytych algorytmów kryptograficznych
5. Architektura wraz z użytym protokołem
6. Analiza odporności na ataki
7. Postęp pracy
8. Bibliografia

Zarys problemu

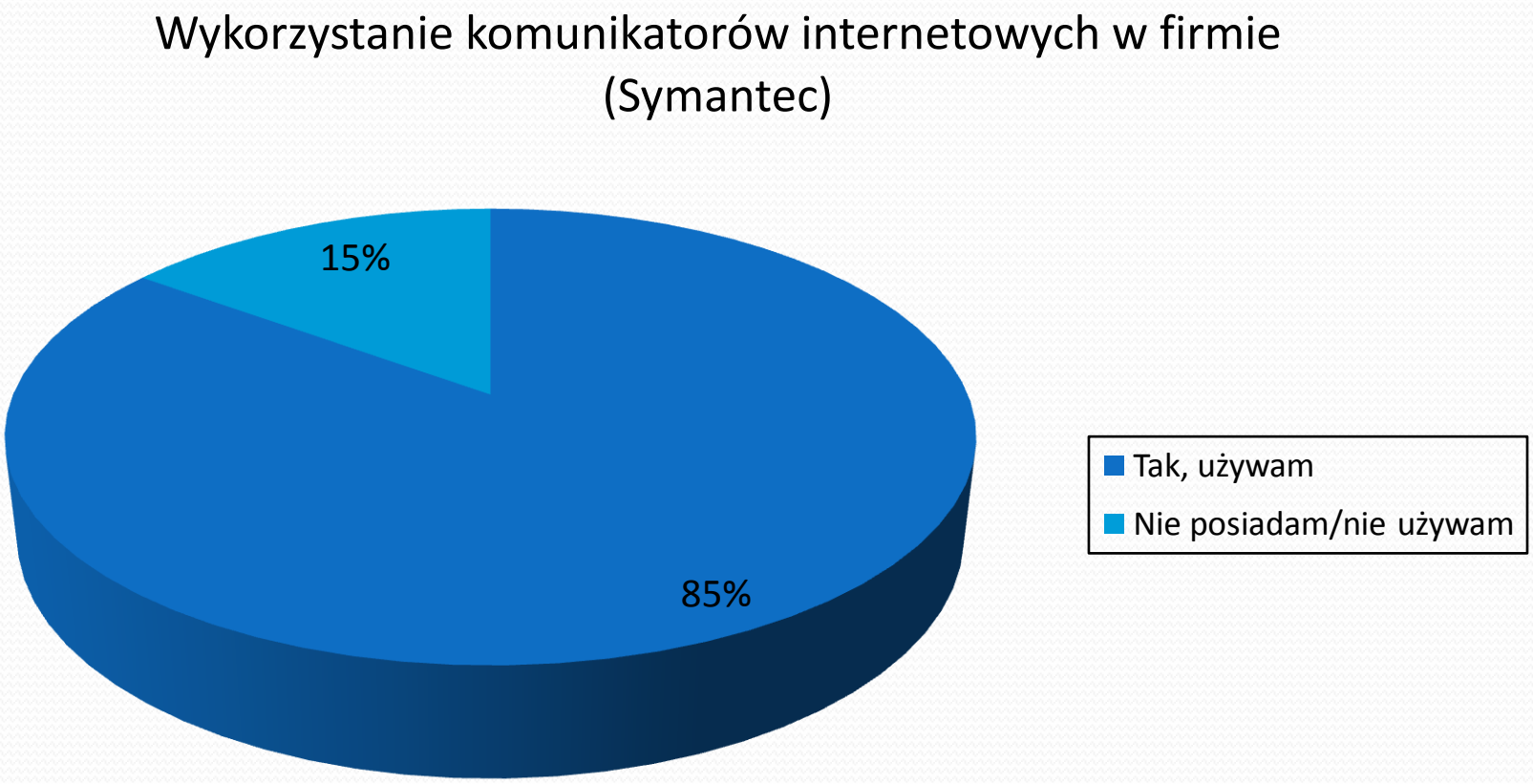
Komunikatory internetowe są przydatnym narzędziem biznesowym z kilku powodów:

- Znacznie mniej formalne od wiadomości e-mail
- Szybkie
- Wygodne
- Zapewniają „świadomość dostępności”

