Bezpieczeństwo systemów agentów mobilnych: zastosowanie protokołów kryptograficznych

Aneta Zwierko

Plan

- Agenci – wprowadzenie
- Bezpieczeństwo
  - Integralność
  - Uwierzytelnienie
  - Anonimowość
- Ochrona integralności agentów mobilnych
- Anonimowi agenci
- Uwierzytelnienie grupowe z częściową anonimowością
- Zastosowania: E-voting

Wprowadzenie

- Podstawowe pojęcia
  - Agent
  - Host – platforma agentowa
  - Manager - TTP
  - System agentowy
- Agenci
  - stacjonarni
  - mobilni

Agenci mobilni

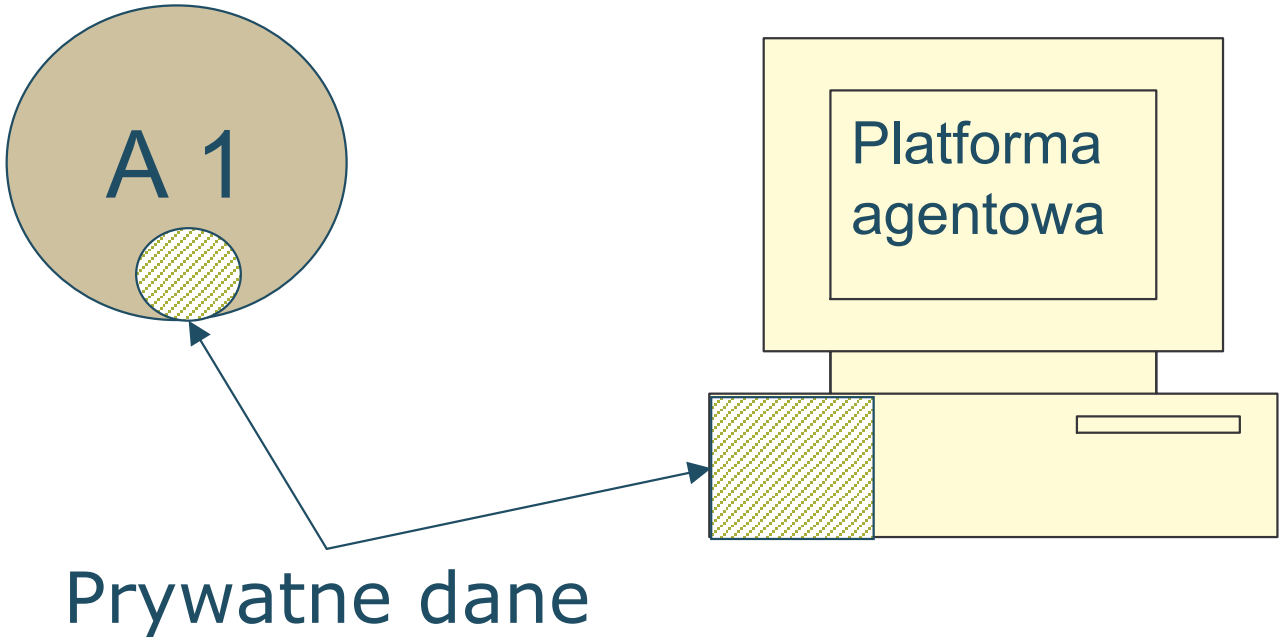
- Mobilność
  - Słaba
  - Silna
- Główne zadania
  - Zbieranie informacji
  - Komunikacja z innymi agentami
  - Analiza danych
  - Podejmowanie decyzji
- Działają niezależnie

Zastosowania

- E-/M-commerce
- Business Process Management
- Telekomunikacja
- Detekcja włamań
- Sprzęt mobilny
- Rozrywka
- Medycyna
- Szkolenia, wytwórstwo, etc.
- Przyszłość?

