HTTP API manuál pro zařízení 2N

v.2.46

www.2n.com

Obsah:

1. Úvod

HTTP API je aplikační rozhraní pro ovládání vybraných funkcí zařízení pomocí **HTTP** protokolu. Toto rozhraní umožňuje jednoduše integrovat zařízení 2N s produkty třetích stran, např. systémy domácí automatizace, zabezpečovací a monitorovací systémy budov apod.

HTTP API je podle funkce rozděleno do následujících služeb:

- **System API** – umožňuje změny konfigurace, získání stavu a upgrade zařízení.
- **Switch API** – umožňuje řízení a sledování stavu spínačů, např. otevírání dveřních zámků apod.
- **I/O API** – umožňuje řízení a sledování logických vstupů a výstupů zařízení.
- **Audio API** – umožňuje řízení přehrávání zvuků a monitorování mikrofonu zařízení.
- **Camera API** – umožňuje řízení a sledování obrazu z kamery.
- **Display API** – umožňuje řízení displeje a zobrazování uživatelských informací na displeji.
- **E-mail API** – umožňuje ze zařízení odesílat uživatelské e-maily.
- **Phone/Call API** – umožňuje řízení a sledování příchozích a odchozích hovorů.
- **Logging API** – umožňuje vyčítat zaznamenané události zařízení

Pro každou službu lze nastavit transportní protokol (**HTTP** nebo **HTTPS**) a způsob autentizace (**žádná**, **Basic** nebo **Digest**). V konfiguraci **HTTP API** lze vytvořit až pět uživatelských účtů (s vlastním jménem a heslem) s možností detailního řízení přístupu k jednotlivým službám a funkcím.

HTTP API se konfiguruje pomocí konfiguračního webového rozhraní zařízení v záložce **Služby / HTTP API**. Zde lze povolovat a konfigurovat jednotlivé služby a nastavovat parametry uživatelských účtů. Pro demonstraci a odzkoušení funkce **HTTP API** slouží speciální nástroj integrovaný v **HTTP** serveru zařízení dostupný na adrese **http(s)://ip_adresa_zařízení/apitest.html**.

Upozornění

Varování
Za účelem dosažení plné funkčnosti a zaručených výkonů důrazně doporučujeme vždy již při instalaci ověřit aktuálnost používané verze produktu či zařízení. Zákazník tímto bere na vědomí, že produkt či zařízení může dosahovat zaručených výkonů a být plně funkční dle propozic výrobce pouze v případě, je-li používána nejnovější verze produktu či zařízení, která byla otestována na plnou interoperabilitu a která nebyla výrobcem označena jako nekompatibilní s určitými verzemi jiných produktů, a to pouze v souladu s pokyny, návodem či doporučením výrobce a pouze ve spojení s vyhovujícími produkty a zařízeními jiných výrobců. Nejnovější verze jsou dostupné na internetových stránkách https://www.2n.com/cs_CZ/, popř. jednotlivá zařízení podle svých technických možností umožňují aktualizaci v konfiguračním rozhraní. Používá-li zákazník jinou než nejnovější verzi produktu či zařízení, popř. používá-li verzi, kterou výrobce označil za nekompatibilní s určitými verzemi jiných produktů, nebo používá-li zákazník produkt či zařízení v rozporu s pokyny, návodem či doporučením výrobce nebo ve spojení s nevyhovujícími produkty či zařízeními jiných výrobců, je srozuměn s veškerými případnými omezeními funkčnosti takového produktu či zařízení a s důsledky s tím spojenými. Použitím jiné než nejnovější verze produktu či zařízení, popř. verze, kterou výrobce označil za nekompatibilní s určitými verzemi jiných produktů, nebo použitím produktu či zařízení v rozporu s pokyny, návodem či doporučením výrobce, popř. použitím s nevyhovujícími produkty či zařízeními jiných výrobců, zákazník souhlasí s tím, že společnost 2N TELEKOMUNIKACE a.s. není odpovědná za jakékoli omezení funkčnosti takového produktu ani za újmu související s takovým případným omezením funkčnosti.

1.1 Revize dokumentu

1.1 Revize dokumentu

Verze	Popis změn
2.44	• Podpora HTTP API zařízením **2N LiftIP 2.0**. • Doplněna nová funkce **api/system/timezone**. • Doplněna nová funkce **api/system/timezone/caps**. • Rozšířena funkce **api/system/time** o metodu **PUT** pro konfiguraci času.
2.42	• Doplněna nová funkce **api/cert/ca** • Doplněna nová funkce **api/cert/user**
2.40	• Rozšířena funkce **/api/lpr/licenseplate** o parametry **lprID** a **lprDir**.

Verze	Popis změn
2.39	• Doplněna nová událost **DtmfSent**.
2.38	• Událost **MotionDetection** rozšířena o parametr **ID** specifikující číslo profilu detekce pohybu ve webovém rozhraní.
2.37	• Doplněna nová funkce **api/accesspoint/grantaccess**. • Doplněna nová událost **ApiAccessRequested** generovaná při zaslání požadavku na /api/accesspoint/grantaccess s výsledkem "success" : true.
2.36	• Rozšířena funkce **api/switch/status** o parametr **holdTimeout**. • Rozšířena funkce **api/switch/ctrl** o parametr **timeout**. • Doplněna funkce **api/phone/calllog** pro stahování záznamu volání a mazání celého obsahu nebo jeho jednotlivých záznamů.
2.35	• Doplněna funkce **api/system/time** pro získávání času zařízení. • Doplněna funkce **api/system/time/set** pro nastavení času zařízení. • Doplněna funkce api/automation/trigger pro aktivaci funkce automatizace události HttpTrigger. • Doplněn nový parametr **users** do funkce **api/call/dial** pro volání na jednoho či více uživatelů.
2.34	• Doplněna funkce **api/lift/grantaccess** pro aktivaci pater výtahu na základě autorizace v jiném zařízení.
2.33	• Doplněna funkce **/api/pcap/live**, **api/pcap/live/stop**, **api/pcap/stats** pro ovládání zachytávání příchozích a odchozích paketů.
2.32	• Doplněna funkce **/api/lpr/licenseplate** pro řízení přístupu pomocí rozpoznání registrační značky vozidla. • Doplněna funkce **api/lpr/image** pro získávání obrázků přijatých z rozpoznávání registračních značek vozidla.
2.31	• Doplněna funkce **/api/mobilekey/config** pro čtení a zapisování identifikátorů lokace a šifrovacích klíčů pro autentizaci přes Bluetooth.

Verze	Popis změn
2.30	• Parametr **apbBroken** odstraněn z události **AccessTaken**. • Parametr **apbBroken** přidán do události **UserAuthenticated**.
2.29	• Doplněna nová funkce **api system caps**. • Doplněna nová událost **CapabilitiesChanged**.
2.28	• Beze změny.
2.27	• Doplněny nové události: **LiftStatusChanged**, **LiftConfigChanged**, **LiftFloorEnabled**. • Doplněny funkce **api holidays**, **api config holidays**, **api dir template**, **api dir create**, **api dir update**, **api dir delete**, **api dir get**, **api dir query**.
2.26	• Doplněny nové události: **DtmfEntered**, **AccessTaken**, **ApLockStateChanged**, **RexActivated**.
2.25	• Beze změny.
2.24	• Změna nahrávání uživatelů do adresáře z důvodu zrušení pozic.
2.23	• Událost **UserRejected** rozšířena o parametr **switchDisabled**.
2.22	• Doplněny nové události **CardHeld**, **PairingStateChanged**, **SwitchesBlocked**, **FingerEntered**, **MobKeyEntered**, **DoorStateChanged**, **UserRejected**, **DisplayTouched**.
2.21	• Rozšířena funkce **api/display/image** (parametry **display**, **blob-image**, **blob-video**, **duration**, **repeat**) pro **2N® IP Verso**. • Doplněny nové události **UserAuthenticated**, **SilentAlarm**, **AccessLimited**. • Doplněn nový parametr **timeSpan** do funkce **/api/email/send**.

Verze	Popis změn
2.15	• Doplněny nové události **TamperSwitchActivated**, **UnauthorizedDoorOpen**, **DoorOpenTooLong**, **LoginBlocked**. • Události rozšířeny o parametr **tzShift** udávající rozdíl mezi místním a UTC časem. • Funkce **email/send** rozšířena o možnost nastavení rozlišení odesílaných obrázků.
2.14	• Doplněny funkce **api/pcap**, **api/pcap/restart**, **api/pcap/stop** pro stahování a řízení záznamu síťového provozu. • Doplněna funkce **audio/test** pro spuštění automatického audio testu. • Doplněna funkce **email/send** pro odesílání e-mailu. • Doplněn nový parametr **response** do funkcí **/api/io/ctrl** a **/api/switch/ctrl**. • Funkce **/call/hangup** rozšířena o nový parametr **reason** umožňující zadat důvod ukončení hovoru. • Doplněny nové události **MotionDetected, NoiseDetected** a **SwitchStateChanged**. • Událost **CallStateChanged** rozšířena o parametr **reason** specifikující důvod ukončení hovoru.
2.13	• První verze dokumentu.

