

Integračná špecifikácia

pre klientov   
Reporting API a Operator API

Implementácia vyhlášky c.445/2020

verzia 5.0

|  |  |
| --- | --- |
| **Označenie** | Návrh riešenia - Implementácia vyhlášky c.445/2020  Integračná špecifikácia pre klientov |
| **Vlastník** | Asseco Central Europe, a.s. |
| **Klasifikácia informácií** | CONFIDENTIAL |

# História zmien

|  |  |  |  |
| --- | --- | --- | --- |
| **Verzia** | **Dátum vydania** | **Spracovateľ** | **Popis zmien** |
| 1.0 | 01.06.2021 | ACE | Prvá verzia |
| 2.0 | 04.06.2021 | cbData | Aktualizácia |
| 3.0 | 06.06.2021 | cbData | Aktualizácia |
| 4.0 | 10.06.2021 | cbData | Aktualizácia |
| 5.0 | 14.06.2021 | cbData | Aktualizácia |

# Distribúcia

|  |  |
| --- | --- |
| **P. č.** | **Meno a priezvisko / Miesto distribúcie** |
| 1 | Príslušní zamestnanci Úradu pre reguláciu hazardných hier/nakopírovanie na FTP server |
|  |  |

Obsah

[História zmien 2](#_Toc74654197)

[Distribúcia 2](#_Toc74654198)

[1 Úvod 7](#_Toc74654199)

[2 Skratky a pojmy 7](#_Toc74654200)

[2.1 Použité skratky, pojmy, medzinárodné normy a štandardy 7](#_Toc74654201)

[2.2 Github repositáře 7](#_Toc74654202)

[3 Architektúra 8](#_Toc74654203)

[3.1 Úvod 8](#_Toc74654204)

[3.2 Principy 9](#_Toc74654205)

[3.3 Autentifikácia a autorizácia 10](#_Toc74654206)

[3.4 Šifrovanie 10](#_Toc74654207)

[3.5 Pečatenie 10](#_Toc74654208)

[3.6 Certifikáty a ich použitie 10](#_Toc74654209)

[3.6.1 Registrace certifikátů 12](#_Toc74654210)

[3.7 Komponenty 14](#_Toc74654211)

[3.7.1 Server rAPI 15](#_Toc74654212)

[3.7.2 Klient oAPI 15](#_Toc74654213)

[4 Datový balíček 16](#_Toc74654214)

[4.1 ReportPackage 16](#_Toc74654215)

[4.2 ResponsePackage 17](#_Toc74654216)

[4.3 Zabezpečení balíčku (požadavku a odpovědi) 18](#_Toc74654217)

[4.4 Validace 19](#_Toc74654218)

[5 Aplikačné rozhranie pre reporting - Reporting API 20](#_Toc74654219)

[5.1 Table of Contents 20](#_Toc74654220)

[5.1.1 Diagnostics 20](#_Toc74654221)

[5.1.2 Enumerators 20](#_Toc74654222)

[5.1.3 Health 20](#_Toc74654223)

[5.1.4 Reporting 20](#_Toc74654224)

[5.2 Diagnostics 20](#_Toc74654225)

[5.2.1 post /rapi/diagnostics/v1/diagnostics/isAlive 20](#_Toc74654226)

[5.2.2 post /rapi/diagnostics/v1/diagnostics/version 22](#_Toc74654227)

[5.3 Enumerators 23](#_Toc74654228)

[5.3.1 get /rapi/enumerators/v1/classificationCodes/list 23](#_Toc74654229)

[5.3.2 get /rapi/enumerators/v1/currencyCodes/list 25](#_Toc74654230)

[5.3.3 get /rapi/enumerators/v1/currencyCodes/translate 26](#_Toc74654231)

[5.3.4 get /rapi/enumerators/v1/errorCodes/list 27](#_Toc74654232)

[5.3.5 get /rapi/enumerators/v1/errorCodes/translate 28](#_Toc74654233)

[5.3.6 get /rapi/enumerators/v1/packageKinds/list 29](#_Toc74654234)

[5.3.7 get /rapi/enumerators/v1/packageStates/list 30](#_Toc74654235)

[5.3.8 get /rapi/enumerators/v1/processStates/list 32](#_Toc74654236)

[5.3.9 get /rapi/enumerators/v1/systemStates/list 33](#_Toc74654237)

[5.4 Health 34](#_Toc74654238)

[5.4.1 get /health 34](#_Toc74654239)

[5.5 Reporting 35](#_Toc74654240)

[5.5.1 post /rapi/reporting/v1/package/getStatus 35](#_Toc74654241)

[5.5.2 post /rapi/reporting/v1/package/list 37](#_Toc74654242)

[5.5.3 post /rapi/reporting/v1/package/rollback 38](#_Toc74654243)

[5.5.4 post /rapi/reporting/v1/package/load 40](#_Toc74654244)

[5.6 Models 41](#_Toc74654245)

[5.6.1 Table of Contents 42](#_Toc74654246)

[5.6.2 ClassificationCode 42](#_Toc74654247)

[5.6.3 CurrencyCode 42](#_Toc74654248)

[5.6.4 EmptyRequest 42](#_Toc74654249)

[5.6.5 ErrorCode 42](#_Toc74654250)

[5.6.6 Fault 42](#_Toc74654251)

[5.6.7 FaultCode 42](#_Toc74654252)

[5.6.8 FaultResponse 42](#_Toc74654253)

[5.6.9 GetStatusResponse 42](#_Toc74654254)

[5.6.10 HealthReport 43](#_Toc74654255)

[5.6.11 HealthReportEntry 43](#_Toc74654256)

[5.6.12 HealthStatus 43](#_Toc74654257)

[5.6.13 IsAliveRequest 43](#_Toc74654258)

[5.6.14 IsAliveResponse 44](#_Toc74654259)

[5.6.15 ListRequest 44](#_Toc74654260)

[5.6.16 ListResponse 44](#_Toc74654261)

[5.6.17 OperatorPackage 44](#_Toc74654262)

[5.6.18 PackageKind 45](#_Toc74654263)

[5.6.19 PackageState 45](#_Toc74654264)

[5.6.20 PackageloadRequest 45](#_Toc74654265)

[5.6.21 ProcessState 45](#_Toc74654266)

[5.6.22 RequestIdRequest 45](#_Toc74654267)

[5.6.23 StatusCode 46](#_Toc74654268)

[5.6.24 StatusDetail 46](#_Toc74654269)

[5.6.25 StatusResponse 46](#_Toc74654270)

[5.6.26 SystemState 46](#_Toc74654271)

[5.6.27 VersionResponse 46](#_Toc74654272)

[6 Aplikačné rozhranie pre reporting – Operator API 46](#_Toc74654273)

[6.1 Table of Contents 47](#_Toc74654274)

[6.1.1 Notifier 47](#_Toc74654275)

[6.2 Notifier 47](#_Toc74654276)

[6.2.1 post /oapi/reporting/v1/notifier/setStatus 47](#_Toc74654277)

[6.2.2 Table of Contents 48](#_Toc74654278)

[6.2.3 ClassificationCode 48](#_Toc74654279)

[6.2.4 EmptyResponse 48](#_Toc74654280)

[6.2.5 ErrorCode 48](#_Toc74654281)

[6.2.6 Fault 48](#_Toc74654282)

[6.2.7 FaultCode 48](#_Toc74654283)

[6.2.8 FaultResponse 48](#_Toc74654284)

[6.2.9 NotifierSetStatusRequest 49](#_Toc74654285)

[6.2.10 PackageState 49](#_Toc74654286)

[6.2.11 ProcessState 49](#_Toc74654287)

[6.2.12 StatusDetail 49](#_Toc74654288)

[6.3 Prípady komunikácie 50](#_Toc74654289)