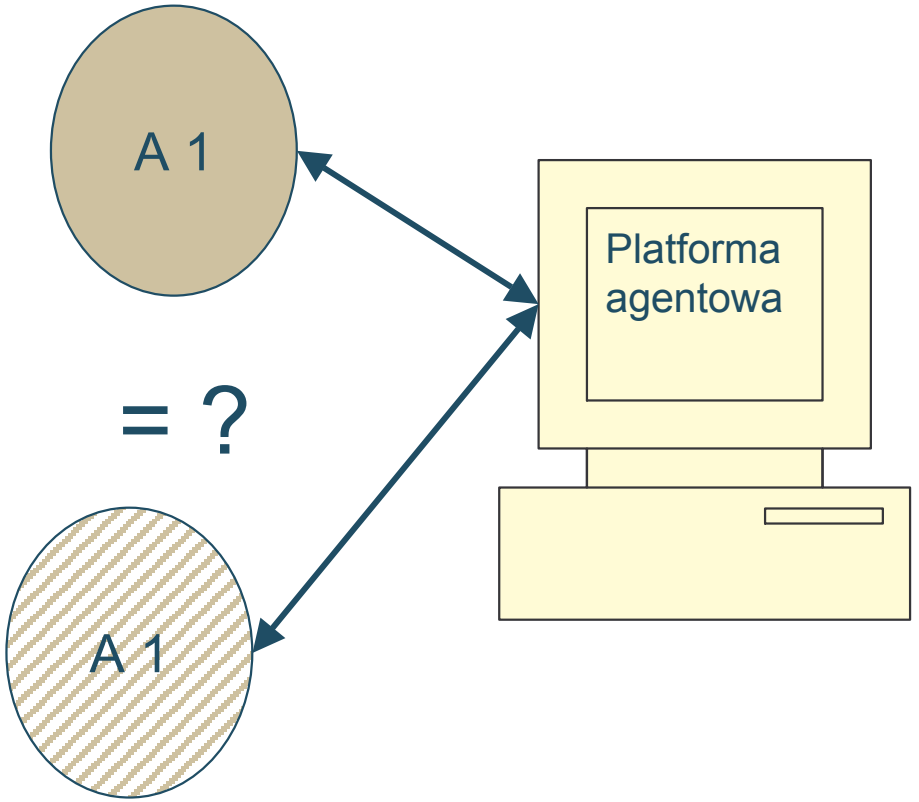
Bezpieczeństwo

- Główne wymagania
  - **Poufność**



Bezpieczeństwo

- Główne wymagania
  - **Integralność**

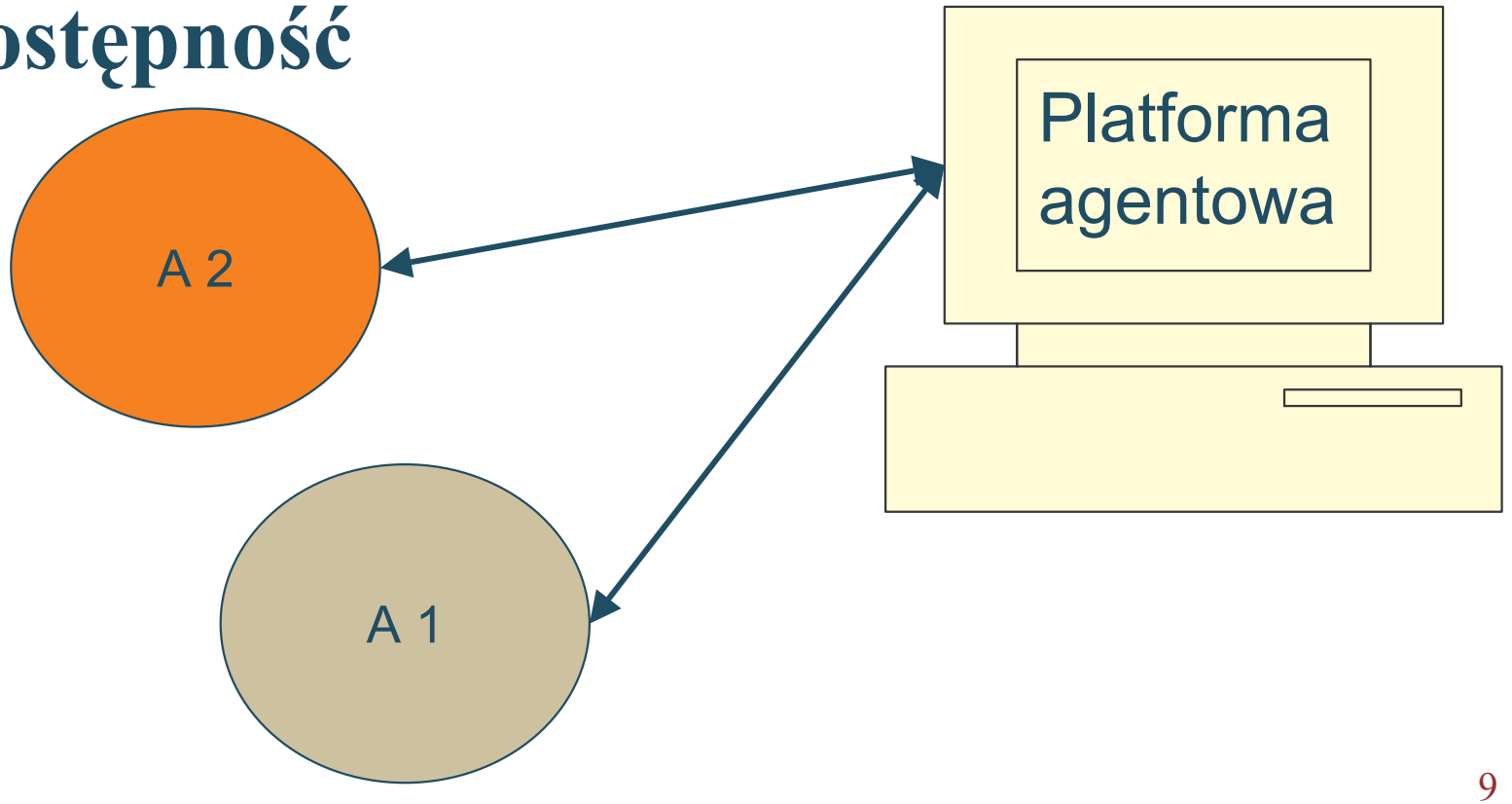


Bezpieczeństwo

- Główne wymagania
  - **Rozliczalność**

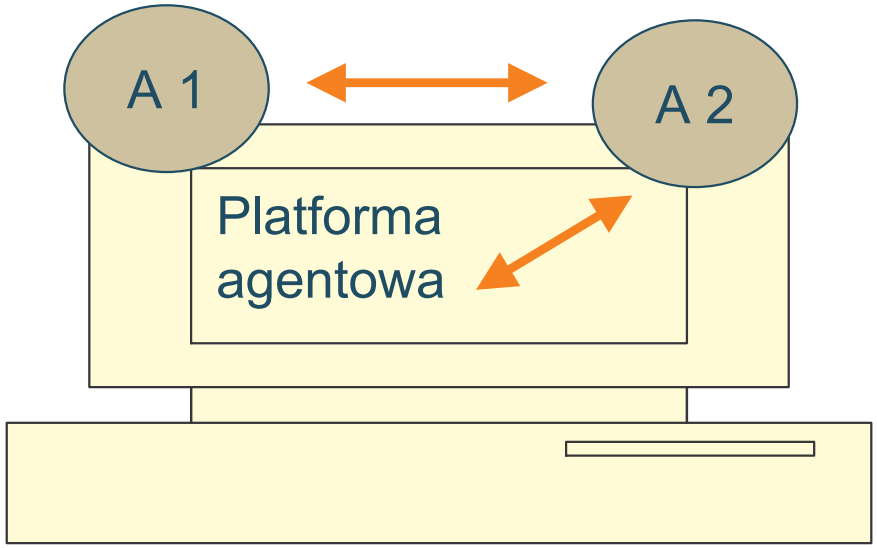
Bezpieczeństwo

- Główne wymagania
  - **Dostępność**



Bezpieczeństwo

- Główne zagrożenia
  - Agent atakujący hosta
  - Host atakujący agenta
  - Agent atakujący innego agenta
  - Inne



Anonimowość & agenci

- Anonimowość vs prywatność
- Poufność/Anonimowość
  - Agenta
  - Lokalizacji
  - Trasy
- Must-have or nice-to-have?
- Problemy
  - Rozliczalność
  - Autoryzacja ⟺ Identyfikacja
  - Praktyczne zastosowania

Ochrona integralności

Ochrona integralności

- Agent mobilny (jego kod i dane) są niezależnie przesyłane między kolejnymi platformami – duże ryzyko, że ktoś nieuprawniony zmodyfikuje kod.
- Agent nie może temu przeciwdziałać!!!
- Można jednak temu zapobiegać…
- Podstawowe podejścia
  - detekcja lub uniemożliwienie manipulacji
  - Zaufane środowisko dla „wykonania” agenta
- Rozwiązania wbudowane w agenta lub w cały system agentowy.

Podstawowe pojęcia

- Integralność agenta nie została naruszona jeśli nie można tak zmodyfikować agenta, że jego właściciel tego nie wykryje.

- Typy integralności
  - Weak forward integrity
  - Strong forward integrity
  - Publicly verifiable forward integrity
  - Black-box

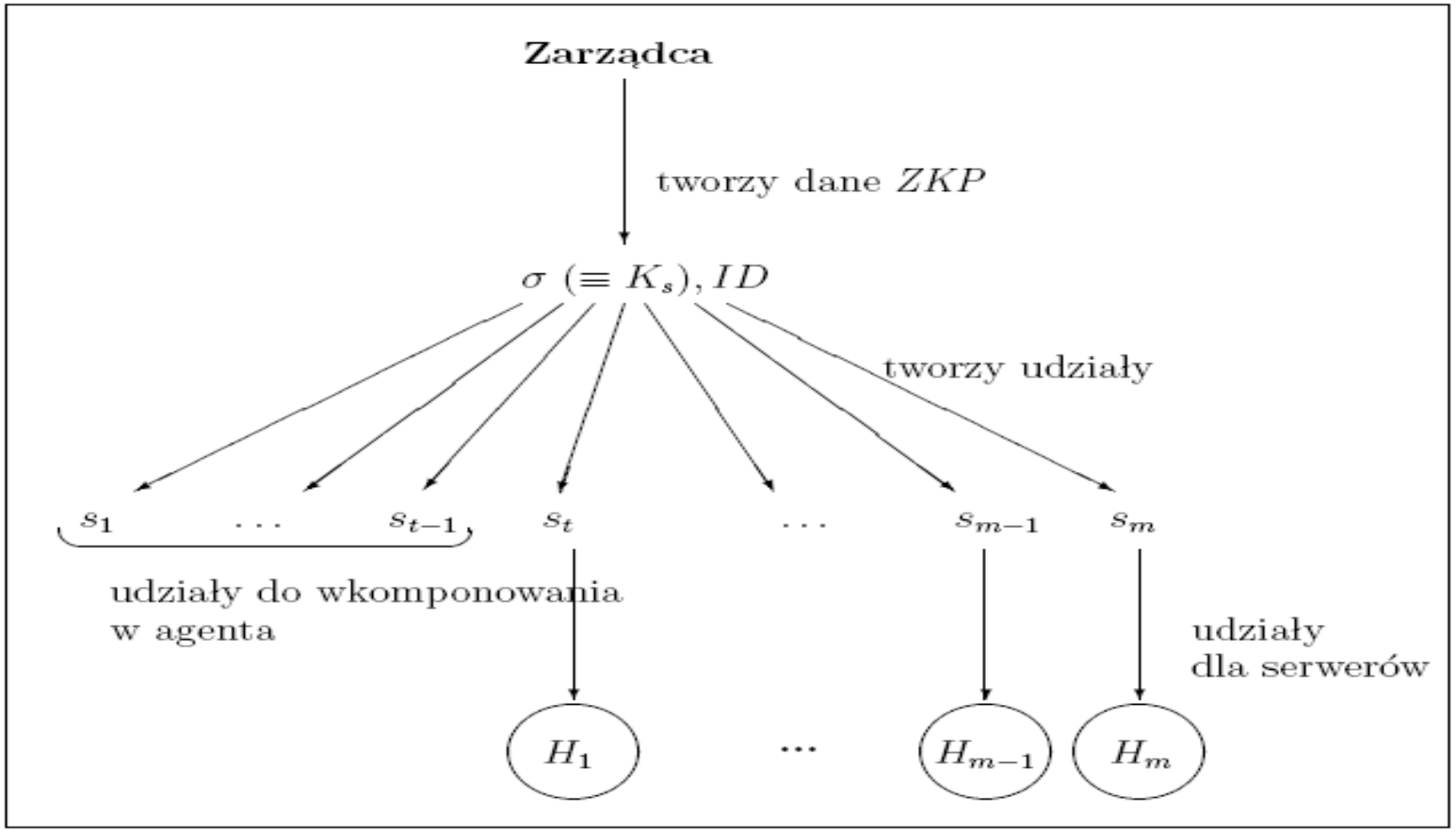
Pomysl

- Protokół wiedzy zerowej wykorzystuje się do sprawdzenia, czy udowadniający zna sekret. W naszym wypadku udowadniającym byłby agent, a sprawdzający manager/właściciel.
  - W fazie wstępnej manager generuje parę sekretów, które następnie wbudowuje w agenta.
  - Jeśli manager zażąda od agenta wykonania pewnych obliczeń (ozn. je jako funkcję *f*) to rezultatem powinien być odpowiedni sekret.
    - Ta funkcja powinna mieć następujące właściwości
      - Jeśli mamy x1 i f(x1) to znalezienie takiego x2, że f(x2) = f(x1) powinno być obliczeniowo trudne.
- Jeśli sekret jest trzymany w agencie, wtedy także host może go zweryfikować.
- Każda zmiana stanu agenta powinna powodować zmianę sekretu:
  - Nieuprawnione zmiany powodują „zniszczenie” sekretu
  - Funkcja f powinna być odporna na ataki (funkcja skrótu)
- Detekcja
- Wykorzystamy schemat identyfikacji GQ.