Wyniki przeprowadzonych badań 1/6

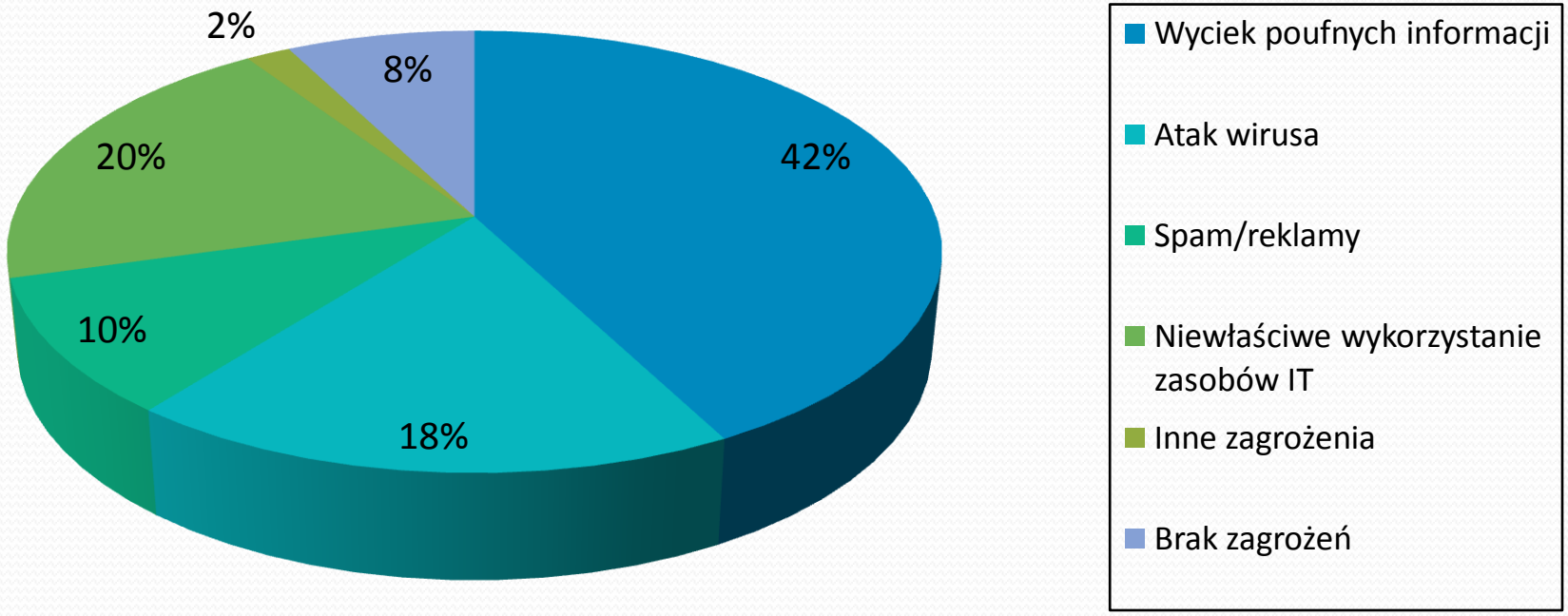


Wyniki przeprowadzonych badań 2/6

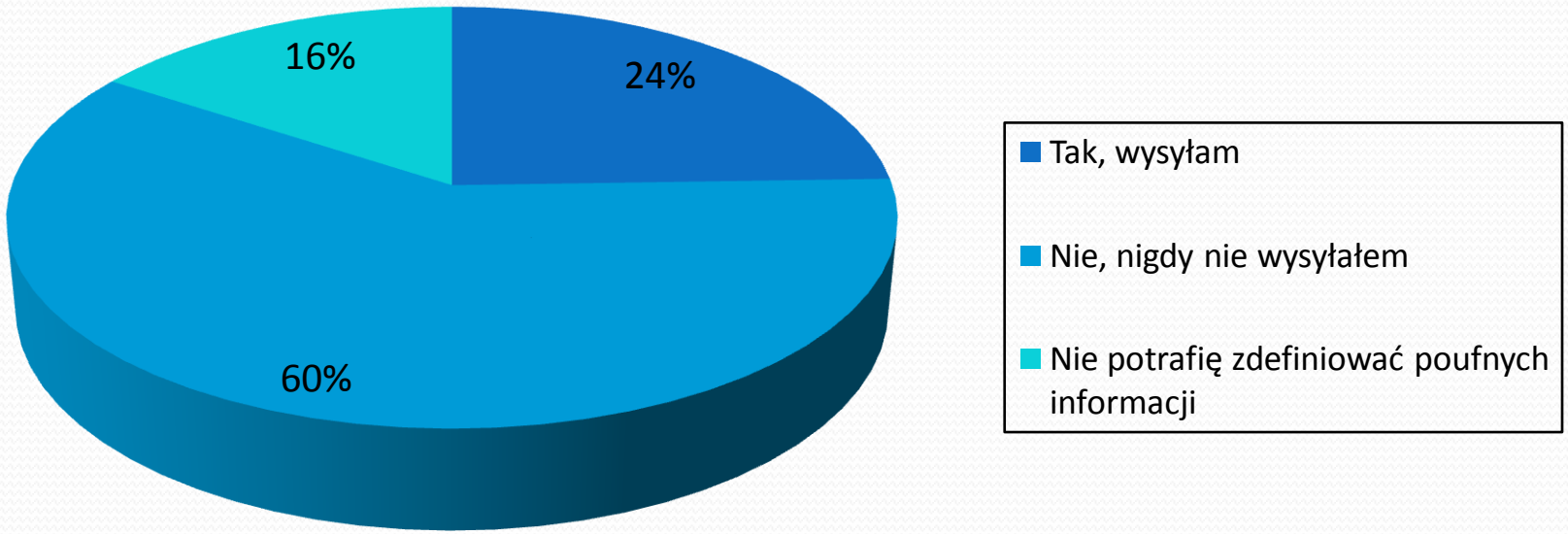


Wyniki przeprowadzonych badań 3/6

Zagrożenia związane z korzystaniem z komunikatorów internetowych (KasperskyLab)