[6.3.1 Kód: SEQ\_1 50](#_Toc74654290)

[6.3.2 Kód: SEQ\_2 50](#_Toc74654291)

[6.3.3 Kód: SEQ\_3 51](#_Toc74654292)

[6.3.4 Kód: SEQ\_4 52](#_Toc74654293)

[7 Referenční klient komunikačního rozhraní dle vyhlášky č. 445/2020 – Nenahrazuje QSCD 53](#_Toc74654294)

[7.1 Úvod 53](#_Toc74654295)

[7.2 Popis 54](#_Toc74654296)

[7.3 Komprimace 54](#_Toc74654297)

[7.4 Dekomprimace 55](#_Toc74654298)

[7.5 Šifrování 55](#_Toc74654299)

[7.6 Dešifrování 56](#_Toc74654300)

[7.7 Vytvoření elektronické pečeti 56](#_Toc74654301)

[7.8 Ověření elektronické pečeti 56](#_Toc74654302)

# Úvod

Cílem dokumentu je technický popis integrace pro klientské systémy. Dokument rozšiřuje konkrétní kapitoly vyhlášky č. 445/2020 Z. z. ve smyslu zpřesnění technické výkladu příslušné kapitoly.

# Skratky a pojmy

## Použité skratky, pojmy, medzinárodné normy a štandardy

|  |  |
| --- | --- |
| **Skratka** | **Popis** |
| REST API | Representational State Transfer API <https://en.wikipedia.org/wiki/Representational_state_transfer> |
| Swagger | Swagger je open source framework pro návrh, tvorbu, dokumentaci a konzumaci RESTful web API <https://cs.wikipedia.org/wiki/Swagger_(software)> |
| Let's Encrypt | Let's Encrypt je certifikační autorita. Za pomoci automatizovaného procesu, navrženého tak, aby odstranil složitý proces manuální tvorby, ověřování, podepisování, instalace a obnovování certifikátu. Zdarma poskytuje doménově ověřené certifikáty (DV, anglicky Domain Validated) typu X.509 pro šifrování protokolu TLS. <https://letsencrypt.org/> <https://cs.wikipedia.org/wiki/Let%27s_Encrypt>  <https://en.wikipedia.org/wiki/Let%27s_Encrypt> |
| CAdES | CMS Advanced Electronic Signature. <https://en.wikipedia.org/wiki/CAdES_(computing)> <https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eSignature+standards#eSignaturestandards-CAdES(CMSAdvancedElectronicSignature)BaselineProfile> |
| eIDAS | Nařízení Evropské unie č. 910/2014 o elektronické identifikaci a důvěryhodných službách pro elektronické transakce na vnitřním evropském trhu.  <https://en.wikipedia.org/wiki/EIDAS> |
|  |  |
|  |  |
|  |  |
|  |  |

## Github repositáře

|  |  |
| --- | --- |
| **Pojem** | **Popis** |
| Repositář EU | Digital Signature Service : creation, extension and validation of advanced electronic signatures <https://github.com/esig/dss> <https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/DSS> <https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eSignature+standards> |
| Repositář MF ČR | Referenční klient komunikačního rozhraní dle vyhlášky č. 10/2019 <https://github.com/Ministerstvo-financi/hazard-komunikacni-rozhrani> |
| Repositář URHH | Podklady pro vývoj klienta rozhraní dle vyhlášky č. 445/2020  <https://github.com/cbData/sk_reporting-work> |

# Architektúra

## Úvod

Základním principem níže popsaného řešení je mechanismus, na jehož základě probíhá výměna zpráv mezi systémy státní správy a provozovatelem.



Předmětem výměny informací jsou dva principy:

* datové balíčky
* request/response

Vzájemná výměna dat mezi systémy je realizována prostřednictvím https komunikace, která jako aplikační síťová vrstva slouží k přenosu obsahu ve formě datových balíčků ve formátu XML.





Na předmět výmeny (datové balíčky) jsou kladeny požadavky na jejich zabezpečení a jednoznačnou identifikaci odesílatele. Identifikace odesílatele je řešena v souladu s nařízením Evropské unie č. 910/2014 o elektronické identifikaci a důvěryhodných službách pro elektronické transakce na vnitřním evropském trhu a následně vyhláškou č. 445/2020 Z. z. Ministerstva financií Slovenskej republiky. Vyhláška stanovuje využívání kvalifikovaných prostředků elektronického podpisu nebo pečetě prostředky QSCD.

## Principy

* Serverová strana URHH vystavuje REST API zvané rAPI
* Serverová strana URHH zveřejňuje následující přístupové parametry
  + url adresu, kde je dostupné sdílené prostředí rAPI pro všechny klienty (provozovatele)
  + veřejnou část šifrovacího certifikátu
  + veřejnou část pečetícího certifikátu
  + veřejnou část klientského certifikátu + CA
* Klientská strana provozovatele vystavuje REST API zvané oAPI
* Klientská strana provozovatele zveřejňuje následující přístupové parametry
  + url adresu, kde je dostupné oAPI pro server URHH
  + veřejnou část šifrovacího certifikátu
  + veřejnou část pečetícího certifikátu
  + veřejnou část klientského certifikátu + CA
* Komunikace je iniciována na straně klienta a volá příslušné metody rAPI dle specifikace za účelem odeslání datového balíčku
* Výsledky zpracování požadavku na straně rAPI jsou propagovány do klientského oAPI

## Autentifikácia a autorizácia

Authentizace je realizována prostřednictvím klientského certifikátu, jehož veřejnou část deklarativně zveřejňuje vždy příslušný klient. Provozovatel sveřejňuje veřejnou část přístupového certifikátu. Stejně tak serverový systém ÚRHH zveřejňuje svou veřejnou část certifikátu straně provozovatele.

Authorizace je řešena na základě přístupu k REST API s užitím klientských certifikátů.

## Šifrovanie

S ohledem na požadavky ochrany dat jsou herní a finanční data šifrována veřejnými klíči komerčních certifikátů provozovatele a serveru státní správy. Následně je možné na straně příjemce data dešifrovat. Při šifrování se zašifrovaný soubor uloží ve formátu Cryptographic message syntax (CMS) s využitím typu obsahu Enveloped-Data podle standardu RFC 5652.

Data jsou šifrována dvěmi veřejnými klíčí

* ŠIFROVACÍ CERTIFIKÁT klienta (provozovatele)
* ŠIFROVACÍ CERTIFIKÁT serveru (URHH)

Data jsou dešifrována vždy příslušným privátním klíčem protistrany.

Na základě šifrování obsahu veřejným klíčem systému státní správy dojde k dešifrování dat privátním klíčem umístěným na systému příjemce zprávy.

Dešifrování je reverzní proces k procesu šifrování. Data jsou dešifrována vždy příslušným vlastním privátním klíčem.

## Pečatenie

Jak je uvedeno, vyhláška č. 445/2020 Z. z. vyžaduje kvalizikovaný proces pečetění prostředky QSCD. Za tímto účelem je nutné zajistit tento požadavek prostřednictvím certifikovaných služeb autorizovaných subjetů.

## Certifikáty a ich použitie

Klientská i serverová strana využívají pro svou komunikaci sadu certifikátů. Každá strana je zodpovědná za jejich získání a následnou registraci prostředky systému URHH.

