Uživatelská příručka

Implementace komunikačních rozhraní v monitorovacích systémech přenosových soustav

Software byl vyvinut v rámci projektu „Národní centrum kompetence pro kyberbezpečnost“ č. TN01000077 a byl spolufinancován se státní podporou Technologické agentury v ČR v rámci programu Národní centra kompetence 1.

Obsah[1]

[1] Verze manuálu: 03/2021

1. Zprovoznění aplikace TASE.2 adaptéru

V následujícím textu bude krok za krokem popsáno, jak rozchodit aplikaci **usy-ocberos-tase2adapterg01**.

1.1 Prerekvizity

Pro fungování TASE.2 adaptéru je potřeba mít někde nasazenou aplikaci **usy-ocberos-dataprocg0**, na kterou se z adaptéru posílají data. Dále je třeba mít následovní přístupy:

1. Vytvořený Plus4U account
2. *Přístup na GIT repositář*

   ssh://git@codebase.plus4u.net:9422/usy_ocberos_tase2adapterg01.git
3. Přístup na Docker repositář

   usylibra/usy_ocberos_tase2adapterg01

1.2 Příprava prostředí

Nejprve je potřebné připravit prostředí jak na cílovém počítači (tam kde bude běžet adaptér), tak i na lokálním počítači (ze kterého pracujeme). Může taky jít o ten samý počítač.

1. Nainstalovat Docker na cílový počítač. Způsob instalace se může lišit v závislosti od operačního systému. Instrukce pro instalaci na jednotlivé operační systémy je možné najít zde - https://www.docker.com/products/docker-desktop
2. Zkontrolujte, zda byl na cílovém počítači nainstalován také program Docker Compose, nejjednodušeji příkazem ***docker-compose -version***. Jestli nebyl, řiďte se instrukcemi zde - https://docs.docker.com/compose/install/
3. Na lokálním počítači stáhnete GIT repozitář:

   ssh://git@codebase.plus4u.net:9422/usy_ocberos_tase2adapterg01.git

   Z repozitáře zkopírujte následovní soubor na cílový počítač:

   usy_ocberos_tase2adapterg01-server/docker/docker-compose.yml.
4. Vejděte do adresáře se zkopírovaným souborem a zkontrolujte ho. Soubor obsahuje konfiguraci jak pro adaptér samotný, tak i pro MongoDB a RabbitMQ, na kterých je adaptér závislý a taky budou běžet v rámci Dockeru. Důležité jsou zejména hodnoty těchhle parametrů:
   - asid – identifikátor aplikace
   - awid – identifikátor aplikačního workspace
   - asidOwner – identifikátor správce aplikace
   - uuEE.uuIdentity – identifikátor technického uživatele pro komunikaci s **usy-ocberos-dataprocg0** aplikací
   - uuEE.ac1 – první heslo uživatele uuEE.uuIdentity
   - uuEE.ac2 – druhé heslo uživatele uuEE.uuIdentity
   - dataproc.server.url – adresa, kde běží **usy-ocberos-dataprocg0** aplikace
5. Na lokálním počítači nainstalujte program Insomnia pro spouštění HTTP requestů - https://insomnia.rest/

6. Importujte Insomnia workspace. Kliknete na Application -> Preferences -> Data -> Import Data -> From File a vyberte soubor v GIT repozitáři **usy_ocberos_tase2adapterg01-server/src/test/insomnia/insomnia-workspace.json**

1.3 Inicializace aplikace

Když už máme nainstalovány potřebný programy a připraveny soubory, můžeme spustit aplikaci.

1. Na cílovém počítači najděte adresář se zkopírovaným souborem **docker-compose.yml** a spusťte příkaz ***docker-compose up -d***. Tím se nastartuje adaptér i DB a MQ v pozadí. Příkazem ***docker ps*** je možné zkontrolovat stav běžících containerů a taky jejich ID. Příkazem ***docker logs {container_id}*** je pak možné sledovat jejich logy. Plná dokumentace příkazů na oficiální stránce.