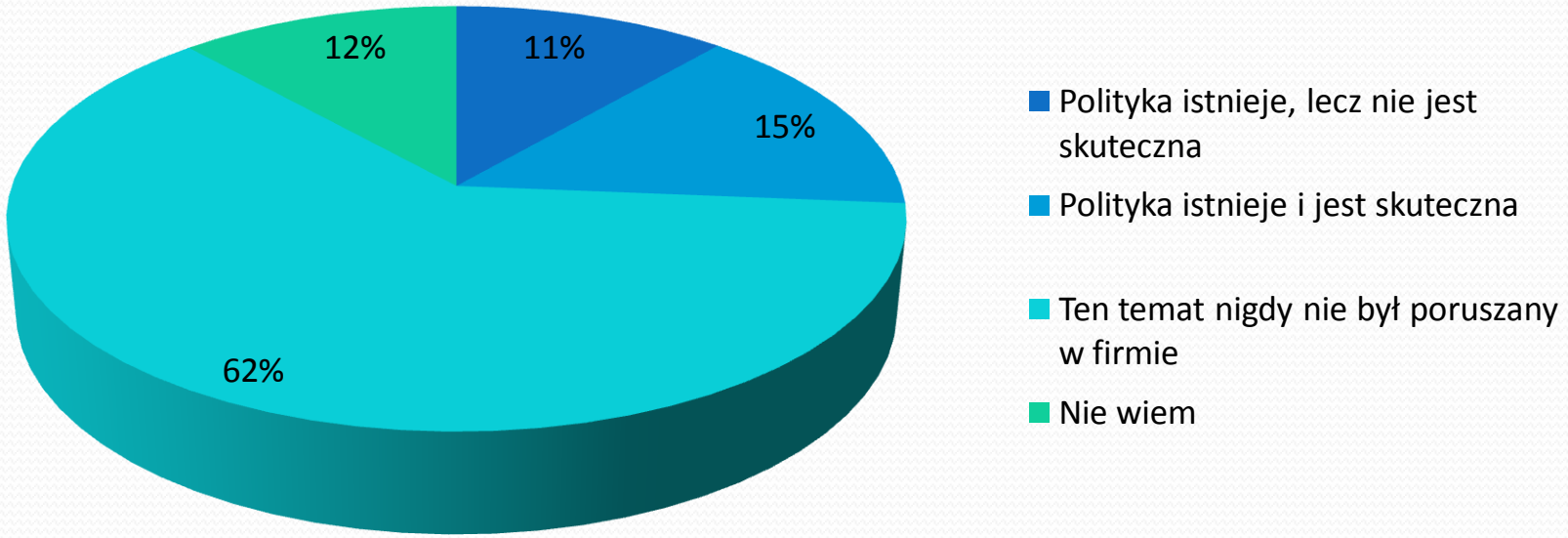


Wyniki przeprowadzonych badań 4/6

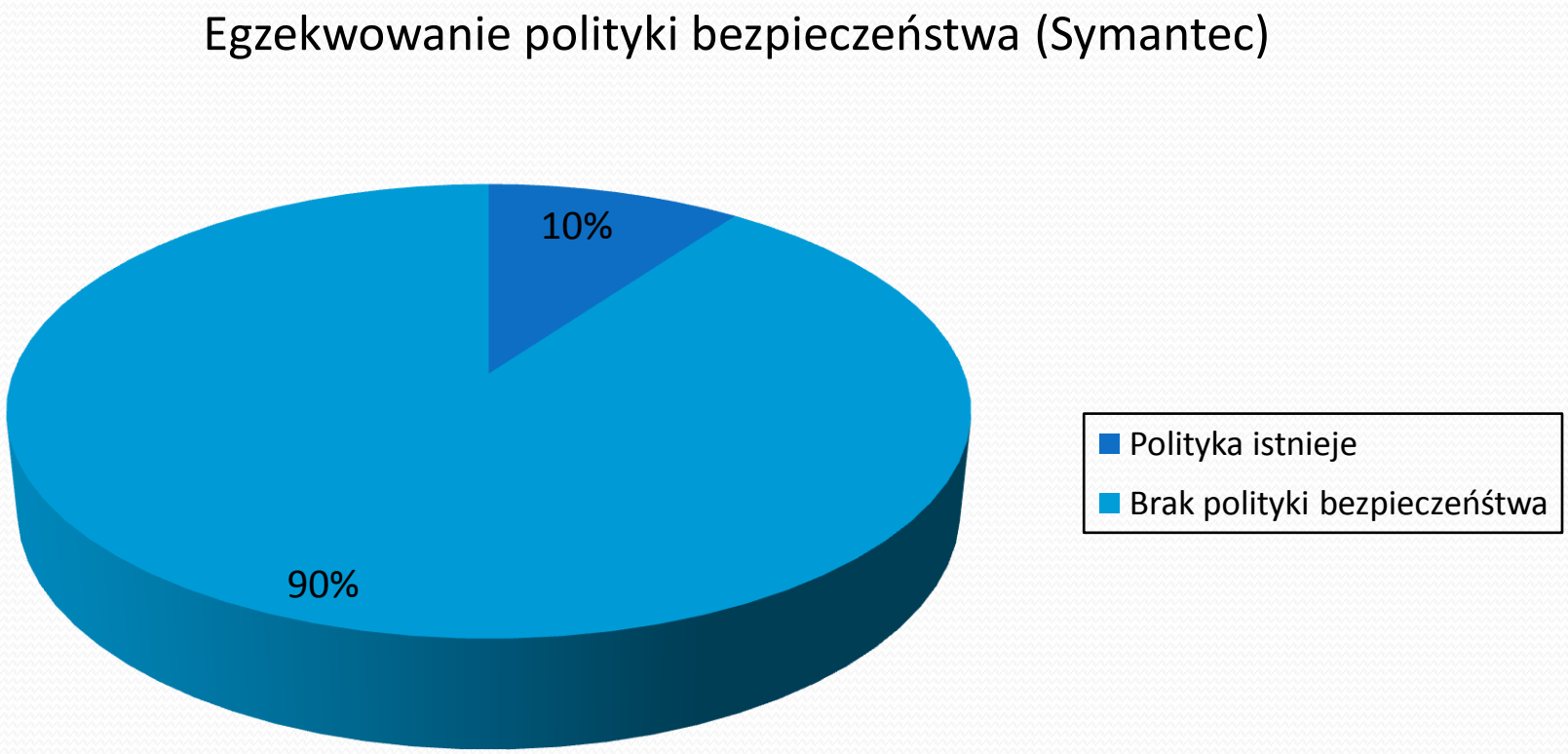


Wyniki przeprowadzonych badań 5/6

Egzekwowanie polityki bezpieczeństwa (KasperskyLab)



Wyniki przeprowadzonych badań 6/6



Cel pracy:

Implementacja komunikatora który:

- Umożliwi uwierzytelnianie, autoryzacje i rozliczalność użytkowników
- Zapewni bezpieczną transmisję end to end w sieci Internet
- Dostosowany będzie zarówno do komputerów osobistych jak i urządzeń mobilnych

Realizacja:

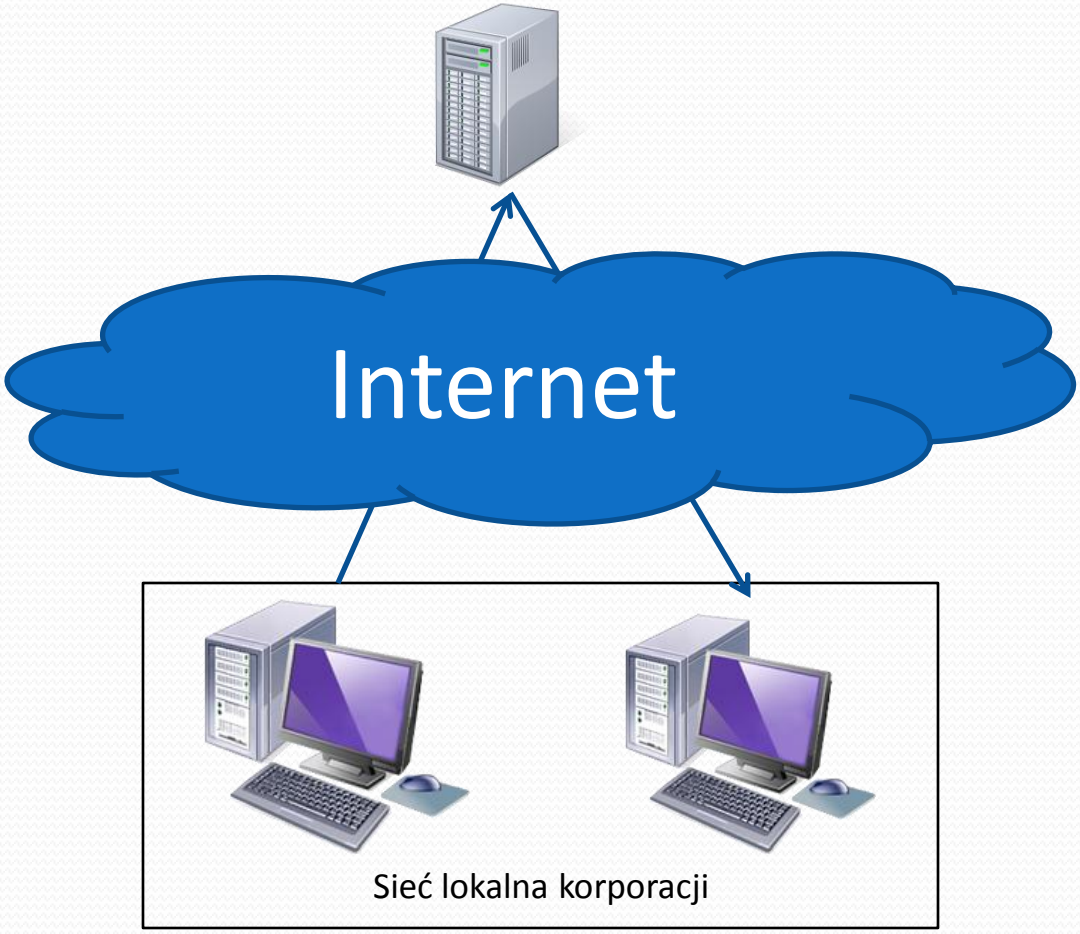
J2ME – aplikacja klienta dla urządzeń mobilnych

- LWUIT – interfejs użytkownika
- lcrypto-j2me-145 – usługi kryptograficzne

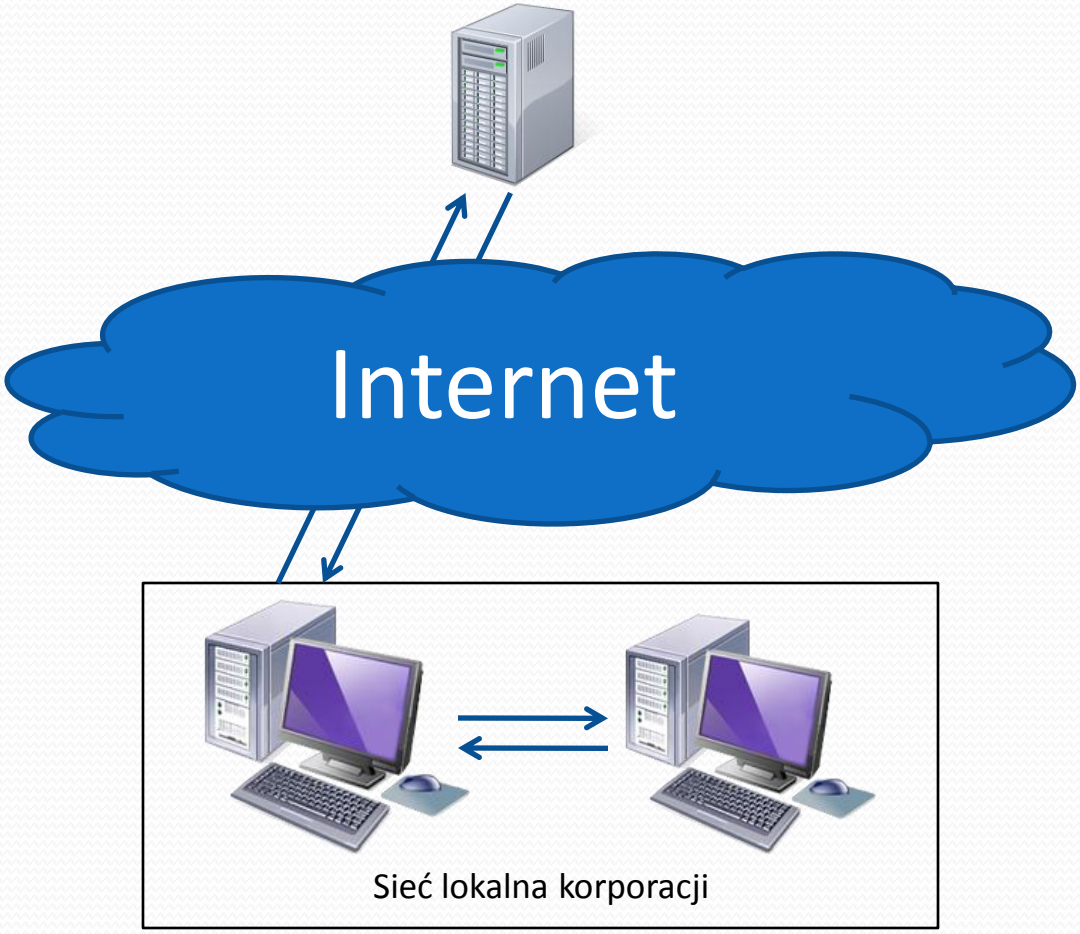
J2SE – aplikacja serwera (baza danych MySQL) oraz aplikacja klienta