Faza wstępna

- Manager tworzy zestaw ID (tożsamości) dla schematu GQ i wstępne klucze.
- Dla każdego ID tworzy sekret σ
- σ jest tym sekret, który będzie wbudowany w agenta. Do wbudowania wykorzystamy system bezpiecznego podziału sekretu Asmuth, Bloom.
  - (t-1) udziałów jest wbudowanych w agenta, pozostałe są rozsyłane do hostów
- Manager musi tak wbudować udziały w agenta, aby przy poprawnym stanie, kodzie można było odzyskać prawidłowy sekret.
  - Można skorzystać z faktu, że każdy program jest FSM.
  - Do stworzenia („odzyskania z agenta”) udziałów można wykorzystać metodę opartą o szyfrowanie symetryczne lub z kluczem publicznym (z/bez interakcji z hostem)

Faza wstępna



Weryfikacja agenta

- Host, który chce zweryfikować agenta, wysyła mu swój udział
- Agent "odzyskuje" resztę udziałów ze swojego kodu i stanu wykonywania i wylicza sekret σ.
- Agent wykorzystuje σ teraz jako sekret w protokole GQ:
  - Wysyła do hosta losowe wyzwanie u: $u \equiv r^{K_P} \bmod N$
    - Na podstawie losowej liczby r
  - Po otrzymaniu r, host generuje losową liczbę b
  - Agent wylicza wartość v na podstawie b i sekretu σ:

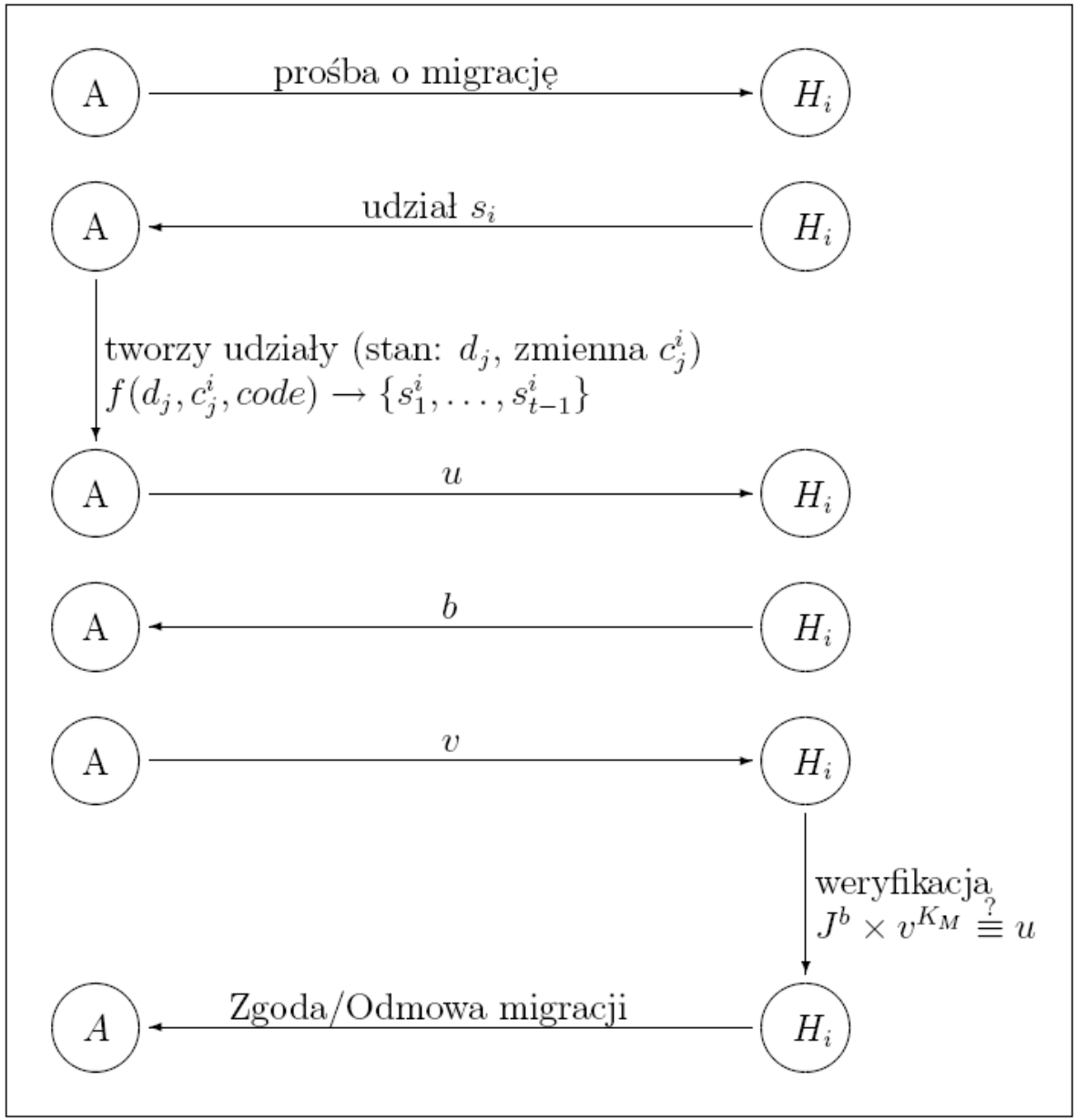
$$v \equiv r \cdot \sigma^{b} \bmod N$$

  - Host sprawdza poprawność v, sprawdzając poniższą równość:

$$J_P^b \cdot v^{K_P} \equiv u \bmod N$$

  - Jeśli jest prawdziwa, to agent zna sekret, a jego stan lub kod nie zostały zmienione.

Weryfikacja danych



Weryfikacja danych

- Podobny scenariusz może zostać wykorzystany do zabezpieczenia danych.
  - Złośliwy host może manipulować danymi wyliczonymi u innych hostów
  - Wykorzystując GQ dla każdych danych *d*, agent może wylosować
    $$r \in \{1,\ldots,N-1\}$$
  - I wyliczyć $v \equiv r \cdot \sigma^d \bmod N$
  - Manager może sprawdzić, że dane nie zostały zmienione, obliczając
    $$J_P^d \cdot v^{K_P} \equiv u \bmod N$$
- W ten sposób dla każdych danych *d* agent będzie miał unikalny dowód, że dane są poprawne.

Analiza bezpieczeństwa

- Proponowany protokół powinien
  - Wykorzystywać wiele tożsamości
  - Jeden sekret dla jednego hosta
- Zakładamy, że host może manipulować agentem i jego danymi
  - Nie znając innych sekretów wbudowanych poza własnym, nawet odzyskawszy udziały z agenta nie są w stanie odtworzyć sekretów dla innych hostów
  - Kolejny host wykryje manipulację
- Dzięki zabezpieczeniu danych, host nie może manipulować danymi uzyskanymi od innych hostów: nie umie stworzyć poprawnego *v,* nie znając sekretu.
- Proponowane rozwiązanie spełnia warunki *forward integrity*.
- Rozwiązanie nie może przeciwdziałać zniszczeniu agenta lub uszkodzeniu kodu.

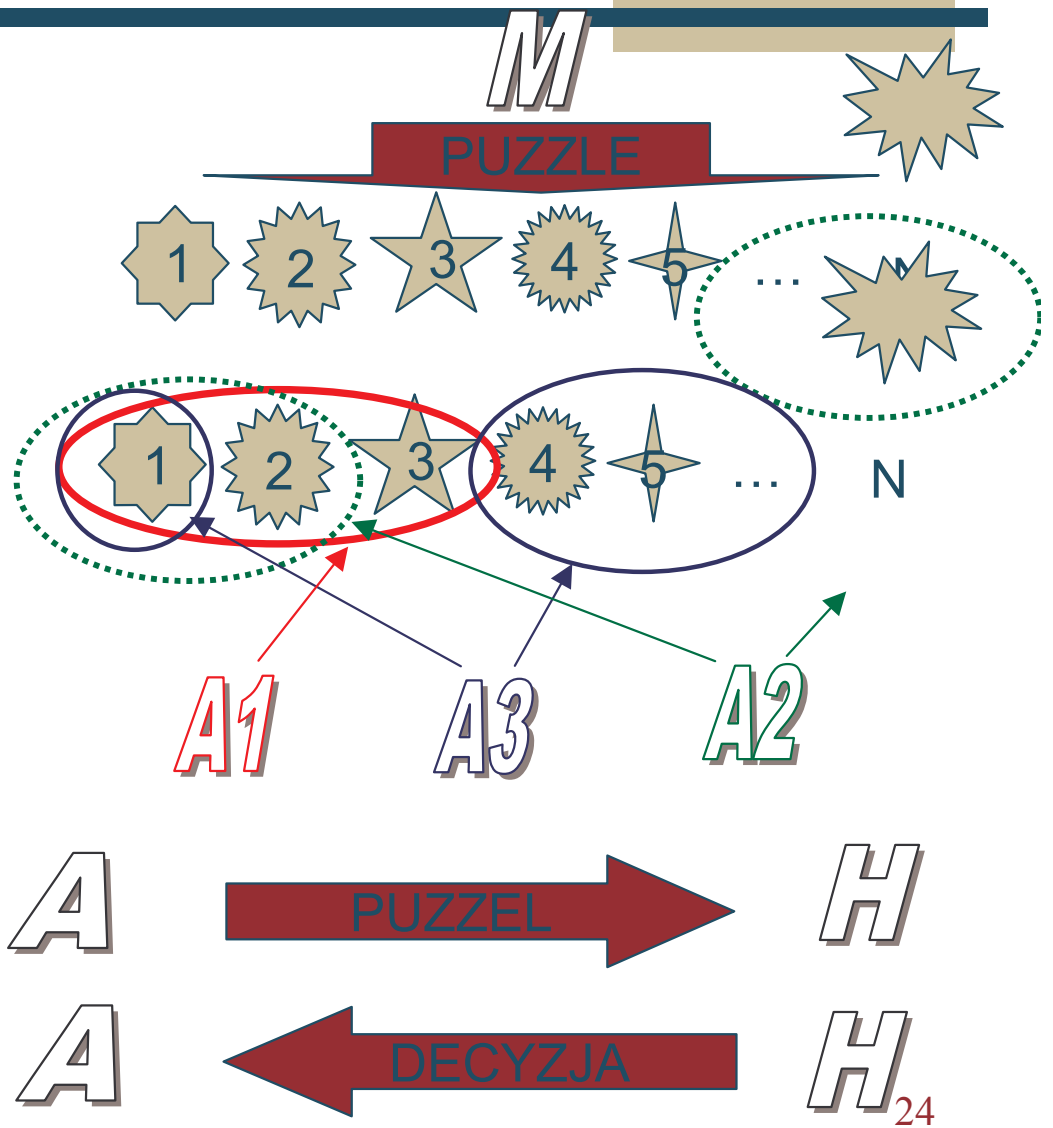
Anonimowi agenci

Architektura bezpieczeństwa dla systemu agentowego z chroniąca prywatność i anonimowość