   https://docs.docker.com/engine/reference/commandline/cli/

2. Spusťte Insomnii, otevřete importovaný workspace a vytvořte nový environment, nebo upravte existující. Změňte hodnoty dle konfigurace aplikace (asid, awid, asidOwner, awidOwner)
3. Získejte autentizační token, využitím requestu “Grant Token”. Insomnia vyžádá Access Code 1 a Access Code 2. Je potřeba, aby to byli kódy uživatele, který je jako asidOwner.
4. Získaný Token zkopírujte a vložte ho do proměnné **asidOwnerToken** v Basic Environment.
5. Spusťte request **sys/initApp** v složce **uuAppWorkspace**
6. Spusťte request **sys/initAppWorkspace** v složce **uuAppWorkspace** a po vyžádaní zadejte 2x id uživatele awidOwner
7. Jestli asidOwner a awidOwner bude ten samý uživatel, doplňte v Basic Environment ten samý token do proměnných **token** a **awidOwnertoken**. V opačném případe znova vygenerujte Token s přístupovými údaji jiného uživatele a využijte ten.
8. Spusťte request **init** ve složce **usyOcberosTase2adapterg01**
9. Spusťte request **taskScheduler/set** ve složce **usyOcberosTase2adapterg01/Task Scheduler**, jako atribut **lastRun** stačí jakýkoliv datum v minulosti
10. Přidejte povolení pro uživatele, který budou v budoucnu využívat REST rozhraní adaptéru (jestli jde o jiné než asidOwner a awidOwner) a to vykonáním requestu **sys/createPermission** ve složce **uuAppWorkspace**. Seznam požadovaných uživatelů zadejte do atributu **uuIdentityList**.

1.4 Konfigurace PUSH režimu

Konfigurovat TASE.2 komunikaci je možné jen pro inicializovanou aplikaci, tedy poté co se vykonali kroky v předešlých kapitolách. Režim PUSH znamená, že adaptér funguje jako pasivní prvek a očekává přicházející správy od jednotlivých stanic. Pro konfiguraci je potřeba několik kroků.

1. Spusťte Insomnii a otevřete importovaný workspace
2. Vygenerujte token pro svého uživatele (musí mít ovšem práva na používaní aplikace – role Authorities, nebo AwidOwner) a token zkopírujte do proměnné **token** v Basic Environment

3. Spusťte request **initPassiveComm** ve složce **usyOcberosTase2adapterg01.** Konfiguraci v JSON formátu změňte dle potřeby.
4. Restartujte aplikaci pomocí příkazu ***docker-compose restart tase2adapterg***. Restart je nutný z důvodu, že konfigurace pro PUSH režim se čte z databáze pouze při startu aplikace.

Příklad JSON konfigurace **push** režimu:

```
{
    "stations": [
        {
            "apTitle": "1.1.997.1",
            "aeQualifier": 12,
            "domain": "icc1"
        },
        {
            "apTitle": "1.1.997.2",
            "aeQualifier": 12,
            "domain": "icc2"
        },
        {
            "apTitle": "1.1.997.3",
            "aeQualifier": 12,
            "domain": "icc3"
        }
    ]
}
```

Jde o seznam stanic, ze kterých má adaptér přijímat správy, pro každou evidujeme tři údaje:

- apTitle – první identifikátor stanice
- aeQualifier – druhý identifikár stanice
- domain – identifikátor domény (jakoby workspace), do které bude klient zapisovat

1.5 Konfigurace PULL režimu

Stejně jako při PUSH režimu i tady je nutné mít nejdřív inicializovanou aplikaci. Režim PULL znamená, že adaptér se aktivně dotazuje jednotlivých stanic, aby od nich získal data. Pro konfiguraci je znova potřeba několik kroků.

1. Spusťte Insomnii a otevřete importovaný workspace
2. Vygenerujte token pro svého uživatele (musí mít ovšem práva na používaní aplikace – role Authorities, nebo AwidOwner) a token zkopírujte do proměnné **token** v Basic Environment
3. Spusťte request **initActiveComm** ve složce **usyOcberosTase2adapterg01.** Konfiguraci v JSON formátu změňte dle potřeby.

Příklad JSON konfigurace **pull** režimu:

```
{
    "stations": [
        {
            "address": "172.30.0.11",
            "port": 102,
            "apTitle": "1.1.998.1",
            "qualifier": 12,
            "variableName": "Frequency",
            "domain": "icc1",
            "queryPeriod": 1
        },
        {
            "address": "172.30.0.12",
            "port": 102,
            "apTitle": "1.1.998.2",
            "qualifier": 12,
            "variableName": "Frequency",
            "domain": "icc1",
            "queryPeriod": 1
        },
        {
            "address": "172.30.0.13",
            "port": 102,
            "apTitle": "1.1.998.3",
            "qualifier": 12,
            "variableName": "Frequency",
            "domain": "icc1",
            "queryPeriod": 1
        }
    ],
    "commDuration": 3
}
```

Jde o seznam stanic, kterých se má adaptér dotazovat. Pro každou stanici evidujeme sedm údajů:

- address – IP adresa stanice

- port – port na kterém stanice poslouchá
- apTitle – první identifikátor stanice
- qualifier – druhý identifikátor stanice
- variableName – název požadované proměnné
- domain – doména ve které proměnná leží
- queryPeriod – interval, jako často se na hodnotu dotazovat (v sekundách)

Navíc se nastavuje ještě společná hodnota commDuration, která vyjadřuje dobu po kterou se má dotazováni vykonávat v minutách.