2. Popis protokolu HTTP API

Všechny příkazy **HTTP API** jsou odesílány pomocí **HTTP** nebo **HTTPS** protokolu na adresu interkomu s absolutní cestou doplněným prefixem **/api**. Volba protokolu závisí na aktuálním nastavení interkomu v sekci **Služby / HTTP API**. Funkce **HTTP API** jsou rozděleny do služeb a u každé služby je možné nastavit požadovanou úroveň zabezpečení včetně požadavku na **TLS** spojení (tj. **HTTPS** protokol)

Příklad: *Sepnutí spínače 1* http://10.0.23.193/api/switch/ctrl?switch=1&action=on

Absolutní cesta obsahuje název skupiny funkcí (systém, firmware, config, switch apod.) a název funkce samotné (caps, status, ctrl apod.).
Minimalistická varianta požadavku akceptovaná interkomem musí obsahovat řádek požadavku s metodou a absolutní cestou následovaný hlavičkou Host:

Příklad:

```
GET /api/system/info HTTP/1.1
Host: 10.0.23.193
HTTP Server interkomu odpoví zprávou:
HTTP/1.1 200 OK
Server: HIP2.10.0.19.2
Content-Type: application/json
Content-Length: 253
{
  "success" : true,
  "result" : {
    "variant" : "2N IP Vario",
    "serialNumber" : "08-1860-0035",
    "hwVersion" : "535v1",
    "swVersion" : "2.10.0.19.2",
    "buildType" : "beta",
    "deviceName" : "2N IP Vario"
  }
}
```

V této kapitole dále naleznete:

2.1 Metody HTTP protokolu

2N IP interkom využívá následující čtyři metody HTTP protokolu:

- **GET** – požadavky stahující obsah z interkomu nebo provádějící obecné příkazy
- **POST** – požadavky stahující obsah z interkomu nebo provádějící obecné příkazy
- **PUT** – požadavky pro nahrávání obsahu do interkomu
- **DELETE** – požadavky pro odstranění obsahu z interkomu

Metody **GET** a **POST** jsou z hlediska **HTTP API** ekvivalentní a liší se pouze způsobem předávání parametrů (viz následující kapitola). Metody **PUT** a **DELETE** se používají k manipulaci s velkými objekty jako je konfigurace, firmware, obrázky nebo zvukové soubory.

2.2 Parametry požadavků

Prakticky všechny funkce **HTTP API** mohou mít parametry. Parametry jsou pojmenované (switch, action, width, height, blob-image apod.) a jsou uvedeny v popisu příslušné funkce **HTTP API**. Parametry je možné v požadavku předávat třemi způsoby, které lze navzájem kombinovat:

1. v cestě požadavku (uri query, metody **GET**, **POST**, **PUT** a **DELETE**)
2. v obsahu zprávy (application/x-www-form-urlencoded, metody **POST** a **PUT**)
3. v obsahu zprávy (multipart/form-data, metody **POST** a **PUT**) – **RFC-1867**

V případě, že jednotlivé metody předávání parametrů se navzájem kombinují, může nastat situace, kdy je parametr v požadavku uveden vícekrát. V tomto případě se dává přednost poslednímu výskytu parametru.

Parametry funkcí **HTTP API** jsou dvou typů:

1. Parametr s jednoduchou hodnotou (switch, action apod.) může být předán pomocí všech třech výše uvedených metod. Tyto parametry nemají v názvu prefix blob-.
2. Parametr obsahující velká data (např. konfiguraci, firmware, obrázky apod.). Tyto parametry začínají vždy prefixem blob- a lze je předávat pouze pomocí poslední metody (multipart/form-data).

2.3 Odpovědi na požadavky

Odpovědi na požadavky jsou převážně ve formátu **JSON**. Výjimku tvoří pouze požadavky na stažení binárních dat (uživatelské zvuky, obrázky apod.) nebo konfiguraci intercomu v **XML**. Formát odpovědi lze rozlišit dle hlavičky Content-Type. Pro **JSON** jsou definovány tři základní typy odpovědí:

Kladná odpověď bez parametrů

Tato odpověď je odesílána v případě úspěšného provedení požadavku u funkcí nevracejících žádné parametry. Tato odpověď je vždy kombinovaná se stavovým kódem **HTTP** protokolu **200 OK**.

```
{
  "success" : true,
}
```

Kladná odpověď s parametry

Tato odpověď je odesílána v případě úspěšného provedení požadavku u funkcí vracejících doplňkové parametry. Položka **result** obsahuje další parametry odpovědi poplatné dané funkci. Tato odpověď je vždy kombinovaná se stavovým kódem **HTTP** protokolu **200 OK**.

```
{
  "success" : true,
  "result" : {
    ...
  }
}
```

Záporná odpověď při chybě zpracování požadavku

Tento typ odpovědi je odesílán v případě, že při zpracování požadavku dojde k chybě. Odpověď nese kód chyby (parametr **code**), její textový popis (parametr **description**) a případně upřesnění chyby (parametr **param**). Tato odpověď může být kombinovaná se stavovým kódem **HTTP** protokolu **200 OK** nebo **401 Authorization Required**.

```
{
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "port",
    "description" : "invalid parameter value"
  }
}
```

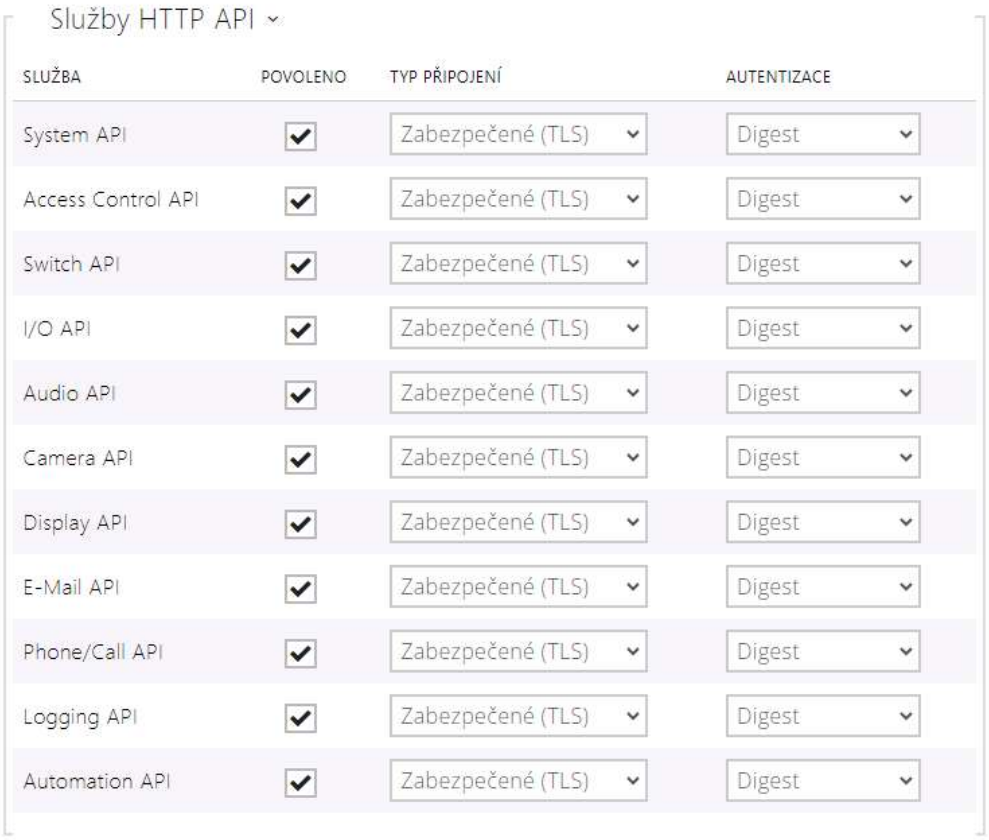
V následující tabulce jsou vyjmenovány možné kódy chyb:

Kód	Popis	
1	function is not supported	Požadovaná funkce není na tomto modelu dostupná.
2	invalid request path	Absolutní cesta specifikovaná v **HTTP** požadavku neodpovídá žádné z funkcí **HTTP API**.
3	invalid request method	Použitá metoda **HTTP** protokolu není pro danou funkci platná.
4	function is disabled	Funkce (příslušná služba) není povolena. Službu je potřeba povolit na stránce konfiguračního rozhraní **Služby / HTTP API**.
7	invalid connection type	Je vyžadováno **HTTPS** připojení.
8	invalid authentication method	Použitá autentizační metoda není pro danou službu povolena. Tato chyba může nastat v případě, kdy pro službu je povolena pouze autentizační metoda Digest a klient se pokouší autentizovat pomocí metody Basic.
9	authorization required	Pro přístup ke službě je vyžadovaná autorizace uživatele. Tato chyba je odesílána společně stavovým kódem **HTTP** protokolu 401 Authorization Required.
10	insufficient user privileges	Autentizovaný uživatel nemá dostatečná privilegia pro provedení funkce.
11	missing mandatory parameter	V požadavku není uveden povinný parametr. Název chybějícího parametru je uveden v položce **param**.
12	invalid parameter value	Hodnota jednoho z parametrů požadavku není platná. Název neplatného parametru je uveden v položce **param**.