**Operator (provozovatel)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Název** | **Typ** | **Cíl** | **Popis** |
| CERTIFIKÁT PRO AUTENTIZACI KLIENTA | Veřejný klíč  .pem | Poskytuje pro URHH za účelem vlastní identifikace přístupu k rAPI | Slouží k identifikaci klienta při přístupu na serverovému rozhraní rAPI.  Postup pro generování uveden v repositáři. |
| SEZNAM CERTIFIKÁTŮ CERTIFIKAČNÍCH AUTORIT PRO AUTENTIZACI KLIENTA | Veřejný klíč  .pem | Poskytuje pro URHH za účelem vlastní identifikace přístupu k rAPI a validaci certifikátu | Slouží k identifikaci klienta při přístupu na serverovému rozhraní rAPI.  Postup pro generování uveden v repositáři.  CA není nutné dodávat za předpokladu, že se jedná o veřejnou CA |
| ŠIFROVACÍ CERTIFIKÁT | Veřejný klíč  .pem | Poskytuje pro URHH za účelem šifrování odpovědi ze serveru na stranu klienta | Technologický (komerční serverový) certifikát <https://www.ica.cz/technologicke-certifikaty> <https://www.postsignum.cz/komercni_certifikaty.html> |
| Privátní klíč  .p12 | Ukládá na straně klienta pro potřeby dešifrování zprávy zaslané serverem pro klienta |
| CERTIFIKÁT PRO PEČETĚNÍ | Veřejný klíč  .pem | Poskytuje pro URHH za účelem ověření pečetě zprávy zaslané klientem na server | Kvalifikovaný certifikát pro elektronický podpis SR <https://www.ica.cz/kvalifikovany-certifikat-pro-ePodpis-SR> <https://www.elektronickypodpis.cz/kvalifikovany-elektronicky-podpis/>  <https://www.postsignum.cz/kvalifikovane_certifikaty.html> |
| Privátní klíč  .p12 | Ukládá na straně klienta pro potřeby pečetění zprávy zaslané klientem na server |

**Government (URHH)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Název** | **Typ** | **Cíl** | **Popis** |
| CERTIFIKÁT PRO AUTENTIZACI KLIENTA | Veřejný klíč  .pem | Poskytuje pro klienty za účelem identifikace přístupu URHH k oAPI | Slouží k identifikaci serveru při přístupu ke klientskému rozhraní oAPI.  Postup pro generování uveden v repositáři. |
| SEZNAM CERTIFIKÁTŮ CERTIFIKAČNÍCH AUTORIT PRO AUTENTIZACI KLIENTA | Veřejný klíč  .pem | Poskytuje pro klienty za účelem identifikace přístupu URHH k oAPI a validaci certifikátu | Slouží k identifikaci serveru při přístupu ke klientskému rozhraní oAPI.  Postup pro generování uveden v repositáři.  CA není nutné dodávat za předpokladu, že se jedná o veřejnou CA |
| ŠIFROVACÍ CERTIFIKÁT | Veřejný klíč  .pem | Poskytuje pro klienty za účelem šifrování balíčku zasílaného na server ze strany klienta | Technologický (komerční serverový) certifikát <https://www.ica.cz/technologicke-certifikaty> <https://www.postsignum.cz/komercni_certifikaty.html> |
| Privátní klíč  .p12 | Ukládá na straně serveru pro potřeby dešifrování zprávy zaslané klientem na server |
| CERTIFIKÁT PRO PEČETĚNÍ | Veřejný klíč  .pem | Poskytuje pro klienty za účelem ověření pečetě zprávy zaslané serverem klientu | Kvalifikovaný certifikát pro elektronický podpis SR <https://www.ica.cz/kvalifikovany-certifikat-pro-ePodpis-SR> <https://www.elektronickypodpis.cz/kvalifikovany-elektronicky-podpis/>  <https://www.postsignum.cz/kvalifikovane_certifikaty.html> |
| Privátní klíč  .p12 | Ukládá na straně serveru pro potřeby pečetění zprávy zaslané serverem klientu |

### Registrace certifikátů

Registrace certifikátů provozovatele probíhá prostředky elektronického fromuláře vydaného úřadek URHH v systému ÚPVS (Ústredný portál verejnej správy SR).

Oznámenie o registrácii prevádzkovateľa hazardnej hry na prenos údajov o hazardných hrách

podľa § 4 vyhlášky č. 445/2020 Z. z.

Prevádzkovateľ

|  |  |  |  |
| --- | --- | --- | --- |
| Obchodné meno/Názov   |  | | --- | |  |  |  | | --- | | Identifikačné číslo organizácie | |  | |

Druh oznámenia

|  |
| --- |
| Registrácia alebo zmena registrácie Ukončenie registrácie |

Technické údaje

|  |  |  |  |
| --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | |  | | --- | | Textová forma certifikátu (Base-64 encoded X.509) | |  |   Priložte súbor certifikátu vo formáte DER encoded binary X.509 |   Kvalifikovaný certifikát pre overenie pečate dátového balíčka |
| |  |  |  | | --- | --- | --- | | |  | | --- | | Textová forma certifikátu (Base-64 encoded X.509) | |  |   Priložte súbor certifikátu vo formáte DER encoded binary X.509 |   Certifikát pre autentifikáciu klienta pre prístup na aplikačné rozhranie úradu |
| |  |  |  | | --- | --- | --- | | |  | | --- | | Textová forma certifikátu (Base-64 encoded X.509) | |  |   Priložte súbor certifikátu vo formáte DER encoded binary X.509 |   Certifikát certifikačnej autority (CA) pre autentifikáciu klienta |
|  |
| |  |  |  | | --- | --- | --- | | |  | | --- | | Textová forma certifikátu (Base-64 encoded X.509) | |  |   Priložte súbor certifikátu vo formáte DER encoded binary X.509 |   Certifikát pre šifrovanie |
|  |
| |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | |  | | --- | |  | | URL adresa aplikačného rozhrania prevádzkovateľa pre odoslanie potvrdenia o prijatí dátového balíčka | |  | |  | | IP adresa servera prevádzkovateľa, z ktorej bude povolený prístup na server úradu pre volanie aplikačného rozhrania úradu | |  | |  | |   Adresy |
|  |

|  |
| --- |
|  |

Dátum účinnosti od:

|  |
| --- |
|  |

Meno a priezvisko osoby oprávnenej konať za prevádzkovateľa hazardnej hry:

|  |
| --- |
|  |

Meno a priezvisko kontaktnej osoby za prevádzkovateľa hazardnej hry:

|  |  |  |
| --- | --- | --- |
| Telefónne číslo kontaktnej osoby |  | E-mailová adresa kontaktnej osoby |
|  |  |  |

## Komponenty

Součástí navržené architektury jsou následující komponenty:



**Operator (provozovatel)**

|  |  |
| --- | --- |
| **Název** | **Popis** |
| Gaming systems | Herní systémy a další subsystém podléhající reportovací povinnosti |
| Client | Systém na straně provozovatele s vlastní aplikační logikou a hostující komunikační rozhraní REST API – oAPI |
| oAPI | Implementace komunikačního rozhraní REST API – oAPI jako vstupní bod do systému provozovatele ze strany systému státní správy |
| Report Package | Datová zpráva, která je komunikována mezi systémy. V tomto případě nesoucí balíček s daty, které jsou předmětem reportovací povinnosti |

**Government (URHH)**

|  |  |
| --- | --- |
| **Název** | **Popis** |
| Administration web server | Administrativní portál systému státní správy poskytující funkcionalitu nutnou pro vzájemnou výměnu technických parametrů pro realizaci zabezpečené komunikace mezi systémy a dalšími artefakty řešení |
| Server | Systém na straně státní správy s vlastní aplikační logikou a hostující komunikační rozhraní REST API – rAPI |
| rAPI | Implementace komunikačního rozhraní REST API – rAPI jako vstupní bod do systému provozovatele ze strany systému státní správy |
| Response Package | Datová zpráva (odpověď), která je komunikována mezi systémy. V tomto případě nesoucí balíček s odpovědí nebo informací o stavu zpracování datového balíčku na straně serveru, které jsou předmětem reportovací povinnosti jako součást JSON struktury |

### Server rAPI

Na serverové straně je implementováno komunikační rozhraní API. Vzhledem k tomu, že se jedná o centrální reportovací API, je pojmenováno rAPI (reportingAPI).