2. Příprava prostředí pro vývoj TASE.2 adaptéru

2.1 Prerekvizity

- Java 8 nebo Java 11
- Docker https://www.docker.com/
- Gradle https://gradle.org/
- Insomnia https://insomnia.rest/
- Idea (nebo jiné IDE)
- Účet v Plus4U (https://www.plus4u.net/)

2.2 Konfigurace

1. Stáhnete project z adresy

   ssh://git@codebase.plus4u.net:9422/usy_ocberos_tase2adapterg01.git

2. Vytvořte property soubor **development.properties** a **production.properties** v adresáři ***C:\\Users\{user}\.uu\config***
3. Přidejte do obou souborů tyhle atributy s hodnotou dle vašeho účtu
   - asidOwner=
   - yourAccessCode1=
   - yourAccessCode2=
4. Registrujte TASE.2 dynamickou knihovnu do Operačního systému. Hledejte v adresáři **development-support/built-libs** a vyberte nejvhodnější verzi
   - Pro Windows: zkopírujte vybranou verzi **.dll** a přejmenujte ji na **libtase2.dll**. Pak ji přesuňte přímo do ***C://Windows/*** nebo jen do kořenového adresáře projektu.
   - Pro Linux: zkopírujte vybranou verzi **.so** a přejmenujte ji na **libtase2.so**. Pak ji přesuňte do ***/usr/lib/***

2.3 Spuštění aplikace

1. Přesuňte se do adresáře **development-support/docker/** a pusťte příkaz **docker-compose up -d** pro spuštění MongoDB a RabbitMQ
2. Přesuňte se do kořenového adresáře projektu a pusťte příkaz **gradle start**
   a. Gradle start můžete spustit i z IDE
   b. Po prvním gradle build a nebo gradle start je možné spouštět přímo **SubAppRunner** main class z IDE
3. Aplikace by teď měla běžet na

   http://localhost:8080/usy-ocberos-tase2adapterg01/00000000000000000000000000000000-11111111111111111111111111111111

2.4 Inicializace aplikace

Jde o jednorázovou aktivitu, dokud se nesmaže Mongo database.

1. Spusťte Insomniu a importujte workspace ze souboru **usy_ocberos_tase2adapterg01-server/src/test/insomnia/insomnia-workspace.json**. Vyberte **localhost** environment.
2. Pro inicializaci je potřebný token, jeho získaní je v kapitol 2.5.
3. Spusťte request **sys/initApp** v složce **uuAppWorkspace**
4. Spusťte request **sys/initAppWorkspace** v složce **uuAppWorkspace** a po vyžádaní zadejte 2x svoje uuIdentity
5. Jestli asidOwner a awidOwner bude ten samý uživatel, doplňte v Basic Environment ten samý token do proměnných **token** a **awidOwnertoken**. V opačném případe znova vygenerujte Token s přístupovými údaji jiného uživatele a využijte ten.
6. Spusťte request **init** ve složce **usyOcberosTase2adapterg01**
7. Spusťte request **taskScheduler/set** ve složce **usyOcberosTase2adapterg01/Task Scheduler**, jako atribut **lastRun** stačí jakýkoliv datum v minulosti

2.5 Grant Token

1. Spusťte request **grantToken**. Insomnia vyžádá Access Code 1 a Access Code 2. Zadejte svoje údaje.
2. Získaný Token zkopírujte a vložte ho do proměnných **token, awidOwnerToken** a taky **asidOwnerToken** v Basic Environment.
3. Token třeba získávat pravidelně, protože má expirační dobu.

2.6 Tvorba Docker image

1. Přesuňte se do adresáře **usy_ocberos_tase2adapterg01-server/docker**
2. Ujistěte se, zda běží RabbitMQ. V **application-development.properties** upravte jeho adresu – musí jít o explicitní adresu počítače, ne localhost (je totiž využíván při Maven testech).
3. Vytvořte Docker image spuštěním **build.bat**
4. Nastavte hodnoty v souboru docker-compose.yml
5. Spusťte aplikaci příkazem **docker-compose up -d**
6. Inicializujte applikaci a otestujte

2.7 Nová verze knihovny TASE.2

V případe nové verze TASE.2 knihovny od MZ Automation je potřeba několik krokú. Knihovnu dostáváme jako komprimovaný archiv **.tar**.