Kód	Popis	
13	parameter data too big	Data parametru překračují maximální povolenou velikost. Název chybného parametru je uveden v položce **param**.
14	unspecified processing error	Nastala nespecifikovaná chyba při zpracování požadavku.
15	no data available	Server nemá k dispozici požadovaná data.
17	parameter shouldn't be present	Kolize parametrů (není možné napsat danou kombinaci parametrů).
18	request is rejected	Požadavek nelze nyní zpracovat a byl zařízením odmítnut.
19	file version is lower than minimum	Zadaná verze souboru je nižší, než je požadováno.

3. Zabezpečení služeb HTTP API

V konfiguračním webovém rozhraní **2N IP** interkomu na stránce **Služby / HTTP API** lze nastavovat úroveň zabezpečení jednotlivých služeb **HTTP API**. Služby lze vypnout, zapnout nebo nastavit požadovaný komunikační protokol a způsob autentizace uživatelů.



U každé služby lze nezávisle nastavit požadovaný transportní protokol:

- **HTTP** – požadavky mohou být odesílány pomocí **HTTP** nebo **HTTPS** protokolu. Oba protokoly jsou povoleny a úroveň zabezpečení definuje klient použitým protokolem.
- **HTTPS** – požadavky musí být odesílány pomocí **HTTPS** protokolu a požadavky odesílané pomocí nezabezpečeného **HTTP** protokolu jsou interkomem odmítány. **HTTPS** protokol zajišťuje, že případný útočník nemůže číst obsah zpráv odesílaných a přijímaných zpráv.

U každé služby lze nastavit vyžadovaný způsob autentizace požadavků odesílaných na interkom. Pokud autentizace není provedena, požadavek je odmítnut. Požadavky jsou autentizovány pomocí standardního autentizačního protokolu popsaného v **RFC-2617**. Je možné volit tyto tři metody autentizace:

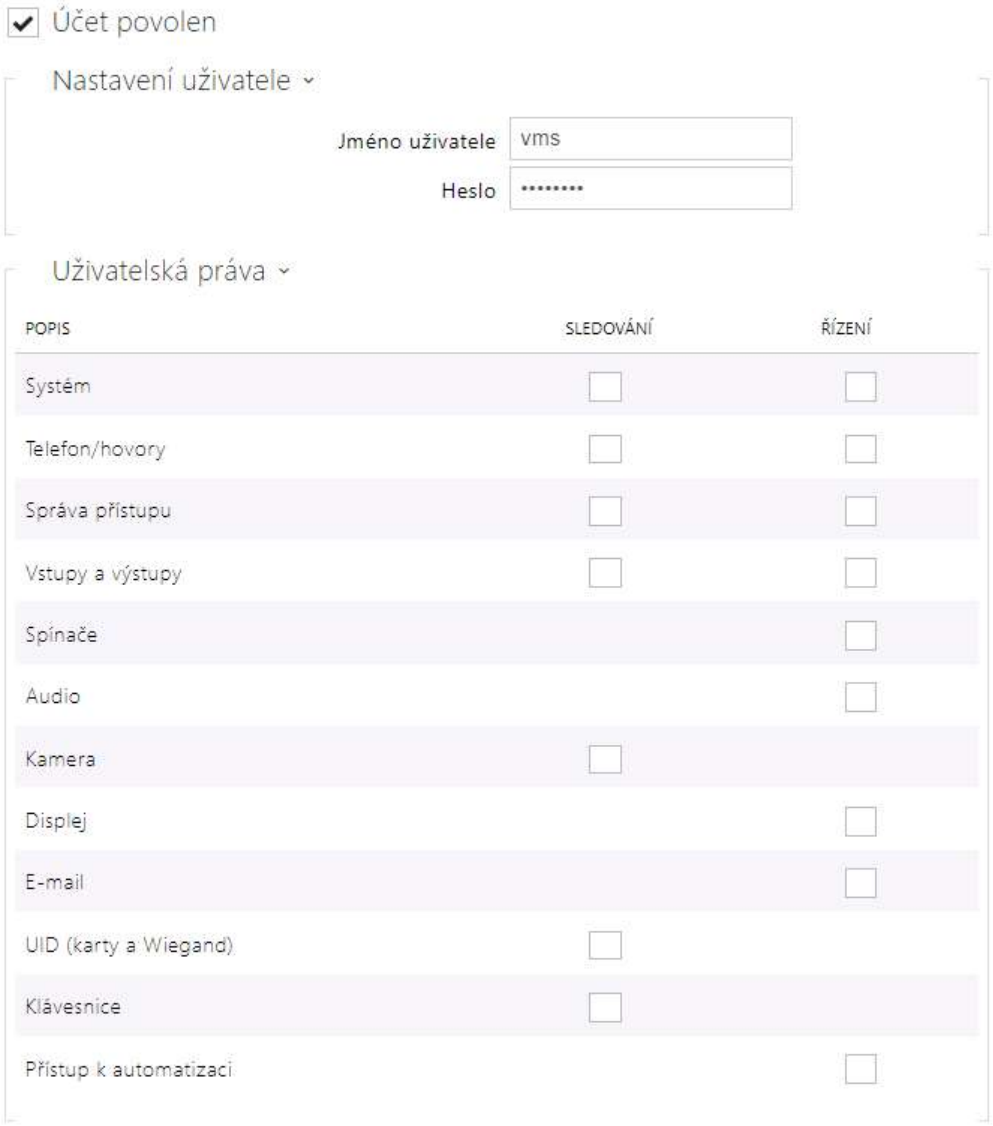
- **Žádná** – služba nevyžaduje žádnou autentizaci. Služba je v tomto případě v lokální síti zcela nechráněná.

- **Basic** – služba vyžaduje autentizaci Basic podle **RFC-2617**. Služba v tomto případě vyžaduje heslo, to je však odesíláno v otevřeném formátu. Doporučujeme tuto volbu kombinovat s **HTTPS** protokolem, pokud je to možné.
- **Digest** – služba vyžaduje autentizaci Digest podle **RFC-2617**. Tato varianta je výchozí a z výše uvedených metod nejbezpečnější.

Pro maximální bezpečnost a odolnost proti zneužití doporučujeme u všech služeb využívat kombinaci **HTTPS + Digest**. V případě, že druhá strana komunikující s interkomem tuto kombinaci nepodporuje, lze konkrétní službě udělit výjimku a úroveň zabezpečení snížit.

4. Uživatelské účty

2N IP interkom umožňuje spravovat až pět uživatelských účtů určených pro přístup ke službám **HTTP API**. Součástí uživatelského účtu je jméno a heslo uživatele a tabulka přístupových práv uživatele k jednotlivým službám **HTTP API**.



Pomocí tabulky přístupových práv lze řídit privilegia uživatelského účtu k jednotlivým službám.

