Techniki zabezpieczania kodu – wzorce programowe

Temat IV

Wzorce programowe (projektowe)

- Rozwiązanie projektowe
  - uniwersalne,
  - sprawdzone w praktyce,

  dla
  - często pojawiających się,
  - powtarzalnych problemów projektowych.

Wzorce projektowe dla bezpieczeństwa

- Według The Open Group
- Główny podział
  - Available System Patterns – nacisk na dostęp do usług
  - Protected System Patterns – nacisk na ochronę skarbów

Available System Patterns

- Checkpointed System – odtwarzanie sensownych stanów
- Standby – przywracanie z innego komponentu
- Comparator-Checked Fault-Tolerant System – szybkie wykrywanie awarii komponentów, niezależne awarie komponentów nie powodują awarii systemu
- Replicated System – możliwa obsługa z wielu punktów, zapewnione zastępowanie
- Error Detection/Correction – dodanie redundancji do danych

Protected System Patterns

- Protected System – wszystkie dostępy do zasobów za pomocą strażnika
- Policy – zasady bezpieczeństwa wyizolowane w osobny komponent
- Authenticator – rozszerzenie autentykacji (do zdefiniowania później)
- Subject Descriptor – dostęp do atrybutów bezpieczeństwa aktywnej jednostki procesu (reprezentacji aktora)

Protected System Patterns c.d.

- Secure Communication – zapewnienie, że wzajemne zasady bezpieczeństwa są przestrzegane
- Security Context – opakowanie dla atrybutów bezpieczeństwa związanych z danym momentem wykonania, procesem, operacją czy akcją
- Security Association – każdy uczestnik Secure Communication ma informacje potrzebne do zabezpieczania i sprawdzania zabezpieczenia
- Secure Proxy – relacja między dwoma Protected Systems, gdy jeden jest zawarty w drugim

Metodologia projektowania

- Zaprojektuj system ze względu na funkcjonalność
- Zidentyfikuj komponenty: interfejsy, banki danych, łącza komunikacyjne, protokoły
- zastosuj Available System Sequence
- zastosuj Protected System Sequence

Stosowanie Available System Sequence

- zidentyfikuj krytyczne komponenty
- jeśli popsucie danych może spowodować awarię systemu, zastosuj Error Detection/Correction
- jeśli awaria konkretnego komponentu musi być wykryta i naprawiona natychmiast, aby zapewnić dostępność, zastosuj wzorzec Comparator-Checked Fault-Tolerant System
- jeśli awaria konkretnego komponentu jest dopuszczalna, ale obsługa musi być ciągła, zastosuj wzorzec Standby
- jeśli dopuszczalny jest częściowy brak obsługi lub brak obsługi na jakimś terenie, to zastosuj wzorzec Replicated System

Stosowanie Protected System Sequence

- zidentyfikuj zasoby (co ma być chronione) i aktorów (osoby sięgające po zasoby),
- określ wzorce Protected System tak, że
  - każdy zasób jest w zakresie jakiegoś Protected System
  - różne PS są albo rozłączne, albo zagnieżdżone (nienachodzące na siebie)
- dla zagnieżdżonych PS użyj wzorca Secure Proxy do określenia relacji między nimi
- dla każdego PS zdefiniuj wzorzec Policy

Stosowanie Protected System Sequence c.d.

- dla każdej Policy zdefiniuj wzorce Secure Communication
  - między rozłącznymi PS,
  - między aktorami a PS
- określ, jak Policy będzie opisywać i identyfikować zasoby
- na podstawie Policy opracuj Security Context dla każdego z końców połączenia Secure Communication,
- dla każdego Security Context opracuj Security Association oraz Subject Descriptor,
- sprawdź każdy Security Context, czy nie trzeba zmienić go w PS z wieloma Security Contexts.

Przegląd

- Po przejściu przez powyższą listę zastanów się:
  - Jakie są najsłabsze elementy projektu zwn. bezpieczeństwo?
  - Co można uprościć, aby w tych miejscach się zabezpieczyć?