1. Zkopírujte komprimovaný soubor do adresáře

   usy_ocberos_tase2adapterg01-server\docker\tase2lib a rozbalte ho vedle
2. Uvnitř rozbaleného archivu najděte složku **jna**, jde o Java wrapper pro knihovnu. Projekt otevřete v Eclipse nebo jiném IDE
3. Exportujte ho jako **libtase2.jar** a zkopírujte do adresáře **usy_ocberos_tase2adapterg01-server\libs**.

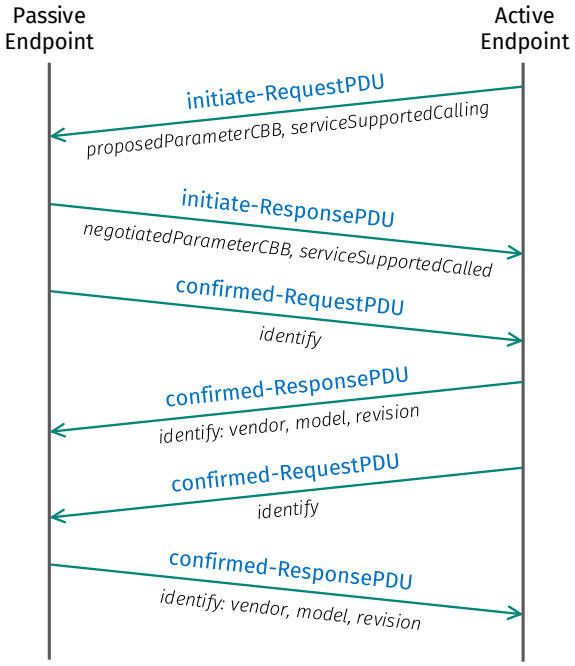
4. Vytvořte dynamickou knihovnu pro Windows (.dll) a pro Linux (.so) ze zdrojových kódu v C
    - Pro Windows
        i. Otevřete Visual Studio 2019
        ii. V menu vyberte Tools -> Command Line -> Developer PowerShell (nebo taky Command Prompt u starších Windows)
        iii. Spusťte následující příkazy, pro vygenerováni C projektu
            - ***cd {extracted_archive_location}/dev-support/libtase2/***
            - ***mdir build***
            - ***cd ./build***
            - ***cmake -G "Visual Studio 16 2019"***
        iv. Přes menu File -> Open -> Project/Solution otevřete soubor ***{extracted_archive_location}/dev-support/libtase2/build/libtase2.sln***
        v. Změňte build configuration na **Release**
        vi. Pravým tlačítkem klikněte na Solution a vyberte Build Solution
        vii. V build adresáři teď najdete soubor **{build_dir}/Release/tase2.dll**, ten skopírujte do **/dev-support/built-libs** a přejmenujte podle OS a verze knihovny
    - Pro Linux
        i. Rozbalte archiv
        ii. Nainstalujte programy **g++**, **make** a **cmake**
        iii. Spusťte následovní příkazy
            - ***mkdir build***
            - ***cd build***
            - ***cmake ..***
            - ***make***
        iv. Po úspěšné kompilaci vzniknou soubory **libtase2.a**, **libtase2.so** a **libtase2.so.x.x**, poslední z nich zkopírujte do **/dev-support/built-libs**
5. Odstraňte celý rozbalený archiv (komprimovaný nechte)
6. V souboru **Dockerfile** v kořenovým adresáři repozitáře, změňte hodnotu proměnné **ARG lib_version=X.Y.Z** na aktuální hodnotu
7. Registrujte novou verzi dynamické knihovny na svém OS, stejně jako v kapitole **2.2 Konfigurace**

3. Popis komunikace protokolu TASE.2

Scénář pro popis komunikace protokolu TASE.2 se sestává z dvou koncových bodů (EndPoints), které obsahují klientskou i serverovou část protokolu. EndPoint 1 je v pasivní roli, čeká tedy na příchozí spojení. Naopak EndPoint 2 představuje aktivní stanici, která se připojuje k EndPoint 1. Demonstrační scénář obsahuje ukázky inicializace komunikace mezi oběma entitami, přenos zpráv skrze Information Message, PULL (metoda Read Point) a PUSH (metoda Transfer Data Set).