5. Přehled funkcí HTTP API

V tabulce níže je souhrn všech dostupných funkci **HTTP API**. Tabulka obsahuje následující informace:

- absolutní cesta **HTTP** požadavku
- podporované **HTTP** metody
- služba, ve které se funkce nachází
- vyžadovaná uživatelská práva uživatele (v případě, že se využívá autentizace)
- od verze FW 2.35 není využití vybrané funkce podmíněno licencí (tzn. funkce je dostupná bez nutnosti vložení licenčního klíče)

Absolutní cesta	Metoda	Služba	Potřebná uživatelská práva
/api/automation/trigger	GET	Automation	Přístup k automatizaci
/api/accesspoint/blocking/ctrl	GET/POST	Access Control	Správa přístupu – řízení
/api/accesspoint/blocking/status	GET/POST	Access Control	Správa přístupu – sledování
/api/accesspoint/grantaccess	GET/POST	Access Control	Správa přístupu – řízení
/api/audio/test	GET/POST	Audio	Audio – řízení
/api/call/answer	GET/POST	Phone/Call	Telefon/hovory – řízení
/api/call/dial	GET/POST	Phone/Call	Telefon/hovory – řízení
/api/call/hangup	GET/POST	Phone/Call	Telefon/hovory – řízení
/api/call/status	GET/POST	Phone/Call	Telefon/hovory – sledování
/api/camera/caps	GET/POST	Camera	Kamera – sledování
/api/camera/snapshot	GET/POST	Camera	Kamera – sledování
/api/config	GET/POST/PUT	System	Systém – řízení
/api/config/factoryreset	GET/POST	System	Systém – řízení
/api/config/holidays	GET/PUT	System	Systém – řízení
/api/dir/create	PUT	System	Systém – řízení
/api/dir/delete	PUT	System	Systém – řízení
/api/dir/get	POST	System	Systém – řízení
/api/dir/query	POST	System	Systém – řízení
/api/dir/template	GET/POST	System	Systém – řízení
/api/dir/update	PUT	System	Systém – řízení

Absolutní cesta	Metoda	Služba	Potřebná uživatelská práva
/api/display/caps	GET/POST	Display	Displej – řízení
/api/display/image	PUT/DELETE	Display	Displej – řízení
/api/email/send	GET/POST	E-mail	E-mail – řízení
/api/firmware	PUT	System	Systém – řízení
/api/firmware/apply	GET/POST	System	Systém – řízení
/api/firmware/reject	GET/POST	System	Systém – řízení
/api/holidays	GET/PUT	System	Systém – řízení
/api/io/caps	GET/POST	I/O	Vstupy a výstupy – sledování
/api/io/ctrl	GET/POST	I/O	Vstupy a výstupy – řízení
/api/io/status	GET/POST	I/O	Vstupy a výstupy – sledování
/api/lift/grantaccess	GET/POST	Access Control	Správa přístupu – řízení
/api/log/caps	GET/POST	Logging	–
/api/log/pull	GET/POST	Logging	–
/api/log/subscribe	GET/POST	Logging	*
/api/log/unsubscribe	GET/POST	Logging	*
/api/lpr/image	GET/POST	Access Control	Správa přístupu – sledování
/api/lpr/licenseplate	POST	Access Control	Správa přístupu – řízení
/api/mobilekey/config	GET/PUT	Access Control	Správa přístupu – sledování
/api/pcap	GET/POST	System	Systém – řízení
/api/pcap/live	GET/POST	System	Systém – řízení

Absolutní cesta	Metoda	Služba	Potřebná uživatelská práva
/api/pcap/live/stats	GET/POST	System	Systém – řízení
/api/pcap/live/stop	GET/POST	System	Systém – řízení
/api/pcap/restart	GET/POST	System	Systém – řízení
/api/pcap/stop	GET/POST	System	Systém – řízení
/api/phone/calllog	DELETE	Phone/Call	Telefon/hovory – řízení
/api/phone/calllog	GET/POST	Phone/Call	Telefon/hovory – sledování
/api/phone/status	GET/POST	Phone/Call	Telefon/hovory – sledování
/api/switch/caps	GET/POST	Switch	Spínače – sledování
/api/switch/ctrl	GET/POST	Switch	Spínače – řízení
/api/switch/status	GET/POST	Switch	Spínače – řízení
/api/system/caps	GET	System	Systém – sledování
/api/system/info	GET/POST	System	–
/api/system/restart	GET/POST	System	Systém – řízení
/api/system/status	GET/POST	System	Systém – řízení
/api/system/time	GET/POST/ PUT	System	Systém – sledování/řízení
/api/system/time/set	GET/POST	System	Systém – řízení
/api/system/timezone	GET/PUT	System	Systém – sledování/řízení
/api/system/timezone/caps	GET	Syste,	Systém – sledování
/api/cert/ca	GET/PUT/ DELETE	System	Systém – řízení
/api/cert/user	GET/PUT/ DELETE	System	Systém – řízení

V této kapitole dále naleznete:

5.1 api system

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/system**.

5.1.1 api system info

Funkce **/api/system/info** slouží k získání základních informací o zařízení, jako je typ, výrobní číslo, verze firmware apod. Funkce je dostupná na všech typech zařízení bez ohledu na nastavená přístupová práva.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje souhrn informací o zařízení:

Parametr	Popis
devType	Interní identifikátor zařízení.
variant	Název modelu (varianty) zařízení
variantId	Interní číselný identifikátor produktu
customerId	Interní číselný identifikátor
serialNumber	Sériové (výrobní) číslo zařízení
macAddr	Síťový identifikátor zařízení
hwVersion	Verze hardware
swVersion	Verze firmware
buildType	Typ sestavení firmware (alpha, beta, release)
firmwarePackage	Označení FW balíčku
deviceName	Název zařízení nastavení v konfiguračním rozhraní v sekci **Služby / Web Server**

Upozornění

- Od verze 2.33.2 se mění rozsah hodnot pro klíč "buildType", pro oficiální verzi bude hodnota obsahovat řetězec "release". Rozsah hodnot pro klíč "buildType" je do verze 2.32.1 pro oficiální verzi prázdný.

Příklad:

```
GET /api/system/info
{
  "success" : true,
  "result" : {
    "devType" : "2-14-0-0",
    "variant" : "2N IP Verso",
    "variantId" : 14,
    "customerId" : 0,
    "serialNumber" : "00-0000-0005",
    "macAddr" : "FC-1E-B3-00-00-05",
    "hwVersion" : "570v1",
    "swVersion" : "2.35.0.45.0",
    "buildType" : "dev",
    "firmwarePackage" : "verso",
    "deviceName" : "2N IP Verso"
  }
}
```

5.1.2 api system status

Funkce **/api/system/status** vrací aktuální stav interkomu.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení).**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje aktuální stav zařízení:

Parametr	Popis
systemTime	Reálný čas v zařízení v sekundách od 00:00 1.1.1970 (unix time)
upTime	Doba chodu zařízení od posledního restartu v sekundách.

Příklad:

```
GET /api/system/status
{
  "success" : true,
  "result" : {
    "systemTime" : 1418225091,
    "upTime" : 190524
  }
}
```

5.1.3 api system restart

Funkce **/api/system/restart** provede restart interkomu.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby měl uživatel přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/system/restart
{
  "success" : true
}
```

5.1.4 api system caps

Funkce **/api/system/caps** slouží k zasílání informací do **2N® Access Commanderu** o změně seznamu dostupných funkcí zařízení.

Funkce je součástí služby **System API** a v případě použití autentizace je nutné, aby měl uživatel přiřazené privilegium **Systém**.

Pro tuto funkci lze použít metodu **GET**.

Odpověď je ve formátu **application/json** a obsahuje souhrn informací o zařízení:

```
{   "success":true,
    "result":{
     "options":{
         "codecG722":"active,licensed",
         "codecG729":"active",
         "codecL16":"active",
         "audioLoopTest":"active,licensed",
         "noiseDetection":"active,licensed",
         "userSounds":"active,licensed",
         "adaptiveVolume":"active",
         "antiHowling":"active",
         "keyBeep":"active",
         "camera":"active",
         "video":"active",
         "cameraPtz":"active,licensed",
         "motionDetection":"active,licensed",
         "encH264":"active",
         "encH263":"active",
         "encMpeg4":"active",
         "encJpeg":"active",
         "decH264":"active",
         "phone":"active",
         "phoneVideo":"active",
         "phoneVideoOut":"active",
         "sips":"active,licensed",
         "srtp":"active,licensed",
         "callAnswerMode":"active",
         "doorOpenCallback":"active",
         "rtspServer":"active,licensed",
         "rtspClient":"active,licensed",
         "audioMulticast":"active",
         "smtpClient":"active,licensed",
         "ftpClient":"active,licensed",
         "onvif":"active,licensed",
         "snmp":"active,licensed",
         "tr069":"active,licensed",
         "knocker":"active",
         "my2n":"active",
         "informacast":"active,licensed",
         "autoProv":"active,licensed",
         "httpApi":"active,licensed",
         "eap":"active,licensed",
         "eapMd5":"active,licensed",
         "eapTls":"active,licensed",
         "vpn":"active",
         "userVpn":"active",
         "rioManager":"active,licensed",
         "siteChannel":"active",
         "localCalls":"active",
         "switches":"active",
         "advancedSwitches":"active,licensed",
         "switchUserCodes":"active",
```