- Wymagania
  - Anonimowość dla agenta względem innych agentów i hostów
  - Bezpieczne uwierzytelnienia
  - Rozliczalność
- Zalety
  - Żaden host (samodzielnie lub działając w grupie) nie może zidentyfikować żadnego agenta w systemie
  - Manager może zidentyfikować każdego agenta – wprowadzenie TTP umożliwia rozliczalność
  - Ataki typu play-back są nieskuteczne
  - Uwierzytelnienia – silne algorytmy kryptograficzne
  - Dodatkowe możliwości
    - Uzgodnienie klucza

Ogólny schemat

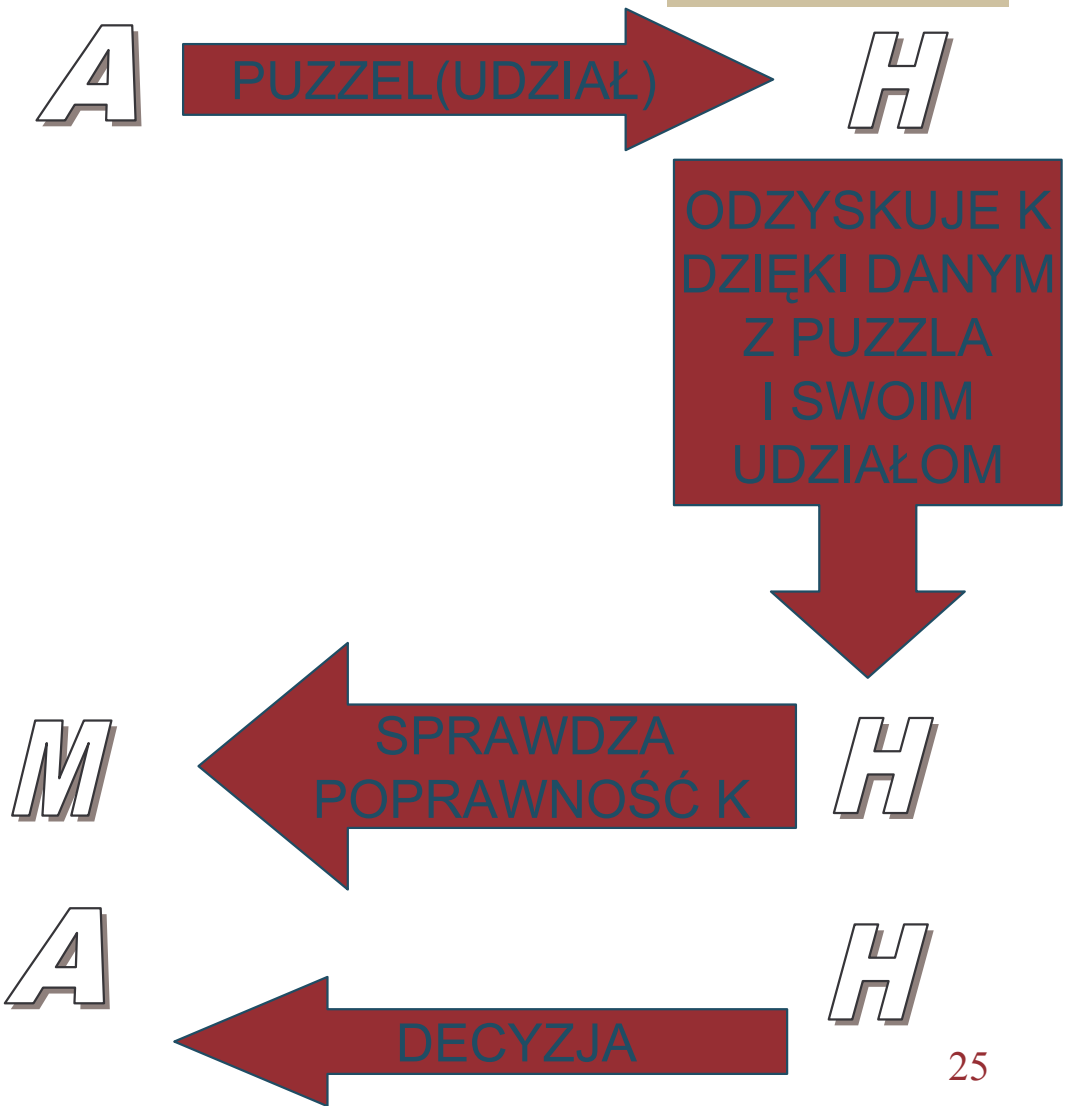
- Faza wstępna
  - Manager tworzy zbiór puzzli
  - Każdy puzzel zawiera „wartość uwierzytelniającą”
  - Puzzle są dystrybuowane między agentów
- Faza uwierzytelnienia
  - Agent wysyła puzzla do hosta
  - Host rozwiązuje puzzla i sprawdza jego poprawność



System z podziałem sekretu

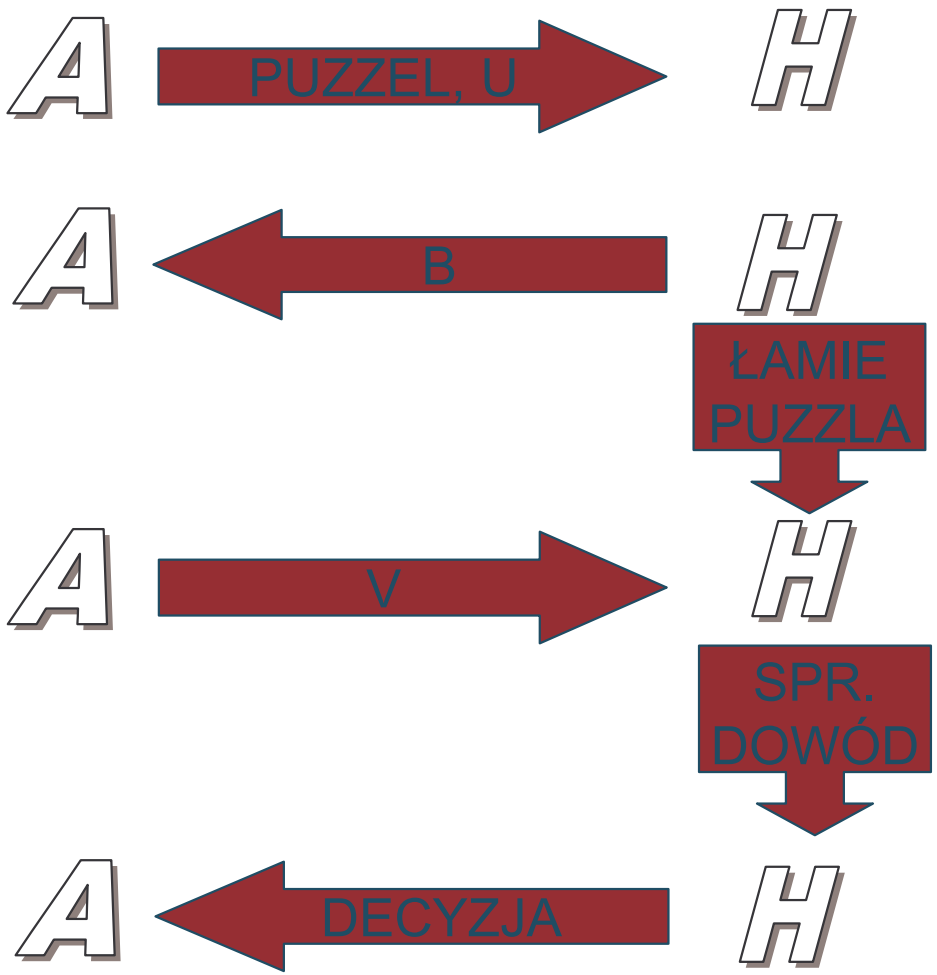
- Faza wstępna
  - Manager tworzy sekret

    i rozdystrybuuje udziały pomiędzy agentów i hosty
- Uwierzytelnienie
  - Agent wysyła swój udział „zapakowany” w puzzel
  - Host odtwarza sekret
  - Host potwierdza ważność sekretu z managerem



System z ZKP

- Faza wstępna
  - Manager tworzy sekrety dla każdego agenta i wylicza wartości publiczne
- Uwierzytelnienie
  - Agent wysyła puzzla i wyzwanie U
  - Host odpowiada wartością B i rozwiązuje puzzla
  - Agent odpowiada V
  - Host sprawdza czy V jest poprawne (czy agent zna odpowiedni sekret)