3.1 Inicializace TASE.2 komunikace

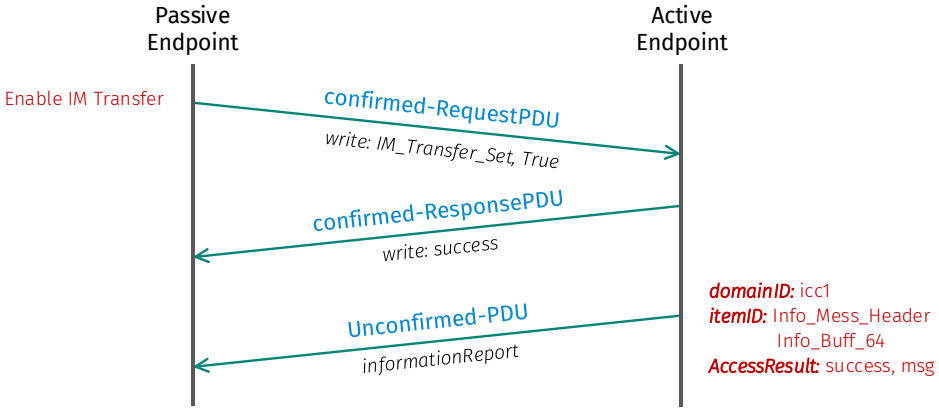
Nejprve je nutné spustit EndPoint 1, který naslouchá na příchozí spojení od EndPoint 2. Celé spojení začíná vysláním inicializačního požadavku na EndPoint 1. Tento požadavek obsahuje informace o navržených CBB (Conformance Building Block) a podporovaných volání služeb (getStatus, read, write, . . .). Na tuto zprávu EndPoint 1 odpoví vyjednanými parametry a podporovanými službami. V poslední fázi už probíhá pouze vzájemná identifikace klientů obsahující informace o výrobci, modelu a revizi. Tato komunikace je znázorněna na obrázkuObrázek 1.



Obrázek 1: Komunikace při připojení zařízení TASE.2

3.2 Komunikace skrze Information Message

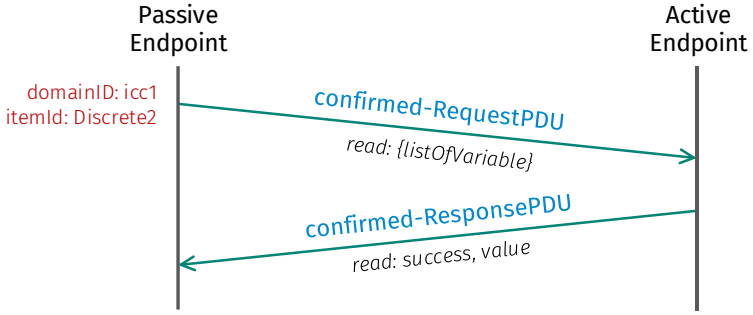
V tomto případě odesílá EndPoint 2 zprávu do EndPoint 1 s využitím služby Information Message. První krok procesu však zahajuje EndPoint 1, který odešle požadavek na povolení IM přenosu na EndPoint 2. Ten poté EndPoint 1 informuje o úspěšné změně nastavení. EndPoint 2 poté může odesílat informační zprávy v libovolný okamžik a EndPoint 1 je úspěšně přijímá. Zprávy jsou odesílány v doméně icc1, samotná zpráva pak obsahuje hlavičku Info_Mess_Header a buffer s vlastní zprávou, hexadecimálně kódovanou. Průběh odeslání Information Message znázorňuje obrázek Obrázek 2.



Obrázek 2: Průběh odeslání informační zprávy TASE.2

3.3 Komunikace typu PULL

V tomto případě vyčítá EndPoint 1 hodnotu z EndPoint 2. EndPoint 1 zahájí komunikaci vytvořením požadavku obsahující seznam žádaných proměnných (v tomto případě Discrete2). EndPoint 2 pak přímo odpoví v následné odpovědi obsahující hodnoty žádaných proměnných. Tato komunikace, znázorněná na obrázku Obrázek 3, odpovídá komunikačnímu model PULL.