```
            "securedInput":"active",
            "rexInput":"active",
            "tamperInput":"active",
            "doorSensor":"active",
            "keypad":"active",
            "buttons":"active",
            "liftControl":"active,licensed",
            "limitFailedAccess":"active,licensed",
            "silentAlarm":"active,licensed",
            "scrambleKeypad":"active,licensed",
            "tamperBlockSwitch":"active,licensed",
            "antiPassback":"active,licensed",
            "dir":"active",
            "dirDeputy":"active",
            "dirPhoto":"active",
            "automation":"active,licensed",
            "licDownload":"active",
            "profiles":"active",
            "licensing":"active",
            "accessControl":"active",
            "doorControl":"active",
            "nfc":"active,licensed",
            "vbus":"active",
            "vbusExtenders":"active",
            "cardReader":"active",
            "fpReader":"active",
            "bleReader":"active",
            "wiegand":"active",
            "powerManager":"active",
            "audioInput":"active",
            "lightSensor":"active",
            "irLed":"active",
            "backlight":"active",
            "backlightDayNight":"active",
            "display":"active"
        }

    }


}
```


5.1.5 api system time

Funkce **/api/system/time** slouží k získání o čase zařízení nebo k nastavení času.

Pro tuto funkci lze použít metodu **GET** k získání informací nebo **PUT** k nahrání konfigurace.

Metoda GET

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (sledování)**.

Pro metodu **GET** nejsou definovány žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje reálný čas v zařízení v sekundách od 00:00 1.1.1970 (unix time).

```
GET /api/system/time
{
  "success" : true,
  "result" : {
    "utcTime" : 1639472172,
    "source" : "My2N",
    "automatic" : true,
  }
}
```

Metoda PUT

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

> **Upozornění**
>
> - Doporučujeme používat tento endpoint k nastavení času zařízení jen v případě, že parametr ***Použít čas z internetu*** je vypnutý. V případě, že je tento parametr zapnutý, čas bude přepsán časem z NTP serveru nebo z časové služby My2N.

Parametry požadavku pro metodu **PUT**:

Parametr	Popis
automatic	Automatické získávání času z NTP serveru Při vyplnění hodnoty true se další parametry nezadávají.
utcTime	číslo = unixový čas, min. BuildTime, max. 2147483647
server	adresa NTP serveru (IPv4 adresa nebo doména)

Odpověď pro metodu **PUT** je ve formátu **application/json**.

```
PUT /api/system/time?automatic=0&server=pool.ntp.org&utcTime=1700829813
{
```

```
  "success" : true,
}
```

5.1.6 api system timezone

Funkce **/api/system/timezone** slouží k získání informací o časové zóně zařízení nebo k nastavení časové zóny.

Pro tuto funkci lze použít metodu **GET** k získání informací nebo **PUT** k nahrání konfigurace.

Metoda GET

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (sledování)**.

Pro metodu **GET** nejsou definovány žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje informace, zda je tato zóna nastavena automaticky a jaká časová zóna je nastavena. V případě, že je časová zóna zadaná manuální volbou, je v parametru custom zobrazeno pravidlo, podle kterého byla zóna nastavena.

Příklad

```
GET /api/system/timezone
{
    "success": true,
    "result": {
        "automatic": true,
        "zone": "America/Los Angeles"
    }
}
nebo
{
    "success": true,
    "result": {
        "automatic": false
        "zone": "custom",
        "custom": "UTC-08:00"
    }
}
```

Metoda PUT

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Parametry požadavku pro metodu **PUT**:

Parametr	Popis
automatic	0 nebo 1 Při vyplnění hodnoty true (1) se další parametry nezadávají.
zone	Definuje zónu. Lze zadat buď název zóny nebo "custom" pro umožnění manuálního zadání zóny. Seznam názvů zón je na endpointu 5.1.7 api system timezone caps.
custom	Definuje zónu vlastním zapsáním ve formátu *UTC−08:00, UTC+03:00* apod. Je možné použít pouze při zone="custom".

Příklad

```
PUT /api/system/timezone?automatic=0&zone=custom&custom=UTC-08:00
{
  "success" : true
}
```

5.1.7 api system timezone caps

Funkce **api/system/timezone/caps** vrací seznam dostupných časových zón.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json**.

```
GET /api/system/timezone/caps
{
  "success": true,
  "result": {
    "timezones": [
      "Africa/Abidjan",
      "Africa/Accra",
      "Africa/Bissau",
      "Africa/Monrovia",
      "Africa/Sao_Tome",
      "America/Danmarkshavn",
      "Antarctica/Troll",
      "Atlantic/Canary",
      "Atlantic/Faroe",
      "Atlantic/Madeira",
      "Atlantic/Reykjavik",
      "Etc/GMT",
```

```
      "Etc/UCT",
      "Etc/UTC",
      "Europe/Lisbon",
      "Europe/London",
        ...

    ]
  }
}
```

5.2 api firmware

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/firmware**.

5.2.1 api firmware

Funkce **api/firmware** umožňuje nahrát soubor s firmwarem za účelem upgradování nebo downgradování.

Metody

- PUT

Služby a privilegia

- Služby: Systém API
- Privilegia: Systém – řízení

Požadavek PUT

Požadavek obsahuje soubor ve formátu **multipart/form-data**.

Tabulka 1. Parametry požadavku

Parametr	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
blob-fw	Ano	platný firmwarový binární soubor	-	Soubor s firmwarem

Příklad požadavku PUT

```
http://192.168.1.1/api/firmware
```

Odpověď na PUT

Odpověď je ve formátu **application/json**. Obsahuje klíče **success** a **result**.
Hodnota **result** obsahuje různé klíče popsané v následující tabulce.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
fileId	náhodný identifikátor (8 HEX znaků)	Obsahuje náhodný identifikátor nahraného souboru s firmwarem. Identifikátor musí být použit k potvrzení nahraného firmwaru pomocí funkce **api/firmware/apply** nebo k odmítnutí nahraného firmwaru pomocí funkce **api/firmware/reject**.
version	Řetězec s identifikací verze major.minor.patch.build.id	Obsahuje identifikaci verze nahraného firmwaru.
downgrade	`true` nebo `false`	Tento příznak je `true`, jestliže má nahraný firmware nižší verzi, než je právě v zařízení.
note	Řetězec s náhradními znaky (URL kódování)	Obsahuje aktualizační zprávu pro nahraný firmware (např. varování před velkými změnami).

Příklad odpovědi na PUT


```
{ "success" : true, "result" : { "fileId" : "7d6adf16", "version" : "2.32.4.41.2",
"downgrade" : false, "note" : "EN:\r\nVER=2.20.0\r\nSome changes associated with the
downgrade to a lower version result in a loss of original settings in a certain part
of configuration.\r\n\r\n* All the cards installed in the **Directory \/ Access
cards** menu are moved to the **Directory \/ Users** menu as new users upon firmware
upgrade. Each user is automatically named as !Visitor #n, where n gives the user
number in the list. This change is irreversible upon downgrade.\r\n* Service cards
are now available in the **Hardware \/ Card reader** menu.\r\n* All the user
access ... ... ... \u043F\u0440\u043E\u0444\u0438\u043B\u0435\u043C
\u043F\u043E\u043B\u044C\u0437\u043E\u0432\u0430\u0442\u0435\u043B\u044F.
\r\n\r\n" } }
```


Vrátit se mohou tyto konkrétní chybové kódy:

- `Error code 12`
  - param = "blob-fw"
  - popis = "neplatná hodnota parametru"
  - Nahraný firmware neodpovídá požadavkům (neplatný soubor, firmware pro jiné zařízení…)
- `Error code 19`
  - popis = "verze souboru je nižší než požadovaná minimální verze"
  - Nahraný firmware má nižší verzi, než je pro zařízení povolena.

> **Upozornění**
>
> Zařízení neodpovídá na požadavky na nahrání dalšího firmwaru, když v něm existuje předchozí verze. Nejprve s pomocí funkce **api/firmware/reject** odmítněte předchozí verzi firmwaru a pak nahrajte jinou. Nahraný firmware bude za 5 minut automaticky odmítnut, pokud se nepoužije.

5.2.2 api firmware apply

Funkce **api/firmware/apply** slouží k potvrzení nahraného firmwaru a provedení upgradu/downgradu zařízení.

Metody

- GET
- POST

Skupiny služeb a privilegií

- Služby: System API
- Privilegia: Systém – řízení

Požadavek GET nebo POST

Požadavek obsahuje soubor v **URL**.

Tabulka 1. Parametry požadavku

Parametr	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
fileid	Ano	Identifikátor firmwarového souboru	-	Tento parametr se musí shodovat s identifikátorem aktuálně nahraného firmwaru.

Příklad požadavku GET nebo POST