Przegląd – rozłożenie

- Różne rodzaje zasobów – różne zasady dostępu (może połączyć)
- Czy zabezpieczane zasoby są rozłożone między różne systemy? (może scentralizować)

Przegląd – ułożenie

- Czy jest dużo zagnieżdżonych Protected Systems?
- Czy jest dużo Secure Communications?
- Czy ta mnogość wpływa na prostotę Policies?

Checkpointed System

- Struktura: Recovery Proxy, Recoverable Component, Memento
- Operacje wykonywane przez Recovery Proxy:
  - pośredniczenie w działaniach na Recoverable Component
  - zapisuje stan Recoverable Component w Memento,
  - w razie czego odtwarza stan Recoverable Component z Memento
- Logika operacji w Recoverable Component

Standby

- Struktura: Recovery Proxy, Active Component, Standby Component, Memento
- Operacje wykonywane przez Recovery Proxy:
  - pośredniczenie w działaniach na Active Component
  - zapisuje stan Active Component w Memento,
  - w razie awarii przenosi przetwarzanie do Standby Component
- Logika operacji w Active/Standby Component
- Standby Component odtwarza stan z Memento

Comparator-Checked Fault-Tolerant System

- Struktura: Recoverable Component 1, Recoverable Component 2, Comparator, Memento 1, Memento 2
- Recoverable Components - zapisują okresowo stany w Memento
- Comparator w wyróżnionych momentach porównuje Memento, reaguje, jak są różne

Replicated System

- Struktura: Replica 1, Replica 2, Workload Management Proxy
- Replica - wykonuje operacje logiczne
- Workload Management Proxy - rozdziela obciążenie, reaguje przekierowaniem na sygnał o awarii repliki

Error Detection/Correction

- Struktura: Error Control Proxy, Redundant Media/Link
- Error Control Proxy - pośrednik dodający redundancję i sprawdzający bezbłędność/korygujący
- Redundant Media/Link - medium lub kanał, w którym można dodać redundantne dane

Protected System

- Struktura:
  - [wersja 1] jeden strażnik do wszystkich chronionych zasobów,
  - [wersja 2] po jednym strażniku dla każdego typu zasobu.
- Uczestnicy: Client, Guard, Policy, Resource
- Guard sprawdza, czy Client sięga po Resource z zachowaniem Policy

Policy

- Uczestnicy: Client, Authenticator, Guard, Securit Context, Policy, Rule
- Policy - obejmuje zestaw Rules, które decydują o przyznaniu dostępu do zasobów w oparciu o:
  - atrybuty, jakie ma Client,
  - atrybuty, jakie ma żądanie,
  - atrybuty, jakie ma punkt docelowy,
  - atrybuty, jakie ma kontekst.

Authenticator

- niezdefiniowany

Subject Descriptor

- Uczestnicy: Attribute List (atrybuty mają typy)

Secure Communication

- Uczestnicy: Communicating Party, Communication Channel, Communication Protection Proxy
- Organizacja:
  - [wersja 1] Communication Protection Proxy - między Party a Channel
  - [wersja 2] Communication Protection Proxy - Party wywołuje obok

Security Context

- Uczestnicy: Communication Protection Proxy, Security Context, Subject Descriptor
- Przechowuje informacje związane z bezpieczeństwem dotyczące pojedynczego aktora – np. długoterminowe klucze do szyfrowania

Security Association

- Uczestnicy: Protection Proxy, Security Association, Security Context
- Struktura: Association Identifier, Partner Identifier, Association Expiration, Cryptographic Keys, Quality of Protection Settings, Delegation Tokens
- Utrzymuje dane związane z sesją pracy

Secure Proxy

- Uczestnicy: Requesters, Secure Proxy, Resources
- Możliwe sposoby organizacji:
  - pośrednictwo per odwołanie,
  - pośrednictwo w autoryzacji,
  - pośrednictwo przez oddanie panowania,
  - delegacja na całą sesję.

Inne wzorce

- NIANIO: wydzielenie operacji bezpieczeństwa w jedno lokalne miejsce
- Observer: śledzenie działania aspektu oprogramowania.