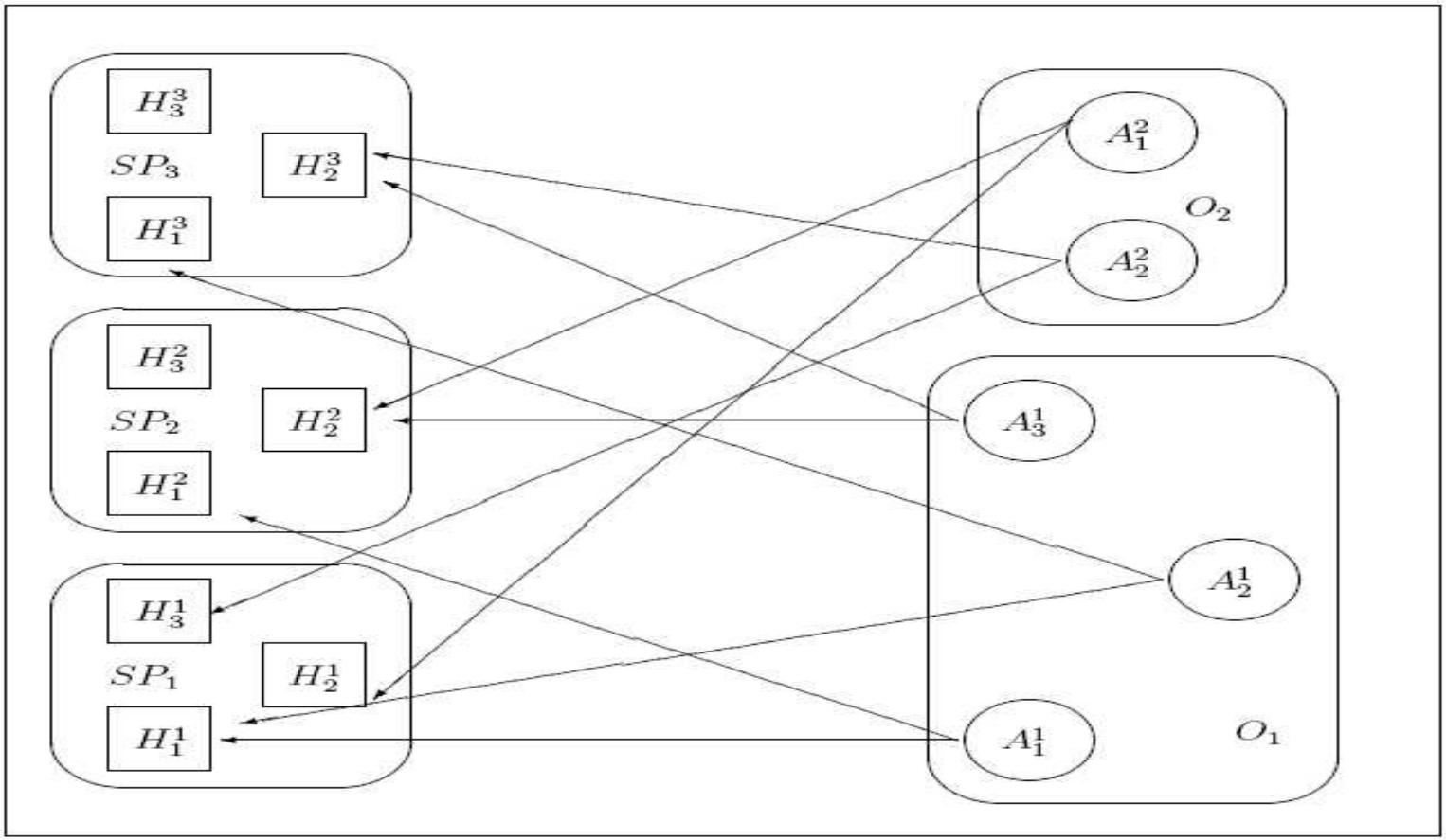


System agentowy z odwoływalną anonimowością

Uwierzytelnienie grupowe z częściową anonimowością

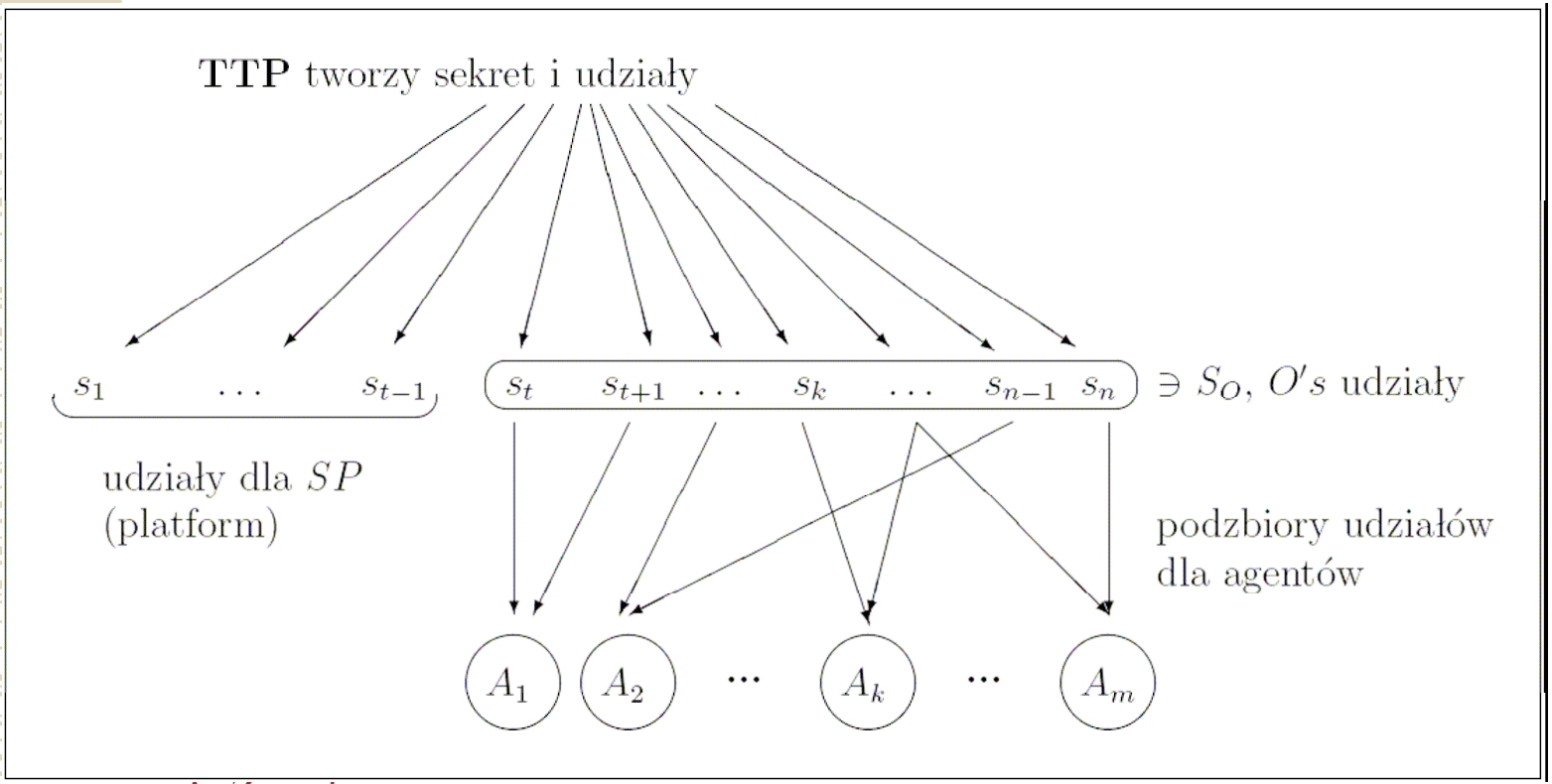
- Architektura
  - Organizacja (klient/usługobiorca)
    - reprezentowany poprzez agentów
  - Trusted authority – w organizacji
  - Operator/Service provider – udostępnia platformy agentowe
- Cechy
  - Anonimowość agentów (np. klienta) względem usługodawcy
  - Możliwość „odwołania anonimowości”
  - Uwierzytelnianie grupy

Organizacje i SP



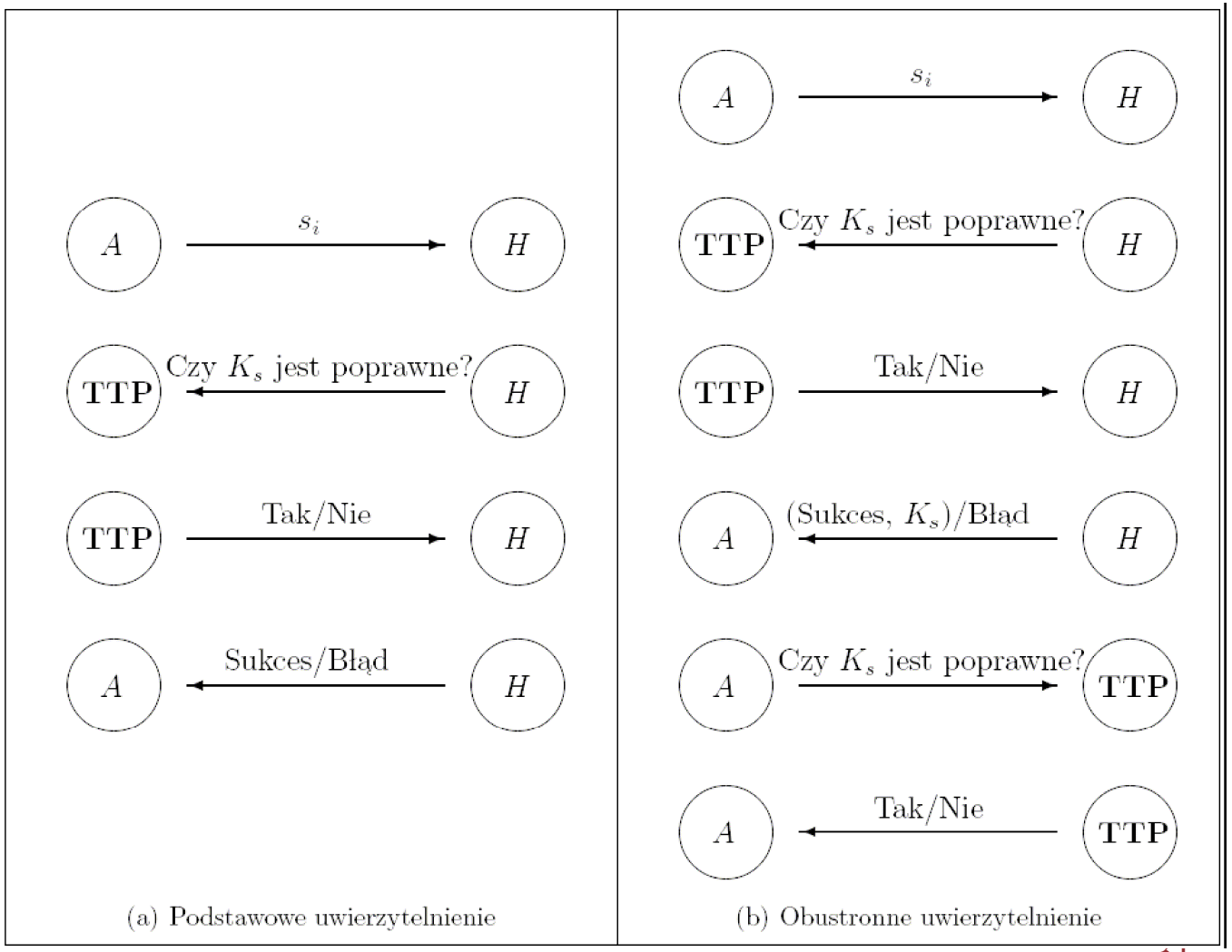
Ogólny schemat

- Oznaczenia
  - O, SP, TA_O
- Faza wstępna
  - Zaufana strona generuje udziały dla SP i O
  - TA_O rozdziela udziały między użytkowników