```
http://192.168.1.1/api/firmware/apply?fileid=7d6adf16
```

Odpověď na GET nebo POST

Odpověď je ve formátu **application/json**. Obsahuje klíč **success**. Je-li success `true`, firmware je použit a zařízení je upgradováno/downgradováno.

Příklad odpovědi na GET nebo POST

```
{ "success" : true }
```

Vrátit se mohou tyto konkrétní chybové kódy:

- `Error code 12`
  - parametr = "fileid"
  - popis = "neplatný parametr"
  - Identifikátor souboru je neplatný (např. obsahuje jiné než hexadecimální znaky).
- `Error code 14`
  - popis = "nový firmware nenalezen"
  - Není nahrán žádný firmwarový soubor s tímto fileid.

5.2.3 api firmware reject

Funkce **api/firmware/reject** slouží k odmítnutí nahraného souboru s firmwarem.

Metody

- GET
- POST

Služby a privilegia

- Služby: System API
- Privilegia: Systém – Řízení

Požadavek PUT

Požadavek obsahuje soubor v **URL**.

Tabulka 1. Parametry požadavku

Parametr	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
fileId	Ano	Identifikátor firmwarového souboru	-	Tento parametr se musí shodovat s identifikátorem aktuálně nahraného firmwaru.

Příklad požadavku GET nebo POST

```
http://192.168.1.1/api/firmware/reject?fileId=7d6adf16
```

Odpověď na GET nebo POST

Odpověď je ve formátu **application/json**. Obsahuje klíč **success**. Je-li success `true`, firmware je odmítnut a s pomocí funkce **api/firmware** je možno nahrát nový soubor.

Příklad odpovědi na GET nebo POST

```
{ "success" : true }
```

Vrátit se mohou tyto konkrétní chybové kódy:

- `Error code 12`
  - parametr = "fileId"
  - popis = "neplatný parametr"
  - Identifikátor souboru je neplatný (např. obsahuje jiné než hexadecimální znaky).
- `Error code 14`
  - popis = "nový firmware nenalezen"
  - Není nahrán žádný firmwarový soubor s tímto fileId.

Upozornění

- Zařízení neodpovídá na požadavky **api/firmware** na nahrání dalšího firmwaru, když v něm existuje předchozí verze. Nejprve s pomocí funkce **api/firmware/reject** odmítněte předchozí verzi firmwaru a pak nahrajte jinou. Nahraný firmware bude za 5 minut automaticky odmítnut, pokud se nepoužije.

5.3 api config

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/config**.

5.3.1 api config

Funkce **/api/config** slouží k uploadu nebo downloadu konfigurace zařízení.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST** pro download konfigurace nebo metodu **PUT** pro upload konfigurace.

Parametry požadavku pro metodu **PUT**:

Parametr	Popis
blob-cfg	Povinný parametr obsahující konfiguraci zařízení (ve formátu XML).

Pro metody GET/POST nejsou definovány žádné parametry.

V případě downloadu konfigurace je odpověď ve formátu **application/xml** a obsahuje kompletní konfigurační soubor zařízení.

Funkce **/api/config** s použitím metody **PUT** provádí upload konfigurace se zpožděním cca 15 s, během tohoto intervalu se zařízení nesmí resetovat ani vypínat.

Příklad:

```
 GET /api/config
<?xml version="1.0" encoding="UTF-8"?>
<!--
        Product name: 2N IP Vario
       Serial number: 08-1860-0035
    Software version: 2.10.0.19.2
    Hardware version: 535v1
  Bootloader version: 2.10.0.19.1
             Display: No
         Card reader: No
-->
<DeviceDatabase Version="4">
<Network>
    <DhcpEnabled>1</DhcpEnabled>
    ...
    ...
```

V případě uploadu konfigurace je odpověď ve formátu **application/json** a neobsahuje žádné další parametry.

Příklad:

```
PUT /api/config
{
  "success" : true
}
```

Upozornění

- Po přechodu na verzi 2.24 dojde ke zrušení pozic uživatelů v adresáři. Při aktualizaci adresáře je nejprve potřeba stáhnout současnou konfiguraci, v této konfiguraci provést požadované změny a poté znovu nahrát.
- Při nedodržení postupu může dojít ke ztrátě dat.

5.3.2 api config factoryreset

Funce **/api/config/factoryreset** nastaví všechny parametry interkomu do výchozího stavu. Tato funkce je shodná se stejnojmennou funkcí webového konfiguračního rozhraní Systém / Údržba – Výchozí nastavení.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
section	ne	`network`	N/A	Použití tohoto parametru vrátí do výchozího stavu také hodnoty všech síťových nastavení (včetně certifikátů). Pokud není parametr zadán, vrátí se do výchozího nastavení veškerá konfigurace vyjma síťových nastavení.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Funkce **/api/config/factoryreset** nastavuje interkom do výchozího stavu se zpožděním cca 15 s, během tohoto intervalu se interkom nesmí resetovat ani vypínat.

Příklad:

```
GET /api/config/factoryreset
{
  "success" : true
}
```

5.3.3 api config holidays

Funkce **/api/config/holidays** slouží k vytváření/nastavování seznamu svátků.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metodu **GET** nebo **PUT**.

Pro metodu **GET** nejsou definovány žádné parametry.

Parametry požadavku pro metodu **PUT**:

Parametr	Popis
blob-json	Povinný parametr obsahující definici svátků (JSON)

Odpověď pro metodu **GET** je ve formátu **application/json** a obsahuje pole svátků. Data jsou ve formátu DD/MM[/YYYY], přičemž rok se uvádí, když svátek platí jen pro zadaný rok.

GET /api/config/holidays

```
{
 "success" : true,
 "result" : { "dates": [ "01\/01", "24\/12", "01\/04\/2018" ] }
```

```
}
```

Formát JSON pro metodu **PUT** je stejný jako pro výsledek metody **GET**.

```
{ "dates": [ "01\/01", "24\/12", "01\/04\/2018" ] }
```

Odpověď pro metodu **PUT** je ve formátu **application/json** a neobsahuje žádné další parametry.

PUT /api/config/holidays

```
{ "success": true
}
```

5.4 api switch

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/switch**.

5.4.1 api switch caps

Funkce **/api/switch/caps** vrací aktuální nastavení a možnosti řízení spínačů. Funkce má volitelný parametr **switch**, který určuje spínač, jehož vlastnosti a nastavení se mají vrátit. Pokud parametr **switch** není uveden, funkce vrací stav všech spínačů.

Funkce je součástí služby **Switch** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Spínače (sledování)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
switch	Volitelný parametr definující číslo spínače (obvykle 1 až 4).

Odpověď je ve formátu **application/json** a obsahuje seznam spínačů (pole **switches**) a jejich aktuální nastavení. V případě použití parametru **switch** obsahuje pole **switches** právě jednu položku.

Parametr	Popis
switch	ID spínače (1 až 4)

Parametr	Popis
enabled	Řízení spínače je povoleno v konfiguračním webovém rozhraní.
mode	Nastavený režim spínače (**monostable**, **bistable**)
switchOnDuration	Doba sepnutí spínače v sekundách (jen pro monostabilní režim)
type	Typ spínače (**normal**, **security**)

Příklad:

```
GET /api/switch/caps
{
  "success" : true,
  "result" : {
    "switches" : [
      {
        "switch" : 1,
        "enabled" : true,
        "mode" : "monostable",
        "switchOnDuration" : 5,
        "type" : "normal"
      },
      {
        "switch" : 2,
        "enabled" : true,
        "mode" : "monostable",
        "switchOnDuration" : 5,
        "type" : "normal"
      },
      {
        "switch" : 3,
        "enabled" : false
      },
      {
        "switch" : 4,
        "enabled" : false
      }
    ]
  }
}
```

5.4.2 api switch status

Funkce **api/switch/status** vrací aktuální stavy spínačů.

Skupiny služeb a privilegií

- Skupina služeb je Switch.
- Skupina privilegií je Spínače – řízení.

Metody

- GET
- POST

Požadavek

Požadavek obsahuje parametry ve formátu URL (nebo **application/x-www-form-urlencoded** při použití POST).

Tabulka 1. Parametry požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
switch	No	celé číslo definující spínač (obvykle 1 až 4)	–	Definuje spínač, jehož stav má být vrácen. Funkce **api/switch/caps** může sloužit ke zjištění počtu spínačů určitého zařízení. Jestliže je tento parametr vynechán, vrátí se stav všech spínačů.

Příklad požadavku