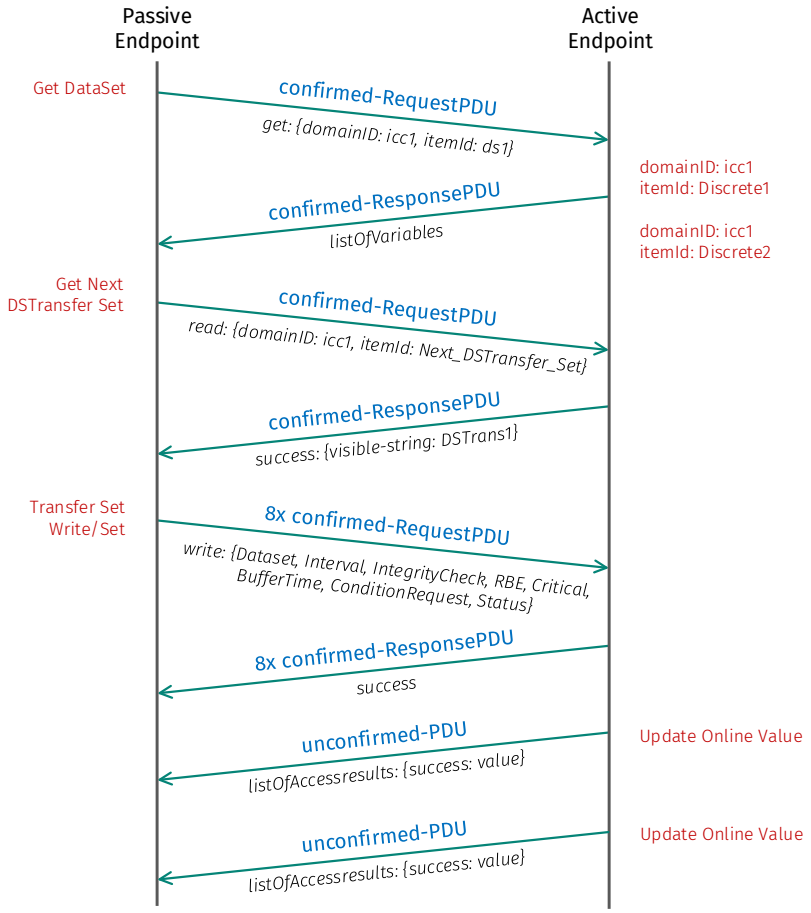


Obrázek 3: Komunikace typu PULL

3.4 Komunikace typu PUSH

V tomto případě EndPoint 1 zahajuje komunikaci odesláním požadavku na získání dostupných proměnných v doméně icc1 v data setu ds1. EndPoint 2 ve své odpovědi odešle seznam dostupných veličin (v tomto případě je využita pouze proměnná s názvem Discrete1). EndPoint 1 si v následující zprávě vyžádá další DSTransfer Set z domény icc1. EndPoint 2 pak ve své odpovědi odešle odpovídající DSTransfer Set (v tomto případě DSTrans1). EndPoint 1 poté odešle 8 požadavků na nastavení parametrů DSTransfer Setu na EndPoint 2 (mezi nimi i četnost vyčítání hodnot). Na všechny tyto zprávy následně EndPoint 2 odpoví zprávou Success.

Proměnná Discrete1 je každé 2 sekundy s inkrementována o hodnotu 1. Aby se tato změna projevila, tak je nutné hodnotu propagovat přes metodu Update Online Value. V tomto okamžiku je nová hodnota odeslána z EndPoint 2 do EndPoint 1. Z EndPoint 1 je úspěšné přijetí zprávy potvrzeno pouze na transportní vrstvě skrze TCP ACK.



Obrázek 4: Komunikace typu PUSH

4. Zprovoznění aplikace IEC101 adaptéru

V následujícím textu bude krok za krokem popsáno, jak rozchodit aplikaci **usy-ocberos-iec101toolg01**.

4.1 Prerekvizity

Pro fungování IEC101 adaptéru je potřeba mít následovní přístupy:

1. Vytvořený Plus4U account
2. *Přístup na GIT repositář*

   ssh://git@codebase.plus4u.net:9422/usy_ocberos_iec101toolg01.git
3. Přístup na Docker repositář

   usylibra/usy_ocberos_iec101toolg01

4.2 Příprava prostředí

Nejprve je potřebné připravit prostředí jak na cílovém počítači (tam kde bude běžet adaptér), tak i na lokálním počítači (ze kterého pracujeme). Může taky jít o ten samý počítač.

1. Nainstalovat Docker na cílový počítač. Způsob instalace se může lišit v závislosti od operačního systému. Instrukce pro instalaci na jednotlivé operační systémy je možné najít zde - https://www.docker.com/products/docker-desktop
2. Zkontrolujte, zda byl na cílovém počítači nainstalován také program Docker Compose, nejjednodušeji příkazem ***docker-compose -version***. Jestli nebyl, řiďte se instrukcemi zde - https://docs.docker.com/compose/install/
3. Na lokálním počítači stáhnete GIT repozitář:

   ssh://git@codebase.plus4u.net:9422/usy_ocberos_iec101toolg01.git

   Z repozitáře zkopírujte následovní soubor na cílový počítač:

   usy_ocberos_iec101toolg01-server/docker/docker-compose.yml.
4. Vejděte do adresáře se zkopírovaným souborem a zkontrolujte ho. Soubor obsahuje konfiguraci jak pro adaptér samotný, tak i pro MongoDB, na kterém je adaptér závislý a taky bude běžet v rámci Dockeru. Důležité jsou zejména hodnoty těchhle parametrů:
   - asid – identifikátor aplikace
   - awid – identifikátor aplikačního workspace
   - asidOwner – identifikátor správce aplikace
   - iec101.connection.commport – sériový port, na kterém adaptér poslouchá

   Za pozornost stojí také hodnota **privileged: true,** která umožní přístup k sériovým portům.
5. Na lokálním počítači nainstalujte program Insomnia pro spouštění HTTP requestů - https://insomnia.rest/
6. Importujte Insomnia workspace. Kliknete na Application -> Preferences -> Data -> Import Data -> From File a vyberte soubor v GIT repozitáři **usy_ocberos_iec101toolg01-server/src/test/insomnia/insomnia-workspace.json**

4.3 Inicializace aplikace

Když už máme nainstalovány potřebný programy a připraveny soubory, můžeme spustit aplikaci.