Ogólny schemat

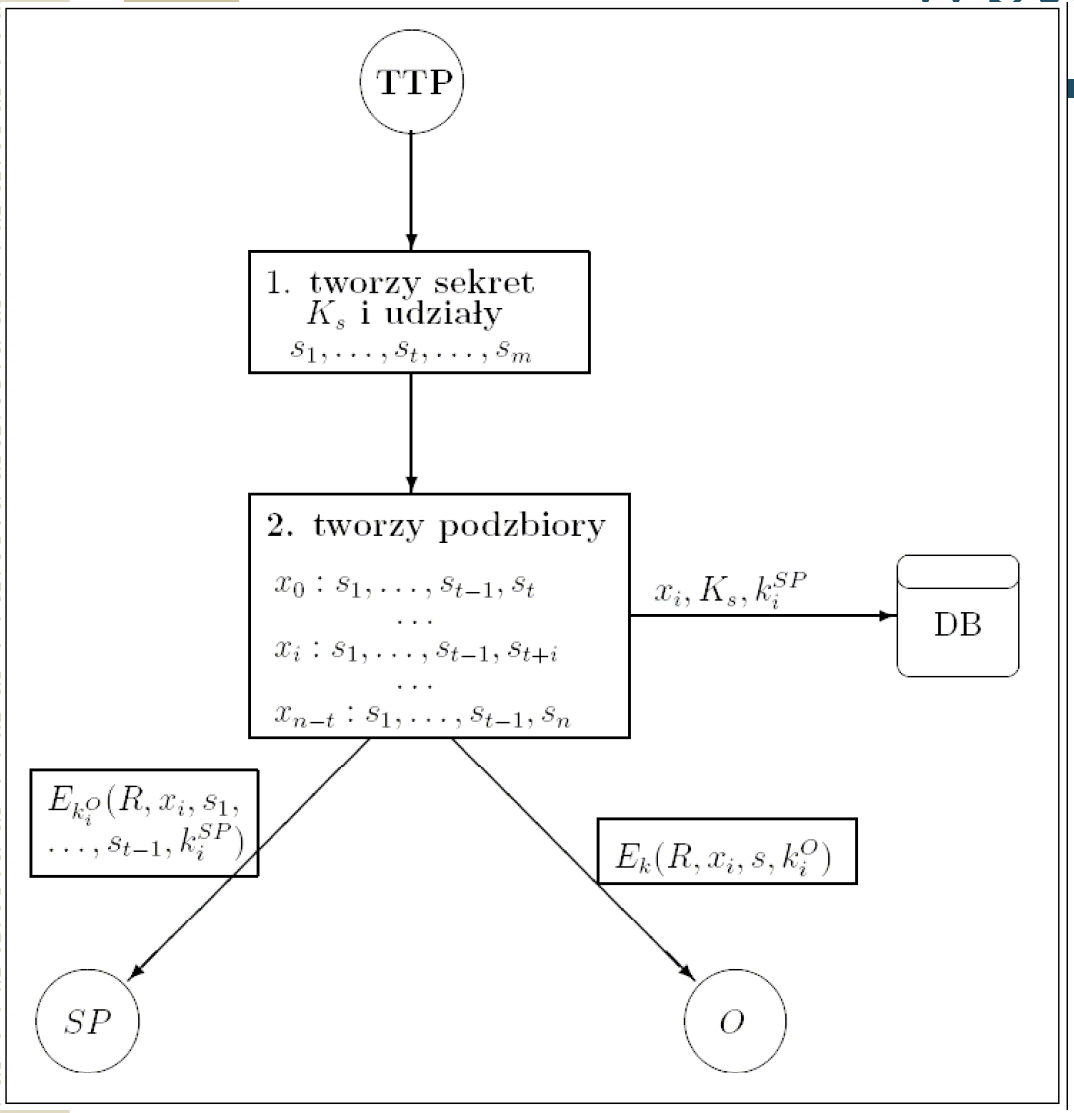
- Udział jest wysyłany przez użytkownika do SP
- SP wylicza sekret i sprawdza go z TTP

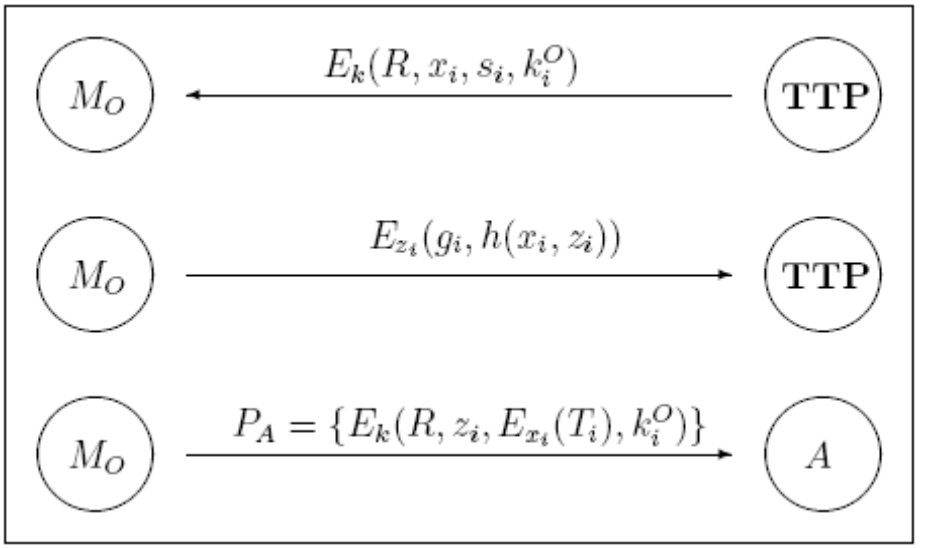


Usprawnienia

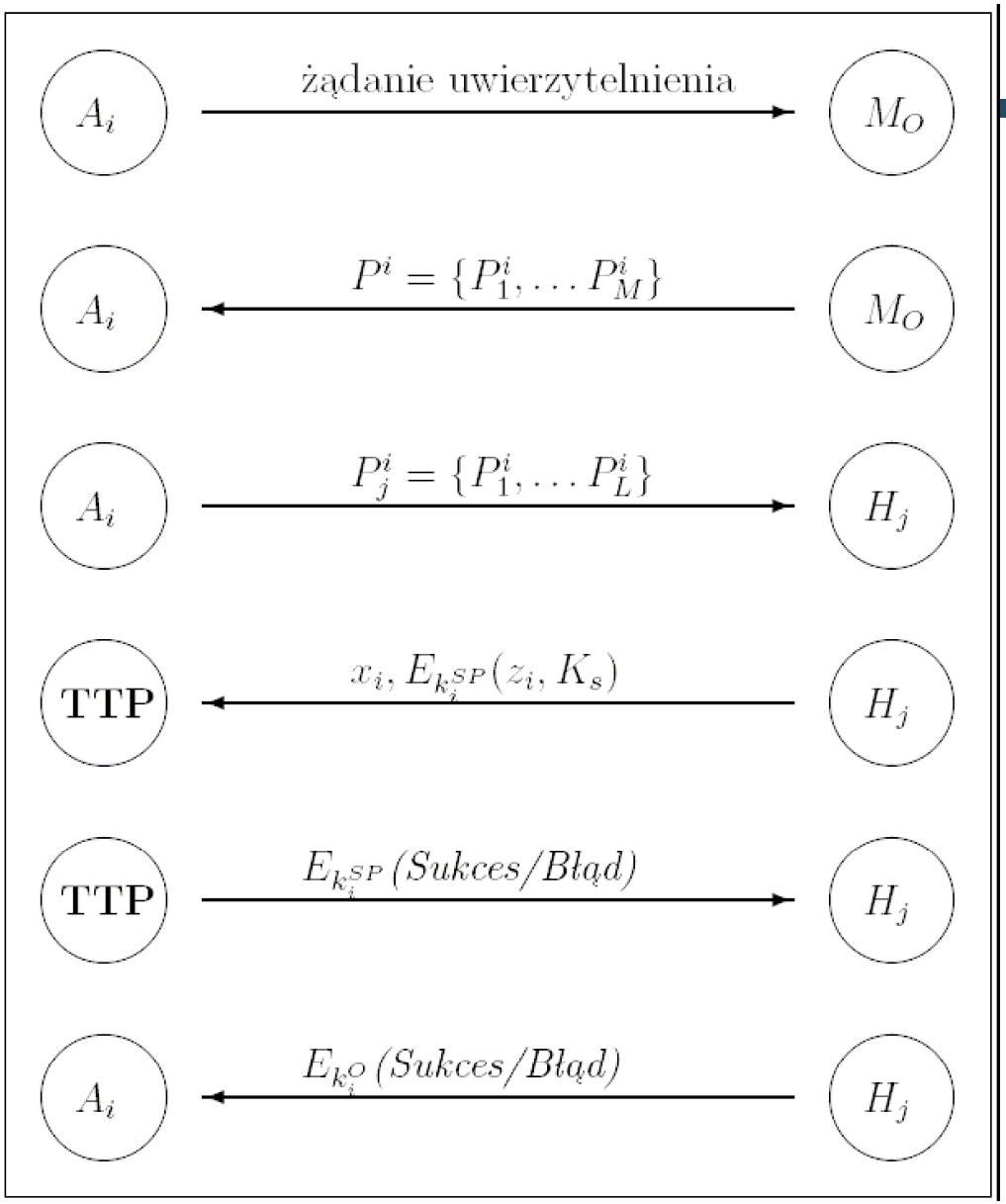
- Usprawnienia
  - Bezpieczny kanał
    - Puzzle
      - Stworzenie podzbiorów udziałów, które umożliwiają odzyskanie sekretu
  - Wielokrotne użycie udziałów
    - Znaczniki czasowe: $T_i = \{s_i, t_i, p_i\}$