- bcprov-jdk16-145 – usługi kryptograficzne

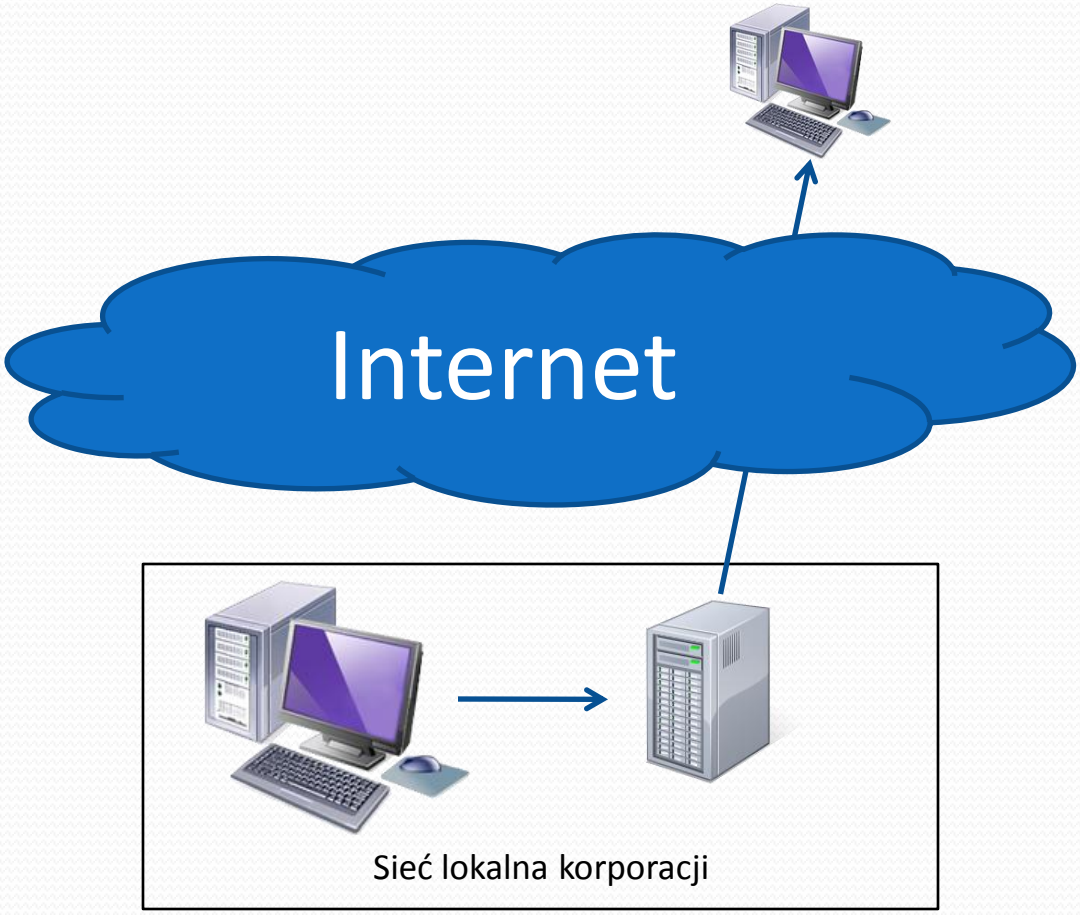
Istniejące rozwiązania 1/3



Istniejące rozwiązania 2/3



Istniejące rozwiązania 3/3



SSL 3.0 / TLS

- ✓ Standard, łatwy w implementacji (biblioteki niemal w każdym języku programowania)
- ✓ Zapewnia poufność, integralność, uwierzytelnianie
- ✘ Bezpieczne połączenie jedynie na linii klient-serwer
- ✘ Możliwość wynegocjowania odmiennych parametrów kryptograficznych (jedna strona bardziej podatna na atak)
- ✘ Często możliwe przejście na SSL 2.0 (mnogość ataków)

Rozwiązanie:

1. Połączenie SSL end to end, pomiędzy każdym klientem:

- ✘ Mało wydajne w przypadku urządzeń mobilnym
- ✘ Problem z rozmowami konferencyjnymi
- ✘ Nie zapewnia rozliczalności

2. Nowy protokół dostosowany do urządzeń mobilnych w oparciu o architekturę klient-serwer

- ✘ ?

Algorytmy kryptograficzne 1/3

1. AES (Rijndael) – w październiku 2000 roku uznany za standard.

✓ Szybki

✓ Odporny na wszystkie znane metody kryptoanalizy

✓ Tryb pracy: *Cipher Block Chaining*

2. HMAC (SHA-2)

✓ SHA-1 wraz z końcem 2010 r. zostanie wycofane

✓ Integralność + uwierzytelnianie danych

Algorytmy kryptograficzne 2/3

3. RSA:

- ✓ Siła algorytmu opiera się na problemie faktoryzacji dużych liczb
- ✓ Klucze 1024 bity (klienci), 2048 bitów (serwer)
- ✓ Zapewnia uwierzytelnianie wiadomości