1. Na cílovém počítači najděte adresář se zkopírovaným souborem **docker-compose.yml** a spusťte příkaz ***docker-compose up -d***. Tím se nastartuje adaptér i DB na pozadí. Příkazem ***docker ps*** je možné zkontrolovat stav běžících containerů a taky jejich ID. Příkazem ***docker logs {container_id}*** je pak možné sledovat jejich logy. Plná dokumentace příkazů na oficiální stránce.

   https://docs.docker.com/engine/reference/commandline/cli/

2. Spusťte Insomnii, otevřete importovaný workspace a vytvořte nový environment, nebo upravte existující. Změňte hodnoty dle konfigurace aplikace (asid, awid, asidOwner, awidOwner)
3. Získejte autentizační token, využitím requestu “Grant Token”. Insomnia vyžádá Access Code 1 a Access Code 2. Je potřeba, aby to byli kódy uživatele, který je jako asidOwner.
4. Získaný Token zkopírujte a vložte ho do proměnné **asidOwnerToken** v Basic Environment.
5. Spusťte request **sys/initApp** v složce **uuAppWorkspace**
6. Spusťte request **sys/initAppWorkspace** v složce **uuAppWorkspace** a po vyžádaní zadejte 2x id uživatele awidOwner
7. Jestli asidOwner a awidOwner bude ten samý uživatel, doplňte v Basic Environment ten samý token do proměnných **token** a **awidOwnertoken**. V opačném případe znova vygenerujte Token s přístupovými údaji jiného uživatele a využijte ten.
8. Spusťte request **init** ve složce **usyOcberosIec101toolg01**
9. Přidejte povolení pro uživatele, který budou v budoucnu využívat REST rozhraní adaptéru (jestli jde o jiné než asidOwner a awidOwner) a to vykonáním requestu **sys/createPermission** ve složce **uuAppWorkspace**. Seznam požadovaných uživatelů zadejte do atributu **uuIdentityList**.

4.4 IEC101 komunikace

Jakmile je aplikace spuštěná, adaptér poslouchá na sériovém portu. Pro přenos správ přes TCP protokol je potřeba dodatečného mechanismu na cílovém počítači, který je mimo rámec této aplikace.

Seznam přijatých dat lze získat pomocí REST dotazu:

1. Spusťte Insomnii a otevřete importovaný workspace
2. Vygenerujte token pro svého uživatele (musí mít ovšem práva na používaní aplikace – role Authorities, nebo AwidOwner) a token zkopírujte do proměnné **token** v Basic Environment
3. Spusťte request **findReportedFrequency** ve složce **usyOcberosIec101toolg01.**

Dotaz vrací data seřazeny sestupně dle času přijetí. Podporuje dva parametry:

- page – číslo stránky
- pageSize– velikost stránky

5. Příprava prostředí pro vývoj IEC101 adaptéru

5.1 Prerekvizity

- Java 8 nebo Java 11
- Docker https://www.docker.com/
- Gradle https://gradle.org/
- Insomnia https://insomnia.rest/
- Idea (nebo jiné IDE)
- Účet v Plus4U (https://www.plus4u.net/)

5.2 Konfigurace

1. Stáhnete project z adresy

   ssh://git@codebase.plus4u.net:9422/usy_ocberos_iec101toolg01.git

2. Vytvořte property soubor **development.properties** a **production.properties** v adresáři

   C:\\Users\{user}\.uu\config

3. Přidejte do obou souborů tyhle atributy s hodnotou dle vašeho účtu

   - asidOwner=
   - yourAccessCode1=
   - yourAccessCode2=

4. Nainstalujte IEC101 Java wrapper knihovnu jako Maven artefakt do lokálního repozitáře spuštěním následovného příkazu:

   ```
   mvn install:install-file -Dfile="./development-support/cplb-java-wrapper/CPLBApiJava.jar" -Dsources="./development-support/cplb-java-wrapper/CPLBApiJava-sources.jar" -DgroupId="usy.ocberos.iec101tool.cplb" -DartifactId="cplbapijava" -Dversion="SNAPSHOT" -Dpackaging="jar"
   ```