Faza wstępna





Uwierzytelnienie



Uwagi

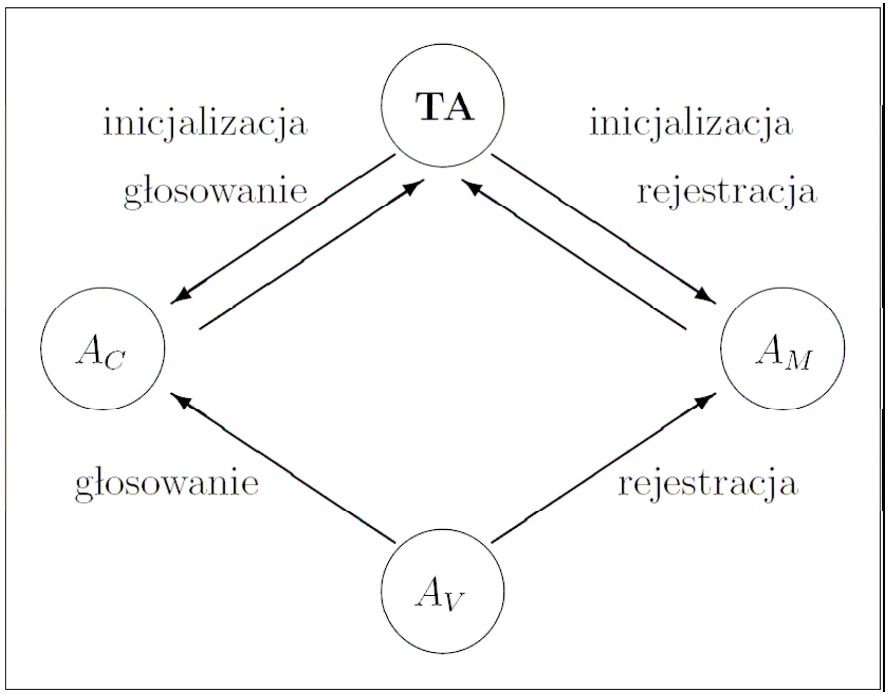
- Nie wymaga zaufania między O a SP
- Zachowanie możliwości późniejszej identyfikacji użytkownika (za pomocą TA_O)
- Odporność na ataki (powtórzenie)
- Zapewnia uwierzytelnienie i anonimowość

Zastosowania

Elektroniczne głosowanie

EVAS: E-Voting Agent System

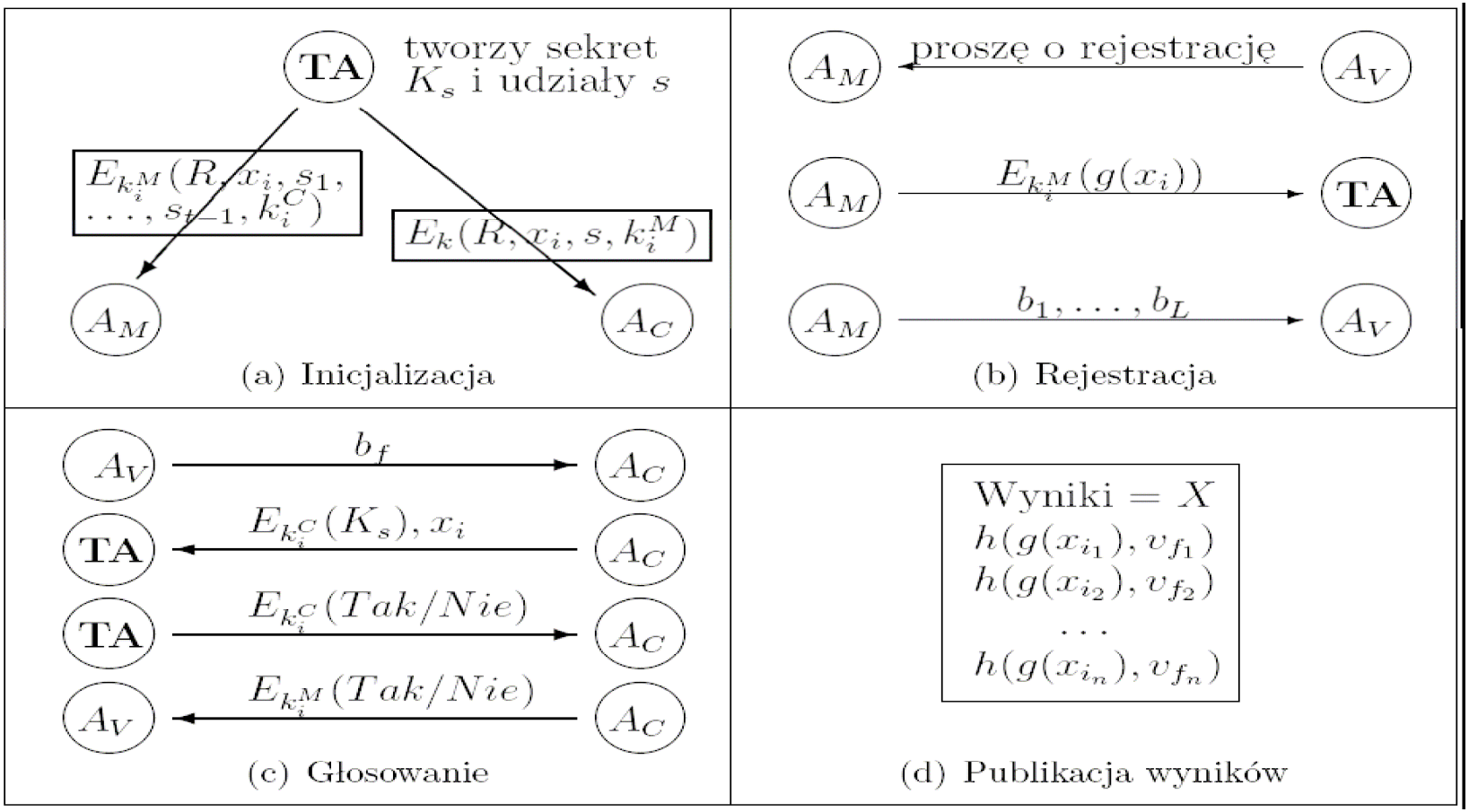
- System głosowania elektronicznego
  - Oparty na agentach, gdyż:
    - Elastyczni
    - Skalowalni
    - Wiele możliwych zastosowań
  - Architektura: *mix-nets*
  - Największa zaleta:
    - Użytkownik nie musi wykonywać jakichkolwiek obliczeń – głos może zostać wysłany do licznika w dowolny sposób
    - Wydajny, skalowalny elastyczny, bezpieczny



Zarys pomysłu

1. **TA** tworzy zbiór danych uwierzytelniających oraz listę zarejestrowanych użytkowników i wysyła ją do agent A_M (*mix'a*).
2. Dla każdego głosującego *mix* tworzy „glosy" (ang. *ballot*) z danych otrzymanych od **TA** (po uprzednim sprawdzeniu czy głosujący jest uprawniony do głosowania).
3. Użytkownik wysyła swój wybrany glos do licznika A_C: agent sprawdza dane uwierzytelniające z **TA** i następnie dodaje do wyników, publikując na koniec odpowiednie dowody poprawności.