4. Podpis cyfrowy w oparciu o RSA i SHA256:

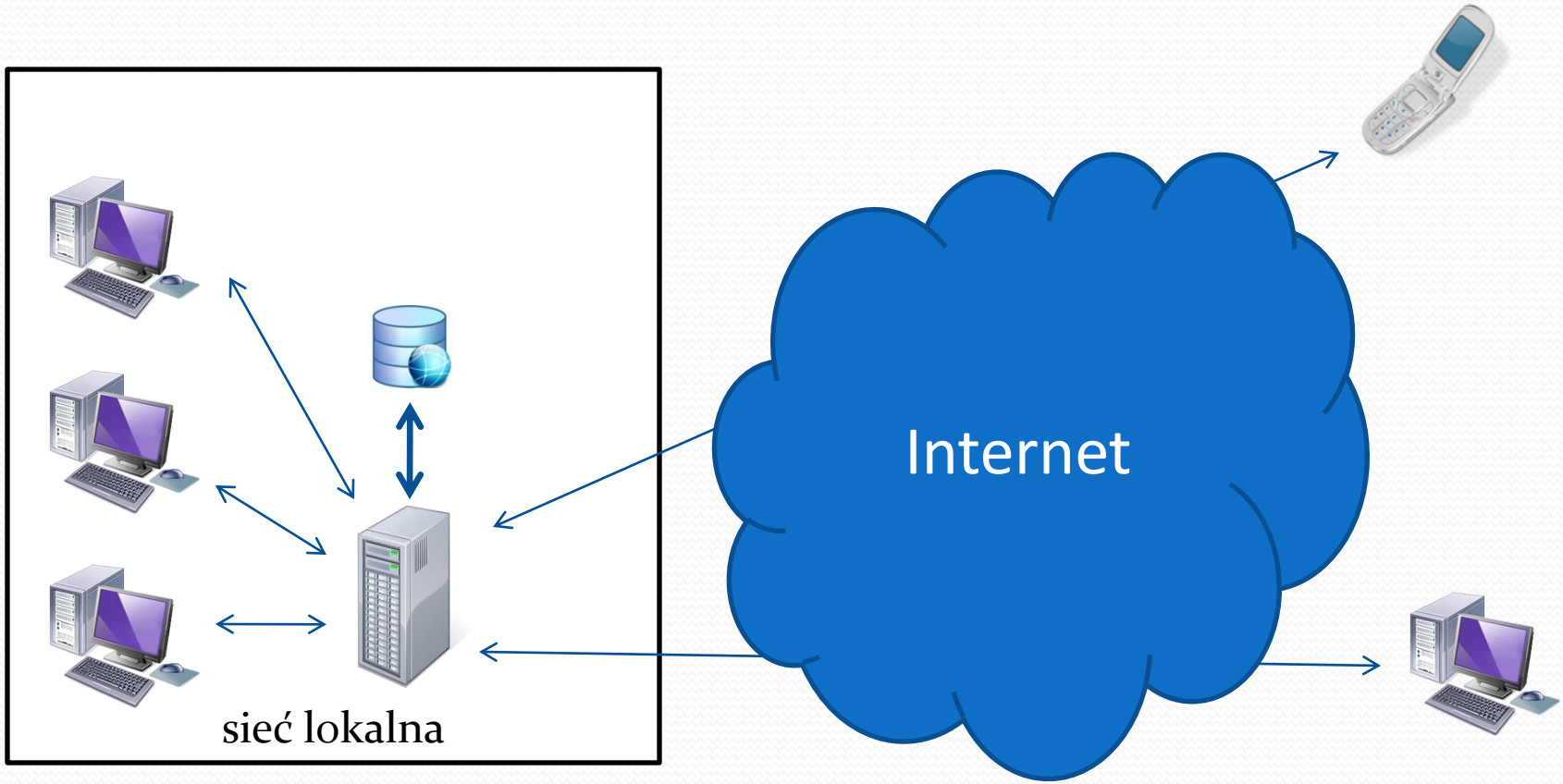
- ✓ Uwierzytelnianie
- ✓ Integralność
- ✓ Niezaprzeczalność

Algorytmy kryptograficzne 3/3

5. Diffie Hellman:

- ✓ Siła opiera się na trudności obliczenia logarytmu dyskretnego z dużych liczb
- ✓ Każda ze stron ma taki sam udział w tworzeniu klucza
- ✓ Możliwość uzgodnienia klucza dla wielu stron (kluczowe podczas rozmów konferencyjnych)