**

Referenční prostředí je dostupné na adrese:

<https://app.swaggerhub.com/apis/URHH/reporting-api/v1>

Lze využít k automatickému generování klienta prostředky Swagger.

### Klient oAPI

Na klientské straně je implementováno komunikační rozhraní API. Vzhledem k tomu, že se jedná o klientské reportovací API příslušného provozovatele, je pojmenováno oAPI (operatorAPI).

**

Referenční prostředí je dostupné na adrese:

<https://app.swaggerhub.com/apis/URHH/operator-api/v1>

# Datový balíček

Předmětem výměny dat jsou kromě základních komunikačních možností aplikační síťové vrstvy http a transportní síťové vrstvy REST API samotné datové balíčky, které jsou předmětem reportovací povinnosti. Tako kapitola popisuje balíčky z technického hlediska.

**

## ReportPackage

Balíček obsahující provozní data provozovatele v souladu s platným ustanovením ve věci povinnosti vykazovat herní a finanční data. Na základě níže popsaného procesu dojde k ochraně dat ve smyslu komprimace, šifrování a pečetění. Takto sestavený balíček je prostřednictvím metody rAPI „upload“ transportován na server:

[https://app.swaggerhub.com/apis/URHH/reporting-api/v1#/Reporting/post\_rapi\_reporting\_v1\_package\_upload](https://app.swaggerhub.com/apis/URHH/reporting-api/v1" \l "/Reporting/post_rapi_reporting_v1_package_upload)

Přenos probíhá dvěma způsoby

* Balíček je integrální součástí JSON requestu  
  Server přijme balíček, zařadí do interní fronty pro zpracování. Při jeho aktivaci stahuje balíček na server a iniciuje jeho zpracování.

*Př.:*

*{*

*"requestId": "8f430525-c9c5-4caf-81ef-728b20e861c2",*

*"operatorCode": "12345678",*

*"packageVersion": "01",*

*"packageKind": "CharitativnaLoteria",*

*"packageTimeFrameStartInclusive": "2021-05-15T00:00:00+02:00",*

*"packageTimeFrameEndExclusive": "2021-05-16T00:00:00+02:00",*

***"packageContent": ""MIIERgYJKoZIhvcNAQcDoIIENz…UbqbpLwP6ajoxnyoT9p",***

*"packageUri": "",*

*"packageStamp": "2021-05-16T15:21:37.3778566+02:00"*

*}*

* Balíček je předán odkazem na externí úložiště klienta (preferovaná)

Server přidělí zpracování balíčku interní časový slot a při jeho aktivaci stahuje balíček na server a iniciuje jeho zpracování.

*Př.:*

*{*

*"requestId": "8f430525-c9c5-4caf-81ef-728b20e861c2",*

*"operatorCode": "12345678",*

*"packageVersion": "01",*

*"packageKind": "CharitativnaLoteria",*

*"packageTimeFrameStartInclusive": "2021-05-15T00:00:00+02:00",*

*"packageTimeFrameEndExclusive": "2021-05-16T00:00:00+02:00",*

*"packageContent": "",*

***"packageUri": "https://example.com/8f430525-c9c5-4caf-81ef-728b20e861c2.zip.p7e.p7s",***

*"packageStamp": "2021-05-16T15:21:37.3778566+02:00"*

*}*

## ResponsePackage

Balíček obsahující data, která jsou směrována ze strany serveru státní správy k provozovateli. Response je JSON struktura komunikovaná ze strany serveru na stranu klienta.

Obdobným způsobem jako je případ zasílání datových balíčků na server je řešena distribuce informací o průběhu zpracování balíčku ze serveru na klienta. Komunikace probíhá prostřednictvím klientského komunikačního rozhraní oAPI. Za tímto účelem se využívá metoda setStatus.

<https://app.swaggerhub.com/apis/URHH/operator-api/v1#/Notifier/post_oapi_reporting_v1_notifier_setStatus>

JSON struktura odpovědi obsahuje detailní popis výsledku zpracování balíčku na serverové straně. Integrální struktury je položka content, která obsahuje komprimovanou, šifrovanou a podepsanou kopii zprávy pro případ nutnosti jejího ověření.

*{*

*"requestId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",*

*"processingState": "Planned",*

*"packageState": "Processing",*

*"details": [*

*{*

*"file": "string",*

*"line": 0,*

*"classificationCode": "Error",*

*"errorCode": "InvalidField",*

*"detail": "string",*

*"result": "string"*

*}*

*],*

***"content": "string"***

*}*

*Příklad:*

*{"requestId":"388faf53-148c-4412-b341-2c0b497a5b5c","processingState":"Planned","packageState":"InvalidContent","details":[{"line":0,"classificationCode":"Error","errorCode":"InvalidField"}],"content":""}*

## Zabezpečení balíčku (požadavku a odpovědi)

Na všech datových balíčcích, které jsou transportovány mezi klienty a servery je realizován proces ochrany a jednoznačné identifikace autora obsahu

* Komprimace (Compression)
* Šifrování (Crypting)
* Ochrana identitou odesílatele (Sender identity protection)



Po přijetí balíčku protistranou je realizovány reverzní proces rovněž ve třech krocích

* Ověření identity odesílatele (Verify the sender's identity)
* Dešifrování (Decrypting)
* Dekomprimace (Decompression)

Výsledkem je předání a přijetí datového balíčku.



## Validace

Validační proces datových balíčků je rozdělen do dvou oblastní

* Formální (Syntaktické) validace
* Databázové validace

# Aplikačné rozhranie pre reporting - Reporting API

Json description

More information: https://helloreverb.com

Contact Info: hello@helloreverb.com

Version: v1

All rights reserved

<http://apache.org/licenses/LICENSE-2.0.html>

## Table of Contents

### Diagnostics

* post /rapi/diagnostics/v1/diagnostics/isAlive
* post /rapi/diagnostics/v1/diagnostics/version

### Enumerators

* get /rapi/enumerators/v1/classificationCodes/list
* get /rapi/enumerators/v1/currencyCodes/list
* get /rapi/enumerators/v1/currencyCodes/translate
* get /rapi/enumerators/v1/errorCodes/list
* get /rapi/enumerators/v1/errorCodes/translate
* get /rapi/enumerators/v1/packageKinds/list
* get /rapi/enumerators/v1/packageStates/list
* get /rapi/enumerators/v1/processStates/list
* get /rapi/enumerators/v1/systemStates/list

### Health

* get /health

### Reporting

* post /rapi/reporting/v1/package/getStatus
* post /rapi/reporting/v1/package/list
* post /rapi/reporting/v1/package/rollback
* post /rapi/reporting/v1/package/load

## Diagnostics

### post /rapi/diagnostics/v1/diagnostics/isAlive

Metóda k opakovanému získavaniu stavu aplikačného rozhrania. Nahrazena metódou /health. (**rapiDiagnosticsV1DiagnosticsIsAlivePost**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Consumes

This API call consumes the following media types via the Content-Type request header:

* application/json

Request body

**body** IsAliveRequest **(optional)**

*Body Parameter* —

*example: { "systemState" : "Ok", "nextClientShutdown" : "2021-06-04T16:03:01.0900127+02:00", "isAlive" : true }*

Return type

IsAliveResponse

Example data

Content-Type: application/json

{

"isAlive" : true,

"systemState" : "Ok",

"nextServerShutdown" : "2000-01-23T04:56:07.000+00:00"

}

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

IsAliveResponse

Example data

Content-Type: application/json

{"systemState":"Ok","nextServerShutdown":"2021-06-05T16:03:01.0933349+02:00","isAlive":true}