```
URL: https://192.168.1.1/api/switch/status?switch=1
```

Odpověď
Úspěšná odpověď je ve formátu **application/json**. Obsahuje dva JSON klíče `success` a `result`, který obsahuje klíč `switches` (stavové informace jednotlivých spínačů jsou v poli s jedním až čtyřmi prvky).

Tabulka 2. Klíče JSON odpovědi switches

Klíč	Typické vrácené hodnoty	Popis
switch	celé číslo (obvykle 1 až 4)	Definuje, ke kterému spínači se stav vztahuje.
active	`true` nebo `false`	Definuje aktuální stav spínače (`true` – spínač je sepnutý, `false` – spínač je vypnutý).
locked	`true` nebo `false`	Definuje, zda je spínač uzamčený, nebo ne (`true` – spínač je uzamčený ve vypnutém stavu a nedá se ovládat, `false` – spínač je odemčený a dá se normálně ovládat). Uzamčení má přednost před přidržením spínače – tj. když je spínač současně uzamčený a přidržený, je vypnutý a nedá se ovládat.
held	`true` nebo `false`	Definuje, zda je spínač přidržený, nebo ne (`true` – spínač je přidržen v sepnutém stavu a nedá se ovládat, `false` – spínač je uvolněn a dá se normálně ovládat). Uzamčení má přednost před přidržením spínače – tj. když je spínač současně uzamčený a přidržený, je vypnutý a nedá se ovládat.

Příklad odpovědi

```
GET /api/switch/status { "success" : true, "result" : { "switches" : [ { "switch" : 1,
 "active" : true "locked" : false "held" : true }, { "switch" : 2, "active" : false
"locked" : false "held" : false }, { "switch" : 3, "active" : false "locked" : true
"held" : false }, { "switch" : 4, "active" : false "locked" : true "held" : true } ]
} }
```

Mohou se vyskytnout různé chyby (např. chybějící povinný parametr). Chyby se vracejí ve formátu .json s odpovědním kódem 200.

5.4.3 api switch ctrl

Funkce **/api/switch/ctrl** se používá k ovládání spínačů.

Skupiny služeb a privilegií

- Skupina služeb je Switch.
- Skupina privilegií je Spínače (řízení).

Metody

- GET
- POST

Požadavek

Požadavek obsahuje parametry ve formátu URL (nebo **application/x-www-form-urlencoded** při použití POST).

Tabulka 1. Parametry požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
switch	Ano	Celé číslo definující spínač (obvykle 1 až 4)	–	Definuje, který spínač má být ovládán. Funkce **api/switch/caps** může sloužit ke zjištění počtu spínačů určitého zařízení.

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
action	Ano	Řetězec definující příkaz	–	Definuje, který příkaz bude pro spínač použit. Lze použít tyto příkazy: • `on` – sepnutí spínače • `off` – vypnutí spínače • `trigger` – sepnutí monostabilního spínače, překlopení stavu bistabilního spínače • `lock` – uzamčení spínače (uzamčený spínač je vypnutý a nedá se ovládat) • `unlock` – odemknutí spínače (povolení normálního provozu) • `hold` – přidržení sepnutého spínače (přidržený spínač je sepnutý, ale nedá se ovládat), jestliže je spínač uzamčen a zároveň přidržen, je vypnutý • `release` – uvolnění spínače z přidržení (povolení normálního provozu)
response	Ne	Řetězec definující text, který má být vrácen místo standardní odpovědi JSON	–	Zařízení vrátí text uvedený v tomto parametru místo standardní odpovědi JSON.
timeout	Ne (zadává se při použití parametru `action=hold`)	Rozsah v sekundách 1–86 400.	–	Definuje dobu v sekundách, po které se spínač po přijetí příkazu `hold` automaticky opět uvolní.

Příklad požadavku

```
URL: https://192.168.1.1/api/switch/ctrl?switch=4&action=trigger&response=TEST
```

Odpověď

Úspěšná odpověď je ve formátu **application/json** (pokud není v parametru `response` uveden jiný text odpovědi).

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
success	`true` nebo `false`	Jestliže byl příkaz proveden správně, je vrácená hodnota `true`, a jestliže nebylo možno dosáhnout požadovaného stavu, je `false` (např. když je spínač uzamčen a požadovaným stavem byl sepnutý spínač).

Příklad odpovědi

```
{ "success": true }
```

Když je vrácena hodnota `false`, odpověď obsahuje další informace. Chybový kód 14, "akce se nezdařila", znamená, že nebylo možno dosáhnout požadovaného výsledku (např. když je spínač uzamčen a byla požadována akce `action=on`). Příkaz ke změně typu operace (tj. přidržení, uzamčení) bude vždy úspěšný, protože typ operace se dá změnit kdykoli s výjimkou případu, kdy je spínač zakázán (zařízení v takovém případě vrátí na všechny příkazy chybu 14).

5.5 api io

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/io**.

5.5.1 api io caps

Funkce **/api/io/caps** vrací seznam dostupných hardwarových vstupů a výstupů zařízení (portů). Funkce má volitelný parametr **port**, který určuje vstup/výstup, jehož vlastnosti se mají vrátit. Pokud parametr **port** není uveden, funkce vrací seznam všech vstupů a vstupů.

Funkce je součástí služby **I/O** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Vstupy a výstupy – sledování.**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
Port	Volitelný parametr definující identifikátor vstupu nebo výstupu.

Odpověď je ve formátu **application/json** a obsahuje seznam portů (pole **ports**) a jejich aktuální nastavení. V případě použití parametru **port** obsahuje pole **ports** právě jednu položku.

Parametr	Popis
port	Identifikátor vstupu nebo výstupu
type	Typ (**input** – pro digitální vstupy, **output** – pro digitální výstupy)

Příklad:

```
GET /api/io/caps
{
  "success" : true,
  "result" : {
    "ports" : [
      {
        "port" : "relay1",
        "type" : "output"
      },
      {
        "port" : "relay2",
        "type" : "output"
      }
   ]
  }
}
```

5.5.2 api io status

Funkce **/api/io/status** vrací aktuální stav logických vstupů a výstupů zařízení (portů). Funkce má volitelný parametr **port**, který určuje vstup/výstup, jehož stav se má vrátit. Pokud parametr **port** není uveden, funkce vrací stav všech vstupů a výstupů.

Funkce je součástí služby **I/O** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Vstupy a výstupy – sledování.**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
port	Volitelný parametr definující identifikátor vstupu nebo výstupu. Identifikátory dostupných vstupů a výstupů lze získat pomocí funkce **/api/io/caps**.

Odpověď je ve formátu **application/json** a obsahuje seznam portů (pole **ports**) a jejich aktuální stav (položka **state**). V případě použití parametru **port** obsahuje pole **ports** právě jednu položku.

Příklad:

```
 GET /api/io/status
{
  "success" : true,
  "result" : {
    "ports" : [
      {
        "port" : "relay1",
        "state" : 0
      },
      {
        "port" : "relay2",
        "state" : 0
      }
    ]
  }
}
```

5.5.3 api io ctrl

Funkce **/api/io/ctrl** řídí stav logických výstupu zařízení. Funkce má povinný parametr **port**, který určuje řízený výstup a povinný parametr **action** definující akci provedenou nad spínačem (sepnutí, vypnutí).

Funkce je součástí služby **I/O** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Vstupy a výstupy – řízení.**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
port	Povinný parametr definující identifikátor vstupu nebo výstupu. Identifikátory dostupných vstupů a výstupů lze získat pomocí funkce **/api/io/caps**.
action	Povinný parametr definující akci (**on** – sepnutí výstupu, log.1, **off** – vypnutí výstupu, log.0).
response	Nepovinný parametr umožňující modifikovat odpověď interkomu tak, aby neobsahovala JSON zprávu, ale obyčejný text odpovídající tomuto parametru.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/io/ctrl?port=relay1&action=on
{
  "success" : true
}
```

V případě použití parametru **response** odpověď interkomu neobsahuje zprávy **json**, server vrací odpověď typu text/plain se zadaným textem (zadaný text může být prázdný).

Příklad:

```
GET /api/io/ctrl?port=relay1&action=on&response=text
text
```

```
GET /api/io/ctrl?port=relay1&action=on&response=
```

5.6 api phone

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/phone**.

5.6.1 api phone status

Funkce **/api/phone/status** slouží k získání aktuálního stavu SIP účtů zařízení.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – sledování.**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
account	Volitelný parametr definující identifikátor SIP účtu (1 nebo 2 nebo 3 nebo 4). Pokud parametr není uveden, funkce vrací stav všech SIP účtů.