Architektura



Uwierzytelnianie 1/2

Znacznik czasowy [8 bajtów]: uniemożliwia atak powtórzeniowy

Numer [4 bajty]: identyfikator użytkownika

SHA1(hasło) [20 bajtów]: hasło uwierzytelniające użytkownika

AES256[32 bajty]: klucz sesyjny

Podpis [128 bajtów]: zapewnia niezaprzeczalność i integralność użytkownika

Szyfrowanie kluczem publicznym serwera RSA 2048

Znacznik czasowy	Numer	SHA1(hasło)	AES 256 klucz sesyjny
Podpis RSA 1024			

Uwierzytelnianie 2/2

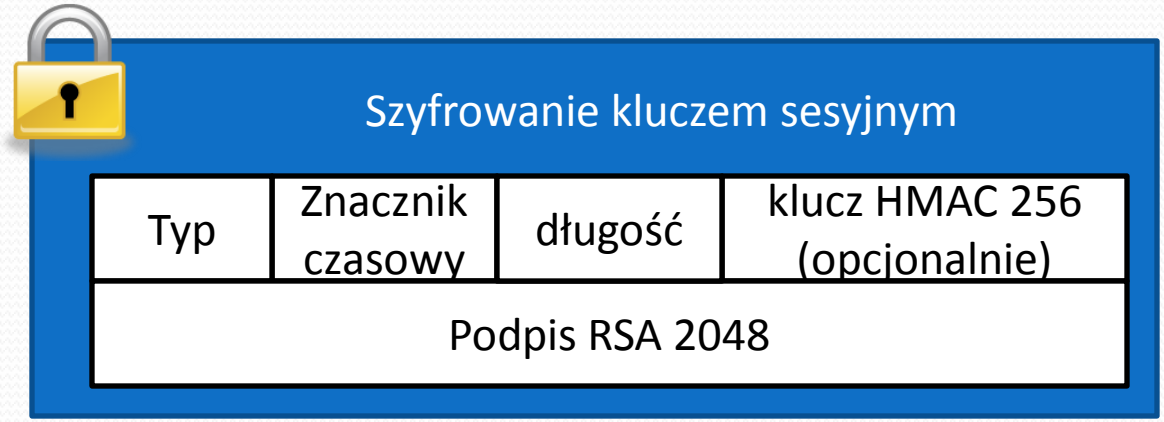
Typ [1 bajt]: określa typ odpowiedzi

Znacznik czasowy [8 bajtów]: uniemożliwia atak powtórzeniowy

HMAC [32 bajty]: opcjonalny klucz zapewniający integralność

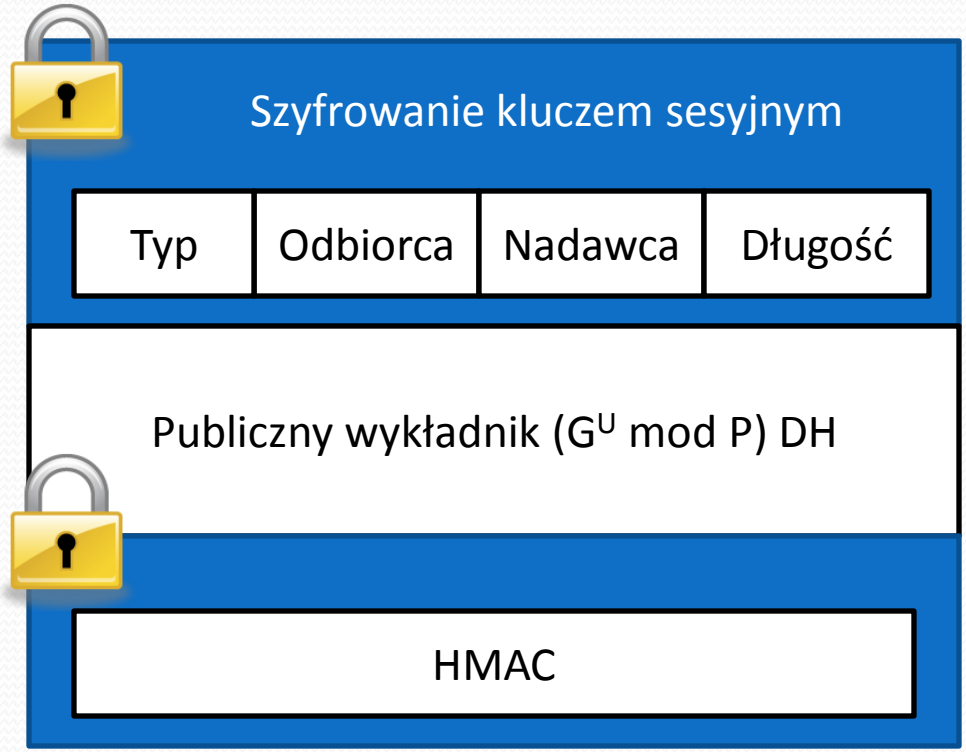
Podpis [256 bajów]: zapewnia niezaprzeczalność i integralność

Długość [4 bajty]: długość nagłówka w bajtach



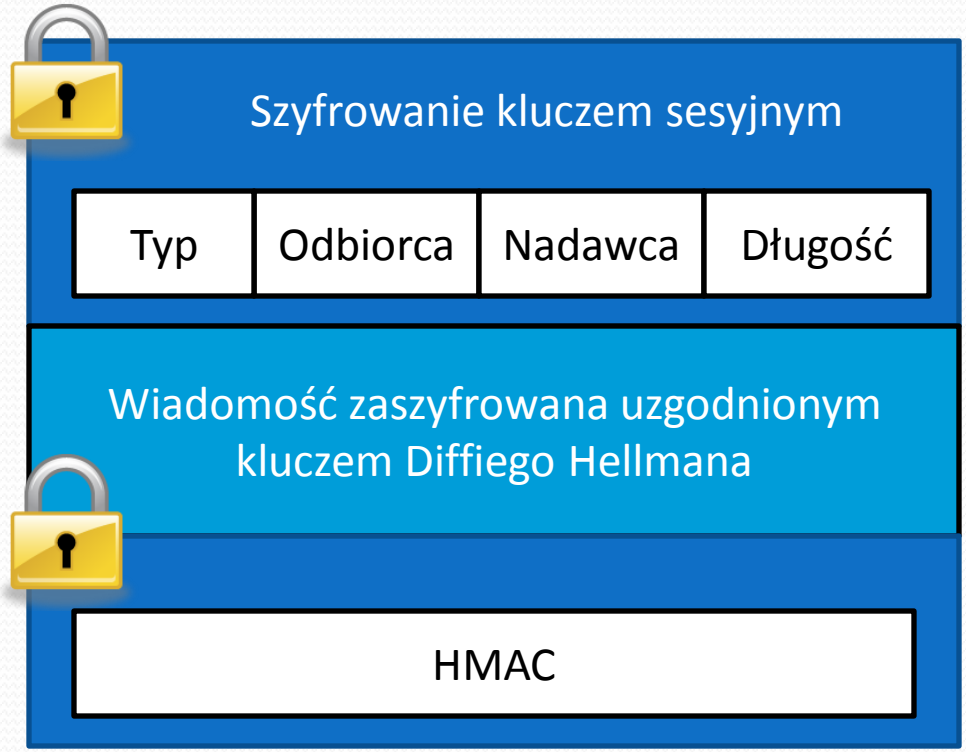
Uzgodnienie kluczy end to end

Odbiorca[4 bajty]: numer odbiorcy
Nadawca[4 bajty]: numer nadawcy
Długość [4 bajty]: długość wiadomości wyrażona w bajtach
HMAC [20 bajtów]: MAC z wymieszanym kluczem



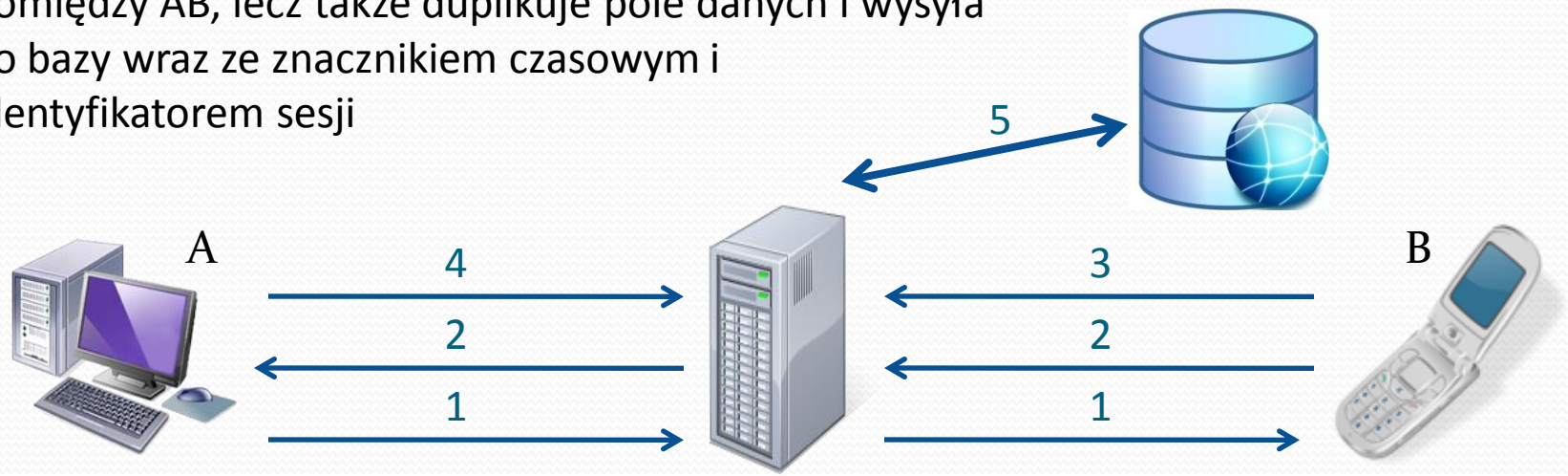
Komunikacja

Treść wiadomości: Szyfrowana jest AESem z kluczem 256 bitowym



Rozliczalność

1. A rozpoczyna wymianę kluczy DH (wysyła G^v mod P)
2. B odsyła wygenerowaną wartość G^u mod P, oraz oblicza $klucz_{AB}$ (G^{uv} mod P)
3. B szyfruje $klucz_{AB}$ swoim kluczem publicznym i wysyła na serwer
4. A robi dokładnie to samo
5. Serwer nie tylko przekazuje dalej wiadomości pomiędzy AB, lecz także duplikuje pole danych i wysyła do bazy wraz ze znacznikiem czasowym i identyfikatorem sesji