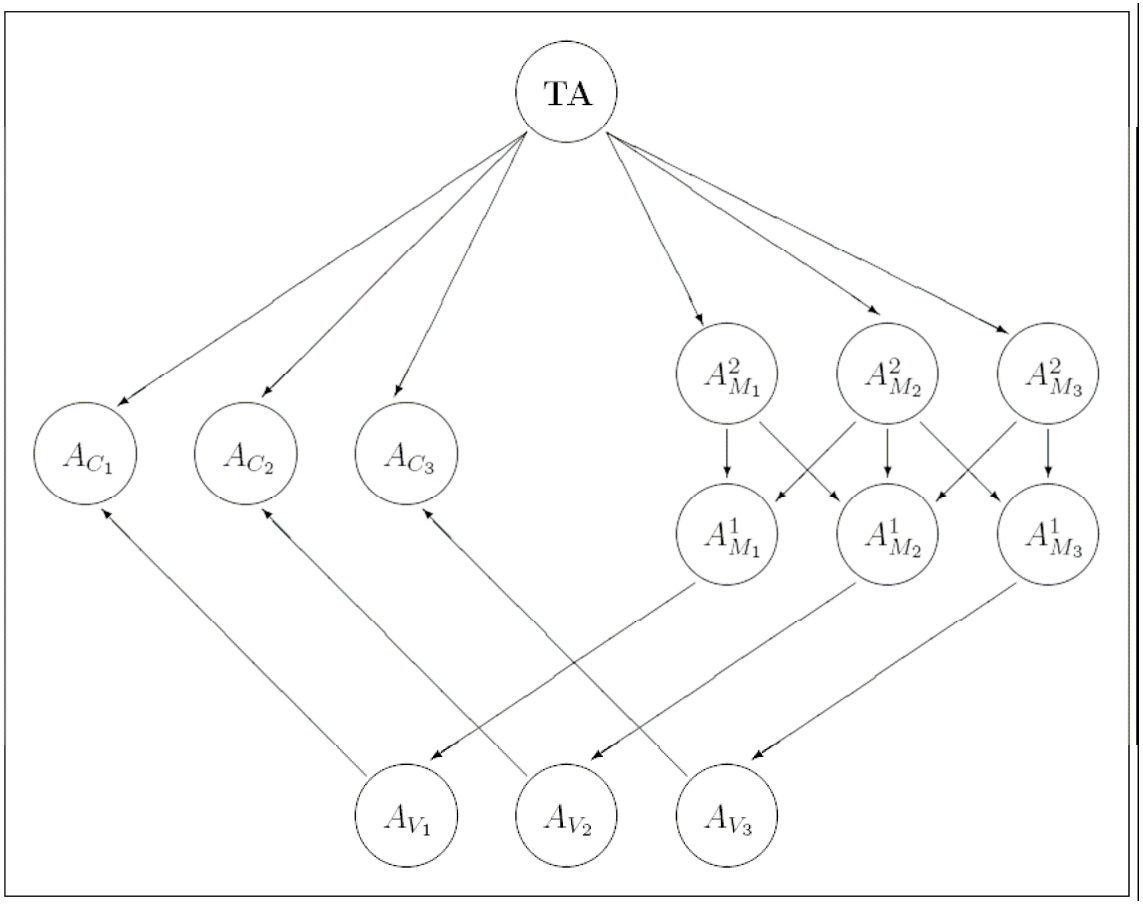
EVAS: E-Voting Agent System

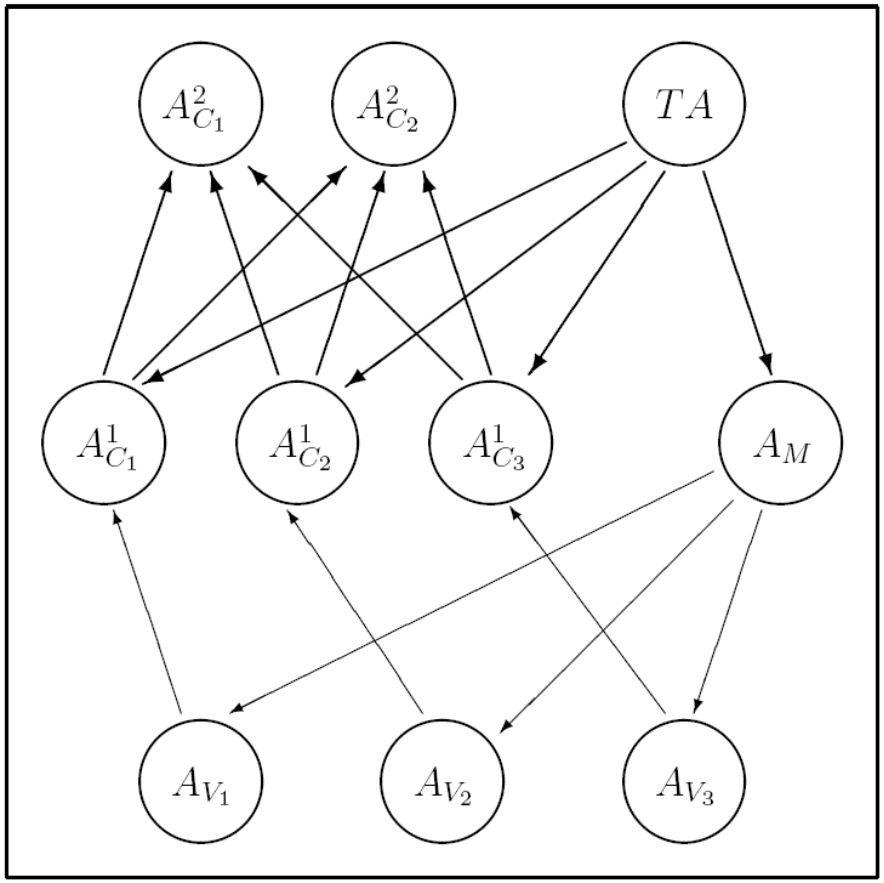


Bezpieczeństwo

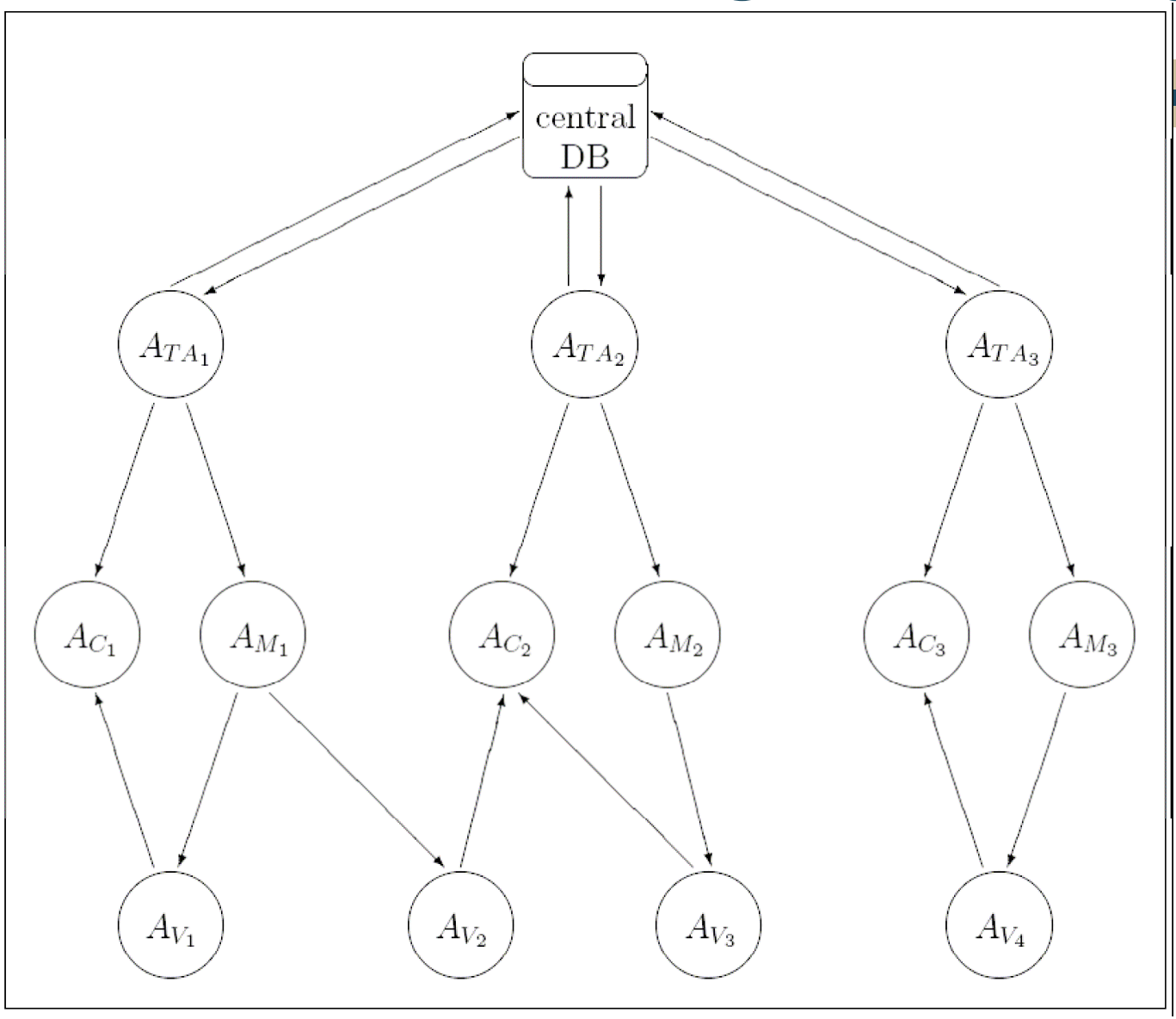
- Spełnione wymagania:
  - Zupełność (Completeness)
  - Poprawność (Soundness)
  - Prywatność (Privacy)
  - Niepowtarzalność (Un-reusability)
  - Weryfikowalność (Verifiability)
  - Brak dowodu głosowania (Receipt-freeness)
  - Odporność (Robustness)
  - *Równoprawność (Eligibility)*
  - *Weryfikowalność przez gosującego (Voter verifiability)*
- Niespełnione wymagania:
  - Nieprzekupywalność (Incoercibility)
  - Uczciwość (Fairness)

Architektura – rozproszone zaufanie





Architektura - regionalizacja



Scenariusze

Mieszacz 1

TTP

Licznik 1

Licznik 2

Mieszacz 2

Pytania?