400

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidContract","data":[]}}

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### post /rapi/diagnostics/v1/diagnostics/version

Metóda k získaniu verzie aplikačného rozhrania. Nahrazena metódou /health. (**rapiDiagnosticsV1DiagnosticsVersionPost**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Consumes

This API call consumes the following media types via the Content-Type request header:

* application/json

Request body

**body** EmptyRequest **(optional)**

*Body Parameter* —

Return type

VersionResponse

Example data

Content-Type: application/json

{

"lastdate" : "2000-01-23T04:56:07.000+00:00",

"version" : "version"

}

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

VersionResponse

Example data

Content-Type: application/json

{"version":"1.0.0.0","lastdate":"2021-06-04T16:03:01.0999361+02:00"}

400

BadRequest - ErrInvalidContract

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

## Enumerators

### get /rapi/enumerators/v1/classificationCodes/list

Získanie číselníka klasifikácie detailu správ pri spracovaní (**rapiEnumeratorsV1ClassificationCodesListGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Return type

array[ClassificationCode]

Example data

Content-Type: application/json

[ "Error", "Error" ]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

Example data

Content-Type: application/json

["Error","Warning","Information"]

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### get /rapi/enumerators/v1/currencyCodes/list

Získanie číselníka mien podľa medzinárodnej normy ISO 4217 Codes for the representation of currencies (**rapiEnumeratorsV1CurrencyCodesListGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Return type

array[CurrencyCode]

Example data

Content-Type: application/json

[ "EUR", "EUR" ]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

Example data

Content-Type: application/json

["EUR"]

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### get /rapi/enumerators/v1/currencyCodes/translate

Preloženie hodnoty číselníka mien (**rapiEnumeratorsV1CurrencyCodesTranslateGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Query parameters

**code (optional)**

*Query Parameter* —

Return type

String

Example data

Content-Type: application/json

""

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

String

Example data

Content-Type: application/json

EUR

400

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidContract","data":[]}}

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### get /rapi/enumerators/v1/errorCodes/list

Získanie číselníka chýb (**rapiEnumeratorsV1ErrorCodesListGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Return type

array[ErrorCode]

Example data

Content-Type: application/json

[ "InvalidField", "InvalidField" ]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

Example data

Content-Type: application/json

["InvalidField","EmptyField","InvalidDataType","MissingFile","MissingPackage","InvalidFileSign","InvalidFileEncryption","InvalidFileCompresion","NonrelativePackage","InconsistentPackage","InvalidRowCount"]

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### get /rapi/enumerators/v1/errorCodes/translate

Preloženie hodnoty číselníka chýb (**rapiEnumeratorsV1ErrorCodesTranslateGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Query parameters

**code (optional)**

*Query Parameter* —

Return type

String

Example data

Content-Type: application/json

""

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

String

Example data

Content-Type: application/json

EmptyField

400

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidContract","data":[]}}

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### get /rapi/enumerators/v1/packageKinds/list

Získanie číselníka druhov dátových balíčkov prevádzkovateľa hazardnej hry (**rapiEnumeratorsV1PackageKindsListGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Return type

array[PackageKind]

Example data

Content-Type: application/json

[ "CharitativnaLoteria", "CharitativnaLoteria" ]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

Example data

Content-Type: application/json

["CharitativnaLoteria","StolovaHra","StavkovaHra","HazardnaHraNaVP","HazardnaHraNaTV","HazardnaHraNaTZ","HazardnaHraNaITZ","CiselnaLoteria","Bingo","InternetovaHra"]

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### get /rapi/enumerators/v1/packageStates/list

Získanie číselníka možných stavov výsledku spracovania dátového balíčka prevádzkovateľa hazardnej hry (**rapiEnumeratorsV1PackageStatesListGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Return type

array[PackageState]

Example data

Content-Type: application/json

[ "Processing", "Processing" ]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

Example data

Content-Type: application/json

["Processing","ProcessedSuccessfully","Rollbacked","InvalidFormat","InvalidContent","InvalidSigningCertificate","InvalidEncryptingCertificate","InvalidZipFile","InvalidPackageUrl"]

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### get /rapi/enumerators/v1/processStates/list

Získanie číselníka možných stavov procesu spracovania na serveri (**rapiEnumeratorsV1ProcessStatesListGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Return type

array[ProcessState]

Example data

Content-Type: application/json

[ "Planned", "Planned" ]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

Example data

Content-Type: application/json

["Planned","Processing","Processed","Failed"]

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### get /rapi/enumerators/v1/systemStates/list

Získanie číselníka možných stavov klientskeho alebo serverového technického prostredia (**rapiEnumeratorsV1SystemStatesListGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Return type

array[SystemState]

Example data

Content-Type: application/json

[ "Ok", "Ok" ]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

Example data

Content-Type: application/json

["Ok","BeforeShutdown","AfterShutdown","UnexpectedShutdown"]

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

## Health

### get /health

Metóda k opakovanému získavaniu stavu aplikačného rozhrania (**healthGet**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Return type

HealthReport

Example data

Content-Type: application/json

{

"totalDuration" : "totalDuration",

"entries" : {

"key" : {

"duration" : "duration",

"exception" : "",

"data" : {

"key" : ""

},

"description" : "description",

"status" : "Unhealthy",

"tags" : [ "tags", "tags" ]

}

}

}

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

Success HealthReport

503

Server Error HealthReport

## Reporting

### post /rapi/reporting/v1/package/getStatus

Získanie stavu o spracovaní dátového balíčka prevádzkovateľa hazardnej hry (**rapiReportingV1PackageGetStatusPost**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Consumes

This API call consumes the following media types via the Content-Type request header:

* application/json

Request body

**body** RequestIdRequest **(optional)**

*Body Parameter* —

*example: { "requestId" : "8f430525-c9c5-4caf-81ef-728b20e861c2" }*

Return type

GetStatusResponse

Example data

Content-Type: application/json

{

"processingState" : "Planned",

"requestId" : "046b6c7f-0b8a-43b9-b35d-6489e6daee91",

"details" : [ {

"result" : "result",

"classificationCode" : "Error",

"file" : "file",

"line" : 0,

"errorCode" : "InvalidField",

"detail" : "detail"

}, {

"result" : "result",

"classificationCode" : "Error",

"file" : "file",

"line" : 0,

"errorCode" : "InvalidField",

"detail" : "detail"

} ],

"packageState" : "Processing"

}

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

GetStatusResponse

Example data

Content-Type: application/json

{"requestId":"8f430525-c9c5-4caf-81ef-728b20e861c2","processingState":"Planned","packageState":"Processing"}

400

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidRequestId","data":[]}}

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### post /rapi/reporting/v1/package/list

Získanie zoznamu dátových balíčkov prevádzkovateľa hazardnej hry (**rapiReportingV1PackageListPost**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Consumes

This API call consumes the following media types via the Content-Type request header:

* application/json

Request body

**body** ListRequest **(optional)**

*Body Parameter* —

Return type

ListResponse

Example data

Content-Type: application/json

{

"packages" : [ {

"processingState" : "Planned",

"packageKind" : "CharitativnaLoteria",

"packageTimeFrameEndExclusive" : "2000-01-23T04:56:07.000+00:00",

"packageTimeFrameStartInclusive" : "2000-01-23T04:56:07.000+00:00",

"packageGuid" : "046b6c7f-0b8a-43b9-b35d-6489e6daee91",

"packageVersion" : "packageVersion",

"packageState" : "Processing"

}, {

"processingState" : "Planned",

"packageKind" : "CharitativnaLoteria",

"packageTimeFrameEndExclusive" : "2000-01-23T04:56:07.000+00:00",

"packageTimeFrameStartInclusive" : "2000-01-23T04:56:07.000+00:00",

"packageGuid" : "046b6c7f-0b8a-43b9-b35d-6489e6daee91",

"packageVersion" : "packageVersion",

"packageState" : "Processing"