5. Registrujte IEC101 dynamickou knihovnu do Operačního systému. Hledejte v adresáři **development-support/built-libs** a vyberte nejvhodnější verzi

   - Pro Windows: zkopírujte vybranou verzi **.dll** a přesuňte ji přímo do ***C://Windows/*** nebo jen do kořenového adresáře projektu.
   - Pro Linux: zkopírujte vybranou verzi **.so** a přesuňte ji do ***/usr/lib/***

6. V případe vývoje na Windows je nutné nainstalovat a spustit program http://com0com.sourceforge.net/ pro simulaci sériových portů

7. V **development.properties** změňte parametr ***iec101.connection.commport*** na hodnotu dle simulovaného portu

5.3 Spuštění aplikace

1. Přesuňte se do adresáře **development-support/docker/** a pusťte příkaz **docker-compose up -d** pro spuštění MongoDB
2. Přesuňte se do kořenového adresáře projektu a pusťte příkaz **gradle start**
   a. Gradle start můžete spustit i z IDE

b. Po prvním gradle build a nebo gradle start je možné spouštět přímo **SubAppRunner** main class z IDE

3. Aplikace by teď měla běžet na

   http://localhost:8080/usy-ocberos-iec101toolg01/00000000000000000000000000000000-11111111111111111111111111111111

5.4 Inicializace aplikace

Jde o jednorázovou aktivitu, dokud se nesmaže Mongo database.

1. Spusťte Insomniu a importujte workspace ze souboru **usy_ocberos_iec101toolg01-server/src/test/insomnia/insomnia-workspace.json**. Vyberte **localhost** environment.
2. Pro inicializaci je potřebný token, jeho získaní je v kapitol 2.5.
3. Spusťte request **sys/initApp** v složce **uuAppWorkspace**
4. Spusťte request **sys/initAppWorkspace** v složce **uuAppWorkspace** a po vyžádaní zadejte 2x svoje uuIdentity
5. Jestli asidOwner a awidOwner bude ten samý uživatel, doplňte v Basic Environment ten samý token do proměnných **token** a **awidOwnertoken**. V opačném případě znova vygenerujte Token s přístupovými údaji jiného uživatele a využijte ten.
6. Spusťte request **init** ve složce **usyOcberosTase2adapterg01**

5.5 Grant Token

1. Spusťte request **grantToken**. Insomnia vyžádá Access Code 1 a Access Code 2. Zadejte svoje údaje.
2. Získaný Token zkopírujte a vložte ho do proměnných **token, awidOwnerToken** a taky **asidOwnerToken** v Basic Environment.
3. Token třeba získávat pravidelně, protože má expirační dobu.

5.6 Tvorba Docker image

1. Přesuňte se do adresáře **usy_ocberos_tase2adapterg01-server**
2. Vytvořte .war pomocí příkazu ***gradle build***
3. Přesuňte se do adresáře **usy_ocberos_tase2adapterg01-server/docker**
4. Vytvořte Docker image spuštěním **build.bat**
5. Nastavte hodnoty v souboru docker-compose.yml
6. Spusťte aplikaci příkazem **docker-compose up -d**
7. Inicializujte aplikaci a otestujte

5.7 Nová verze knihovny IEC101

V případe nové verze IEC101 knihovny je potřeba několik kroků. Knihovna se skládá ze dvou částí:

a. Zkompilovaná knihovna v jazyku C ve formě **.so** nebo **.dll**, jak pro x32 tak pro x64 verzi (CPLBShlibSwKey.zip)
b. Java wrapper obsahující zkompilovaný **.jar** archiv a taky zdrojové kódy

Vykonejte následující kroky:

1. Extrahujte někde **CPLBShlibSwKey.zip** a **CPLBApiJavaSwKey.exe**
2. Zkopírujte zkompilované **.so** a **.dll** soubory z adresáře **{extraction_location}/CPLBShlibSwKey** do adresáře **./development-support/built-libs**
3. Přesuňte se do adresáře **{extraction_location}/CPLBApiJavaSwKey** a spusťte příkaz ***jar cvf CPLBApiJava-sources.jar de*** pro zabalení zdrojového kódu do **jar** archivu. (měl by se vygenerovat soubor **CPLBApiJava-sources.jar**)
4. Zkopírujte soubory **CPLBApiJava.jar** a **CPLB-sources.jar** do **./development-support/cplb-java-wrapper**
5. Zopakujte krok 3 a 4 z kapitoly 5.3 Konfigurace (instalace a registrace nové verze knihovny pro aplikaci)