Analiza odporności na zagrożenia 1/3

Zagrożenia podczas uwierzytelniania:

1. Podszycie – zagrożenie te zostało wyeliminowane, poprzez wprowadzenie podpisu cyfrowego
2. Modyfikacja danych – również przed takim atakiem, zabezpiecza podpis cyfrowy
3. Podsłuch – wykorzystany szyfr RSA jest obliczeniowo bezpieczny, zatem podsłuch jest nieopłacalny (finansowo i czasowo)
4. Atak powtórzeniowy – znacznik czasowy uniemożliwia ponowne wykorzystanie danego pakietu

Zagrożenia po uwierzytelnieniu:

1. Podsłuch nagłówka – teoretycznie możliwy atak z przewidywanym tekstem jawnym (atak na nagłówek), jednak siła algorytmu praktycznie uniemożliwia złamanie szyfru w trakcie trwania sesji (Wektor Inicjalizujący).
2. Podsłuch pola danych – wykorzystanie trybu CBC, a także wektora inicjalizującego IV działającego w trybie licznika, zapewnia wystarczający poziom bezpieczeństwa.

Analiza odporności na zagrożenia 2/3

3. Analiza ruchu – Pola nagłówków TCP i IP są przesyłane w formie jawnej. Połączenie przez serwer utrudnia śledzenie pakietów, jednak go nie uniemożliwia. Atak taki niesie jednak za sobą małe zagrożenie (nieporównywalnie mniejsze od zagrożenia w komunikatorach używających w sposób opcjonalny SSL)
4. Atak powtórzeniowy – Wektor inicjalizujący nagłówka praktycznie uniemożliwia taki atak
5. Podszycie – konieczność znajomości kluczy sesyjnych, a więc złamania tym samym algorytmu uwierzytelniania
6. Modyfikacja danych – wiązałaby się ze złamaniem 256 bitowego klucza używanego w funkcji HMAC. Obliczeniowo nieopłacalne.

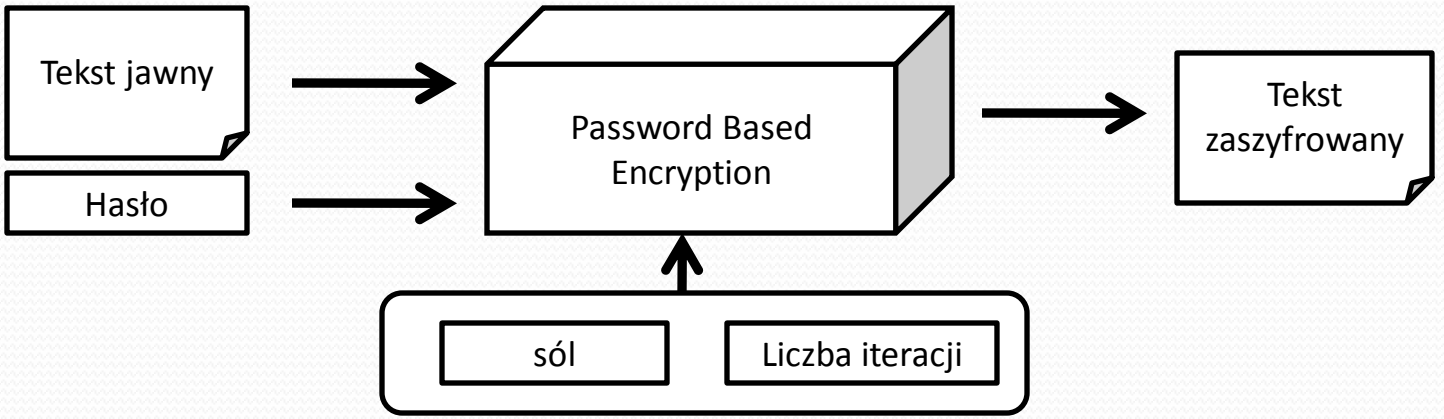
Zagrożenia aplikacji serwera:

1. Włamanie na serwer lub dostęp nieuprawnionego pracownika nie umożliwi wycieku informacji z firmy. Zapewnione jest to poprzez szyfrowanie end to end.
2. Wirusy - ?

Analiza odporności na zagrożenia 3/3

Zagrożenia aplikacji klienta:

1. Włamanie do aplikacji/podgląd klucza prywatnego –wygenerowanego klucza w oparciu o mechanizm PBE



2. Uwierzytelnianie – 3 krotne podanie błędnego hasła powoduje zablokowanie konta.
3. Ingerencja w kod – aplikacja zabezpieczona podpisem cyfrowym.

Bibliografia

1. D.Hook, *Kryptografia w Javie. Od podstaw*, Helion 2006
2. C.Horstmann, G.Cornell, *Java 2. Techniki zaawansowane*, Helion 2005
3. K. Rychlicki-Kicior, *J2ME Praktyczne projekty*, Helion 2007
4. B. Schneier, D.Wagner, *Analysis of the SSL 3.0 protocol,* 1997
5. Zespół Bezpieczeństwa PCSS, *„Bezpieczna" E-Bankowość,* 2006
6. *Securing Instant Messaging,* Symantec Enterprise Security 2002
7. *Managing Instant Messaging for Business Advantage,* Symantec Enterprise Security 2006
8. A.Dolya, *Komunikatory internetowe: ułatwiają komunikację biznesową, ale niosą zagrożenia*, KasperskyLab 2007

Dziękuje za uwagę