} ]

}

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

ListResponse

400

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidContract","data":[]}}

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### post /rapi/reporting/v1/package/rollback

Vrátenie spracovania dátového balíčka prevádzkovateľa hazardnej hry, odstránenie zo spracovania na serveri (**rapiReportingV1PackageRollbackPost**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Consumes

This API call consumes the following media types via the Content-Type request header:

* application/json

Request body

**body** RequestIdRequest **(optional)**

*Body Parameter* —

*example: { "requestId" : "8f430525-c9c5-4caf-81ef-728b20e861c2" }*

Return type

StatusResponse

Example data

Content-Type: application/json

{

"requestId" : "046b6c7f-0b8a-43b9-b35d-6489e6daee91",

"statusCode" : "Awaiting"

}

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

StatusResponse

Example data

Content-Type: application/json

{"requestId":"8f430525-c9c5-4caf-81ef-728b20e861c2","statusCode":"Accepted"}

400

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidRequestId","data":[]}}

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### post /rapi/reporting/v1/package/load

Odovzdáva dátový balíček prevádzkovateľa hazardnej hry na server (**rapiReportingV1PackageloadPost**)

Volanie tejto metódy predpokladá úspešné overenie voči aplikačnému rozhraniu

Consumes

This API call consumes the following media types via the Content-Type request header:

* application/json

Request body

**body** PackageloadRequest **(optional)**

*Body Parameter* —

*example: { "requestId" : "8f430525-c9c5-4caf-81ef-728b20e861c2", "operatorCode" : "12345678", "packageVersion" : "01", "packageKind" : "CharitativnaLoteria", "packageTimeFrameStartInclusive" : "2021-06-03T00:00:00+02:00", "packageTimeFrameEndExclusive" : "2021-06-04T00:00:00+02:00", "packageContent" : "", "packageUri" : "https://example.com/8f430525-c9c5-4caf-81ef-728b20e861c2.zip.p7e.p7s", "packageStamp" : "2021-06-04T16:03:01.1501664+02:00" }*

Return type

StatusResponse

Example data

Content-Type: application/json

{

"requestId" : "046b6c7f-0b8a-43b9-b35d-6489e6daee91",

"statusCode" : "Awaiting"

}

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

StatusResponse

Example data

Content-Type: application/json

{"requestId":"8f430525-c9c5-4caf-81ef-728b20e861c2","statusCode":"Received"}

400

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidRequestId","data":[]}}

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

## Models

[ Jump to Methods ]

### Table of Contents

1. ClassificationCode
2. CurrencyCode
3. EmptyRequest
4. ErrorCode
5. Fault
6. FaultCode
7. FaultResponse
8. GetStatusResponse
9. HealthReport
10. HealthReportEntry
11. HealthStatus
12. IsAliveRequest
13. IsAliveResponse
14. ListRequest
15. ListResponse
16. OperatorPackage
17. PackageKind
18. PackageState
19. PackageloadRequest
20. ProcessState
21. RequestIdRequest
22. StatusCode
23. StatusDetail
24. StatusResponse
25. SystemState
26. VersionResponse

### ClassificationCode

### CurrencyCode

### EmptyRequest

### ErrorCode

### Fault

**code (optional)**

*FaultCode*

**data (optional)**

*array[String]*

### FaultCode

### FaultResponse

**fault (optional)**

*Fault*

### GetStatusResponse

**requestId (optional)**

*UUID* format: uuid

**processingState (optional)**

*ProcessState*

**packageState (optional)**

*PackageState*

**details (optional)**

*array[StatusDetail]*

### HealthReport

**entries (optional)**

*map[String, HealthReportEntry]*

**status (optional)**

*HealthStatus*

**totalDuration (optional)**

*String* format: date-span

### HealthReportEntry

**data (optional)**

*map[String, null]*

**description (optional)**

*String*

**duration (optional)**

*String* format: date-span

**exception (optional)**

**status (optional)**

*HealthStatus*

**tags (optional)**

*array[String]*

### HealthStatus

### IsAliveRequest

**systemState (optional)**

*SystemState*

**nextClientShutdown (optional)**

*Date* Deklarácia odstávky klientskeho systému format: date-time

**isAlive (optional)**

*Boolean* Aktuálny súhrnný stav klientskeho systému

### IsAliveResponse

**systemState (optional)**

*SystemState*

**nextServerShutdown (optional)**

*Date* Deklarácia odstávky serveroveho systému format: date-time

**isAlive (optional)**

*Boolean* Aktuálny súhrnný stav serveroveho systému

### ListRequest

**operatorCode (optional)**

*String*

### ListResponse

**packages (optional)**

*array[OperatorPackage]*

### OperatorPackage

**packageGuid (optional)**

*UUID* format: uuid

**packageVersion (optional)**

*String*

**packageKind (optional)**

*PackageKind*

**packageTimeFrameStartInclusive (optional)**

*Date* format: date-time

**packageTimeFrameEndExclusive (optional)**

*Date* format: date-time

**processingState (optional)**

*ProcessState*

**packageState (optional)**

*PackageState*

### PackageKind

### PackageState

### PackageloadRequest

**requestId (optional)**

*UUID* Unikátny kód požiadavky format: uuid

**operatorCode (optional)**

*String* IČO prevádzkovateľa hazardnej hry

**packageVersion (optional)**

*String* Verzia dátového balíčka prevádzkovateľa hazardnej hry

**packageKind (optional)**

*PackageKind*

**packageTimeFrameStartInclusive (optional)**

*Date* Dátum a čas začiatku vykazovaného časového rámca format: date-time

**packageTimeFrameEndExclusive (optional)**

*Date* Dátum a čas ukončenia vykazovaného časového rámca format: date-time

**packageContent (optional)**

*String* <p>Obsah dátového balíčka prevádzkovateľa hazardnej hry (base64)</p> <p>Keď je to možné, neposielajte balíčky tímto spôsobom,ale použite PackageUri a systém si data stiahne sám.</p>

**packageUri (optional)**

*String* Odkaz na obsah dátového balíčka prevádzkovateľa hazardnej hry

**packageStamp (optional)**

*Date* Doba a čas generovania dátového balíčka prevádzkovateľa hazardnej hry format: date-time

### ProcessState

### RequestIdRequest

**requestId (optional)**

*UUID* Unikátny kód požiadavky format: uuid

### StatusCode

### StatusDetail

**file (optional)**

*String*

**line (optional)**

*Integer* format: int32

**classificationCode (optional)**

*ClassificationCode*

**errorCode (optional)**

*ErrorCode*

**detail (optional)**

*String*

**result (optional)**

*String*

### StatusResponse

**requestId (optional)**

*UUID* Unikátny kód požiadavky format: uuid

**statusCode (optional)**

*StatusCode*

### SystemState

### VersionResponse

**version (optional)**

*String*

**lastdate (optional)**

*Date* format: date-time

# Aplikačné rozhranie pre reporting – Operator API

Json description

More information: https://helloreverb.com

Contact Info: hello@helloreverb.com

Version: v1

All rights reserved

http://apache.org/licenses/LICENSE-2.0.html

## Table of Contents

### Notifier

* post /oapi/reporting/v1/notifier/setStatus

## Notifier

### post /oapi/reporting/v1/notifier/setStatus

(**oapiReportingV1NotifierSetStatusPost**)

Consumes

This API call consumes the following media types via the Content-Type request header:

* application/json

Request body

**body** NotifierSetStatusRequest **(optional)**

*Body Parameter* —

Return type

EmptyResponse

Example data

Content-Type: application/json

{ }

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

* application/json

Responses

200

EmptyResponse

400

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidContract","data":[]}}

403

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInvalidCertificate","data":[]}}

500

FaultResponse

Example data

Content-Type: application/json

{"fault":{"code":"ErrInternalServerError","data":[]}}

### Table of Contents

1. ClassificationCode
2. EmptyResponse
3. ErrorCode
4. Fault
5. FaultCode
6. FaultResponse
7. NotifierSetStatusRequest
8. PackageState
9. ProcessState
10. StatusDetail

### ClassificationCode

### EmptyResponse

### ErrorCode

### Fault

**code (optional)**

*FaultCode*

**data (optional)**

*array[String]*

### FaultCode

### FaultResponse

**fault (optional)**

*Fault*

### NotifierSetStatusRequest

**requestId (optional)**

*UUID* format: uuid

**processingState (optional)**

*ProcessState*

**packageState (optional)**

*PackageState*

**details (optional)**

*array[StatusDetail]*

**content (optional)**

*String*

### PackageState

### ProcessState

### StatusDetail

**file (optional)**

*String*

**line (optional)**

*Integer* format: int32

**classificationCode (optional)**

*ClassificationCode*

**errorCode (optional)**

*ErrorCode*

**detail (optional)**

*String*

**result (optional)**

*String*

## Prípady komunikácie

V reálné situaci mohou nastat následující komunikační scénáře.

### Kód: SEQ\_1

**Název:** Opakované zasílání zpráv při nepřijetí požadavku

**Primární uživatel:** Operator (provozovatel)

**Popis:** Provozovatel poskytuje opakovaně jednotlivý datový balíček do okamžiku, kdy mu orgán státní správy předá potvrzující zprávu o přijetí balíčku. Nepřipouští se možnost, že by zpráva obsahující datový balíček nebyla doručena.

**Speciální požadavky:** Žádné



### Kód: SEQ\_2

**Název:** Opakované zasílání zpráv při nedostupnosti protistrany

**Primární uživatel:** Operator (provozovatel)

**Popis:** Provozovatel opakovaně poskytuje jednotlivý datový balíček také v případě, pokud komunikační rozhraní státní správy není dostupné. Nepřipouští se možnost, že by zpráva obsahující datový balíček nebyla doručena.

**Speciální požadavky:** Žádné



### Kód: SEQ\_3

**Název:** Odeslání informace o stavu zpracování

**Primární uživatel:** Operator (provozovatel)

**Popis:** Orgán státní správy zasílá potvrzující zprávu na komunikační rozhraní provozovatele. V případě, že komunikační rozhraní odesílatele není dostupné, je na vnitřní logice zpracovatelského serveru, zdali bude opakovaně odesílat notifikace o stavu, ale minimálně jednou se musí pokusit doručit zprávu. V případě, že není zpráva úspěšně doručena, má odesílatel možnost kdykoliv si zjistit stav zpracování balíčku přes komunikační rozhraní serveru protistrany.

**Speciální požadavky:** Žádné



### Kód: SEQ\_4

**Název:** Opravné zasílání zpráv při chybném obsahu požadavku

**Primární uživatel:** Operator (provozovatel)

**Popis:** Chyby v automatizovaném výstupu se opravují tak, že provozovatel znovu poskytne datový balíček, ve kterém tyto chyby odstraní.

**Speciální požadavky:** Žádné



# Referenční klient komunikačního rozhraní dle vyhlášky č. 445/2020 – Nenahrazuje QSCD

## Úvod

Referenční klient je publikován a spravován GitHub repositářem „sk\_reporting-work“ dostupný na adrese: <https://github.com/cbData/sk_reporting-work>.

S ohledem na legislativní a technologické okolnosti bylo možné vycházet z implementace referenčního klienta publikovaného Ministerstvem České republiky pro potřeby Vyhlášky č. 10/2019 Sb.- Vyhláška o způsobu oznamování a zasílání informací a přenosu dat provozovatelem hazardních her, rozsahu přenášených dat a jiných technických parametrech přenosu dat.

Všechny podklady vztažené k výše uvedené implementaci referenčního klienta jsou k dispozici na adrese GitHub repositáře: <https://github.com/Ministerstvo-financi/hazard-komunikacni-rozhrani>

Licencování výše uvedené implementaci referenčního klienta je následující:

*Copyright [2019] [CGI IT Czech Republic s.r.o., Ministerstvo financí České republiky, Státní pokladna Centrum sdílených služeb, s. p.]*

*Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at*

[*http://www.apache.org/licenses/LICENSE-2.0*](http://www.apache.org/licenses/LICENSE-2.0)

Více informací ohledně *Apache License, Version 2.0* je k dispozici zde: <https://cs.wikipedia.org/wiki/Licence_Apache>

Tato forma licencování umožňuje uživateli svobodné užívání softwaru k různým účelům; distribuci, upravování, následné redistribuci upravené verze softwaru apod. To vše je možné, aniž by došlo k porušení licenčních práv.

## Popis

GitHub repositářem „sk\_reporting-work“ se odkazuje na základní GitHub repositář MF ČR, z kterého přebírá přeloženou knihovnu.

|  |  |
| --- | --- |
| **Název** | **Popis** |
| Verze JDK | jdk-11.0.8 |
| Soubor | crypto\_cli-1.0-jar-with-dependencies.jar |

Nástroje zveřejněné v tomto repositáři umožňují naplnit požadavky vyhlášky následovně:

**crypto\_utils** a **crypto\_cli**

* Java knihovna a nástroj pro spuštění jednotlivých funkcí z příkazové řádky
* vytváření uznávané elektronické pečeti datového balíčku
* ověřování uznávané a kvalifikované elektronické pečeti potvrzovacího balíčku
* šifrování datového balíčku před
* dešifrování potvrzovacího balíčku

## Komprimace

Z titulu snížení objemu přenášených dat jsou textová data komprimována ve formátu ZIP – ISO/IEC 21320-1:2015.

|  |  |
| --- | --- |
| **Platforma** | **Popis** |
| Windows | Spustit Powershell a zadat následující příkazy upravené podle provozního nastavení  C:  cd c:\hazard  cd data\HazardnaHraNaTV\_2021-05-01T05\_1.dat  Compress-Archive -Path \*.csv -DestinationPath ..\HazardnaHraNaTV\_2021-05-01T05\_1  Ve složce c:\hazard\data vznikne soubor HazardnaHraNaTV\_2021-05-01T05\_1.zip |

## Dekomprimace

Z titulu snížení objemu přenášených dat jsou textová data komprimována ve formátu ZIP – ISO/IEC 21320-1:2015.

|  |  |
| --- | --- |
| **Platforma** | **Popis** |
| Windows | Spustit Powershell a zadat následující příkazy upravené podle provozního nastavení  C:  cd c:\hazard  cd data\HazardnaHraNaTV\_2021-05-01T05\_1.dat.zip  Decompress-Archive -Path \*.csv -DestinationPath ..\HazardnaHraNaTV\_2021-05-01T05\_1  Ve složce c:\hazard\data vznikne soubor HazardnaHraNaTV\_2021-05-01T05\_1.dat |

## Šifrování

S ohledem na požadavky ochrany dat jsou herní a finanční data šifrována veřejnými klíči komerčních certifikátů provozovatele a serveru státní správy. Následně je možné na straně příjemce data dešifrovat. Při šifrování se zašifrovaný soubor uloží ve formátu Cryptographic message syntax (CMS) s využitím typu obsahu Enveloped-Data podle standardu RFC 5652.

Data jsou šifrovány dvěma veřejnými klíčí

* ŠIFROVACÍ CERTIFIKÁT klienta (provozovatele)
* ŠIFROVACÍ CERTIFIKÁT serveru (URHH)

Data jsou dešifrována vždy příslušným privátním klíčem protistrany.

Na základě šifrování obsahu veřejným klíčem systému státní správy dojde k dešifrování dat privátním klíčem umístěným na systému příjemce zprávy. Zašifrovaný soubor je uložen ve formátu Cryptographic message syntax (CMS) s využitím typu obsahu Enveloped-Data podle standardu RFC 5652.

|  |  |
| --- | --- |
| **Platforma** | **Popis** |
| Windows | c:\jdk-11.0.8\bin\java.exe  -jar c:\URHH-github\crypto\_cli\lib\crypto\_cli-1.0-jar-with-dependencies.jar  -f encryptFile  -i c:\URHH-github\data\empty-packages\reponse.zip  -o c:\URHH-github\data\empty-packages\reponse.zip.p7e  -cert c:\URHH-github\data\urhh\certs\encrypt.pem  -cert c:\URHH-github\data\operator\certs\encrypt.pem |

[image](https://user-images.githubusercontent.com/29448695/118829552-7f103e00-b8be-11eb-848b-5f0873673535.png)

## Dešifrování

Dešifrování je reverzní proces k procesu šifrování. Data jsou dešifrována vždy vlastním privátním klíčem a to za předpokladu, že data byla na klientské straně zašifrovány veřejním klíčem serveru.

Zašifrovaný soubor je uložen ve formátu Cryptographic message syntax (CMS) s využitím typu obsahu Enveloped-Data podle standardu RFC 5652.

|  |  |
| --- | --- |
| **Platforma** | **Popis** |
| Windows | c:\jdk-11.0.8\bin\java.exe  -jar c:\URHH-github\crypto\_cli\lib\crypto\_cli-1.0-jar-with-dependencies.jar  -f decryptFile  -i c:\URHH-github\data\empty-packages\2-Bingo\_2021-05-01T05\_1.zip.p7e  -o c:\URHH-github\data\empty-packages\2-Bingo\_2021-05-01T05\_1.zip  -k c:\URHH-github\data\urhh\certs\encrypt.key  -cert c:\URHH-github\data\urhh\certs\encrypt.pem |

## Vytvoření elektronické pečeti

Nabízí se pečetění, kdy soubor provozovatele je opatřen uznávanou elektronickou pečetí ve formátu CAdES. Jak je uvedeno výše, vyhláška č. 445/2020 Z. z. vyžaduje kvalizikovaný proces pečetění prostředky QSCD.

Pro potřeby pilotního nebo testovacího provozu je možní realizovat pečetící proces prostřednictvím zdokonalené elektronické pečetě následujícím procesem:

Soubor se pečetí privátním klíčem provozovatele. URHH je schopno pečeť na základě veřejné části od provozovatele ověřit.

|  |  |
| --- | --- |
| **Platforma** | **Popis** |
| Windows | c:\jdk-11.0.8\bin\java.exe  -jar c:\URHH-github\crypto\_cli\lib\crypto\_cli-1.0-jar-with-dependencies.jar  -f signFile  -i c:\URHH-github\data\empty-packages\data.zip.p7e  -o c:\URHH-github\data\empty-packages\data.zip.p7e.p7s  -spks c:/URHH-github/data/operator/certs/sign.pfx  -pass heslo123 |

## Ověření elektronické pečeti

Dojde k ověření identity pečetě na základě zveřejněného veřejného klíče kvalifikovaného certifikátu odesílatele.

Pro validaci pečeti potvrzovacího balíčku je nezbytné získat certifikát, který komunikační rozhraní používá k vytváření elektronických pečetí.

Pro validaci pečeti souboru je možné použít /crypto\_cli/validateFile.bat INPUT OUTPUT CERT1 CERT2 s následujícími parametry:

* INPUT – zapečetěný soubor, jehož pečeť validujeme
* OUTPUT – výsledek ověření pečeti – data, která jsou v souboru INPUT zapečetěna
* CERT1 – certifikát, kterým se očekává, že je pečeť vytvořena – první kandidát
* CERT2 – certifikát, kterým se očekává, že je pečeť vytvořena – druhý kandidát - může být shodný s prvním, pokud existuje jen jeden možný certifikát, kterým má být zapečetěno

Spustí se validace, která nejprve provádí načtení EU Trust listu. Poté proběhne validace podpisu/pečeti, extrakce podepsaných/pečetěných dat a výpis výsledku validace.

Příklad:

*2019-03-09 23:57:16 INFO DssCommands:57 - Starting validateFile*

*2019-03-09 23:57:16 ERROR DssValidator:245 - FAILED to read keystore file - using default keystore*

*[main] INFO eu.europa.esig.dss.tsl.service.TSLRepository - New version of EU TSL is stored in cache*

*WARNING: An illegal reflective access operation has occurred*

*WARNING: Illegal reflective access by com.sun.xml.bind.v2.runtime.reflect.opt.Injector (file:/T:/proj/SPCSS/2018/analyticky-modul/repo/hazard-komunikacni-rozhrani/build/hazard-2019-03-10.1552227930/crypto\_utils/crypto\_cli-1.0-jar-with-dependencies.jar) to method java.lang.ClassLoader.defineClass(java.lang.String,byte[],int,int)*

*WARNING: Please consider reporting this to the maintainers of com.sun.xml.bind.v2.runtime.reflect.opt.Injector*

*WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations*

*WARNING: All illegal access operations will be denied in a future release*

*[main] INFO eu.europa.esig.dss.tsl.service.TSLRepository - New version of the pivot LOTL 'https://ec.europa.eu/information\_society/policy/esignature/trusted-list/tl-pivot-172-mp.xml' is stored in cache*

*[pool-1-thread-3] INFO eu.europa.esig.dss.validation.CommonCertificateVerifier - + New CommonCertificateVerifier created.*

*[pool-1-thread-3] INFO eu.europa.esig.dss.validation.SignedDocumentValidator - Validator 'eu.europa.esig.dss.asic.validation.ASiCContainerWithCAdESValidator' is registred*

*[pool-1-thread-3] INFO eu.europa.esig.dss.validation.SignedDocumentValidator - Validator 'eu.europa.esig.dss.pades.validation.PDFDocumentValidator' is registred*

*...*

*...*

*[main] INFO eu.europa.esig.dss.tsl.service.TSLRepository - Nb of loaded trusted lists : 30/32*

*[main] INFO eu.europa.esig.dss.tsl.service.TSLRepository - Nb of trusted certificates : 2403*

*[main] INFO eu.europa.esig.dss.tsl.service.TSLRepository - Nb of trusted public keys : 2311*

*[main] INFO eu.europa.esig.dss.validation.CommonCertificateVerifier - + New CommonCertificateVerifier created.*

*[main] INFO eu.europa.esig.dss.validation.SignedDocumentValidator - Document validation...*

*[main] INFO eu.europa.esig.dss.x509.ocsp.OCSPToken - OCSP status is good*

*2019-03-09 23:59:21 INFO Main:74 - ValidateFile INFO\_PKG\_SIG\_CERT\_OK*

V případě úspěšné validace bude také vytvořen OUTPUT soubor.

|  |  |
| --- | --- |
| **Platforma** | **Popis** |
| Windows | c:\jdk-11.0.8\bin\java.exe  -jar c:\URHH-github\crypto\_cli\lib\crypto\_cli-1.0-jar-with-dependencies.jar  -f validateFile  -i c:\URHH-github\data\empty-packages\Bingo\_2021-05-01T05\_1.zip.p7e.p7s  -o c:\URHH-github\data\empty-packages\2-Bingo\_2021-05-01T05\_1.zip.p7e  -cert c:\URHH-github\data\operator\certs\sign.pem  -cert c:\URHH-github\data\operator\certs\sign.pem |