Odpověď je ve formátu **application/json** a obsahuje seznam SIP účtů zařízení (pole **accounts**) a jejich aktuální stav. V případě použití parametru **account** obsahuje pole **accounts** právě jednu položku.

Parametr	Popis
account	Jednoznačný identifikátor SIP účtu (1 nebo 2 nebo 3 nebo 4).
enabled	Signalizuje, zda je SIP účet povolen.
sipNumber	Telefonní číslo SIP účtu.
registrationEnabled	Signalizuje, zda má SIP účet povolenou registraci.
registered	Signalizuje, zda je účet úspěšně zaregistrován u SIP registraru.

Příklad:


```
GET /api/phone/status {
  "success" : true,
  "result" : {
    "accounts" : [
      {
        "account" : 1,
        "accountType" : "general",
        "enabled" : false,
        "sipNumber" : "",
        "registrationEnabled" : false,
        "registered" : false
      },
      {
        "account" : 2,
        "accountType" : "general",
        "enabled" : true,
        "sipNumber" : "1784973567",
        "registrationEnabled" : true,
        "registered" : true,
        "registerTime" : 1721138562
      },
      {
        "account" : 3,
        "accountType" : "local",
        "enabled" : true,
        "sipNumber" : "2NIPStyle-5034500194"
      },
      {
        "account" : 4,
        "accountType" : "msteams",
        "enabled" : true,
        "sipNumber" : "+1784973567",
        "registrationEnabled" : true,
        "registered" : true,
        "registerTime" : 1721138562
      }
    ]
  }
}
```


5.6.2 api phone calllog

Funkce **/api/phone/calllog** umožňuje stahovat nebo mazat všechny nebo vybrané záznamy hovorů.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – sledování.**

Přehled záznamů zobrazuje informace:

- Typ volání
  - Příchozí volání (spojené nebo odmítnuté)
  - Zmeškané volání (příchozí nevyzvednuté)
  - Vyzvednuto jinde (příchozí vyzvednuté na jiném zařízení)
  - Odchozí volání (bez ohledu na výsledek)
  - Zvonkové tlačítko
- Typ kontaktu (nastavení ikony kontaktu)
- ID volaného/volajícího
- Datum a čas volání

Metoda GET nebo POST

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje aktuální stav zařízení:

Parametr	Popis
id	Jednoznačná identifikace záznamu.
callType	Specifikuje typ volání. • incoming • outgoing • missed • voicemail • completedElsewhere • zvonkové tlačítko
devType	Interní identifikátor zařízení.
name	Specifikuje jméno uživatele z telefonního seznamu.
date	Datum záznamu volání.
duration	Definuje dobu volání v sekundách.

Záznamy jsou seřazené od nejnovějšího k nejstaršímu podle absolutního času vytvořeného záznamu.

Upozornění

- Pole je prázdné v případě, že nejsou k dispozici žádné záznamy.

Příklad:

```
{
  "success" : true,
  "result" : {
    "callLog" : [
      {
        "id" : ID,
        "callType" : "incoming",
        "devType" : "2-14-0-0",
        "name" : "Franta Vomáčka",
        "date" : "2027-11-06T12:23:52Z",
        "duration": 1514
      },
      {
        "id" : ID,
        "callType" : "incoming",
        "devType" : "4-13-1-2",
        "name" : "Pepa Vonášek",
        "date" : "2027-12-06T12:23:52Z",
        "duration": 15
      },
      ...
    ]
  }
}
```

Zvonkové tlačítko

```
{
  "success" : true,
  "result" : {
    "callLog" : [
      {
        "id" : ID,
        "callType" : "doorbell",
        "date" : "2027-11-06T12:23:52Z"
      },
      ...
    ]
  }
}
```

Metoda DELETE

Funkce je součástí služby **Phone/Call API** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Phone/Call Access Control**.

Parametry požadavku:

Parametr	Popis
id	Jednoznačná identifikace záznamu, který bude smazán.

Příklad:

```
{
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "id",
    "description" : "record not found"
  }
}
```

5.6.3 api phone config

Funkce **/api/phone/config** slouží ke sledování a kontrolování nastavení SIP účtů.

U této funkce lze použít metodu **GET** pro stažení a **PUT** pro nahrání konfigurace.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – sledování** pro metodu **GET** a **Telefon/hovory – řízení** pro metodu **PUT**.

Metoda GET

Parametry požadavku:

Parametr	Popis
account	Volitelný parametr definující identifikátor SIP účtu (1 nebo 2). Pokud parametr není uveden, funkce vrací stav všech SIP účtů.

Pro metodu **GET** je odpověď ve formátu **application/json** a obsahuje seznam SIP účtů zařízení (pole **accounts**) a jejich aktuální stav. V případě použití specifikování účtu parametrem **account** odpověď obsahuje jen informace o daném účtu.

Upozornění

- Zařízení z bezpečnostních důvodů nevrací heslo při použití metody **GET**.

Příklad:

```
GET /api/phone/config
{
    "success": true,
    "result": {
        "accounts": [
            {
                "account": 1,
                "enabled": false,
                "displayName": "",
                "sipNumber": "",
                "domain": "",
                "domainPort": "",
                "authId": "",
                "proxyAddress": "",
                "proxyPort": "",
                "registrationEnabled": false,
                "registrarAddress": "",
                "registrarPort": "",
                "answerMode": "1"
            },
            {
                "account": 2,
                "enabled": false,
                "displayName": "",
                "sipNumber": "",
                "domain": "",
                "domainPort": "",
                "authId": "",
                "proxyAddress": "",
                "proxyPort": "",
                "registrationEnabled": false,
                "registrarAddress": "",
                "registrarPort": "",
                "answerMode": "1"
            }
        ]
    }
}
```

Metoda PUT

Parametry požadavku:

Parametr	Popis
blob-json	Povinný parametr obsahující konfiguraci SIP účtů (ve formátu JSON).

Pro metodu **PUT** je povinný parametr **blob-json**, který může obsahovat všechny parametry z pole **accounts** ze souboru získaného metodou **GET**. Kromě povinného parametru **account** musí obsahovat minimálně ještě jeden další parametr. Ostatní parametry jsou volitelné. U každého účtu v uploadovaném JSON souboru je možné specifikovat parametr **password** a zadat heslo v otevřené podobě. Tento parametr není z bezpečnostních důvodů součástí odpovědi na metodu **GET**. Odpověď je ve formátu **application/json**. Pokud se při ověřování vyskytne chyba, celý proces skončí neúspěšně a žádný z parametrů nebude použit.

Příklad:

```
PUT /api/phone/config
{
    "success": true,
}
```

Parametry z databáze odpovídají parametrům v JSON souboru následovně:

Parametr z databáze	JSON parametr	Doplňkové informace
Phone.Sip	**account**	Číslování začíná od 1, ne od 0.
Phone.Sip.Enabled	**enabled**	
Phone.Sip.User.DisplayName	**displayName**	
Phone.Sip.User.Id	**sipNumber**	
Phone.Sip.User.AuthId	**authId**	Pokud zůstane parametr prázdný, bude parametr Phone.Sip.User.Id použit místo něj.
Phone.Sip.User.PasswordString	**password**	V otevřené podobě – lze do zařízení nahrát pouze pomocí funkce PUT, ale nelze jej získat pomocí funkce GET.
Phone.Sip.Client.Domain	**domain**	

Parametr z databáze	JSON parametr	Doplňkové informace
Phone.Sip.Client.Port	**domainPort**	
Phone.Sip.Proxy.Address	**proxyAddress**	
Phone.Sip.Proxy.Port	**proxyPort**	
Phone.Sip.Registrar.Enabled	**registrationEnabled**	
Phone.Sip.Registrar.Address	**registrarAddress**	
Phone.Sip.Registrar.Port	**registrarPort**	
Phone.Sip.Misc.AnswerMode	**answerMode**	

5.7 api call

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/call**.

5.7.1 api call status

Funkce **/api/call/status** slouží k získání aktuální stavu probíhajících telefonního hovorů. Funkce vrací seznam probíhajících hovorů a jejich parametry.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – sledování.**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
session	Volitelný parametr obsahující identifikátor hovoru, jehož stav se má vrátit. Pokud parametr není uveden, funkce vrací stav všech probíhajících hovorů.

Odpověď je ve formátu **application/json** a obsahuje seznam probíhajících hovorů (pole **sessions**) a jejich aktuální stav. V případě použití parametru **session** obsahuje pole **sessions** právě jednu položku. Pokud aktuálně neprobíhá žádný hovor, pole **sessions** je prázdné.

Parametr	Popis
session	Identifikátor volání
direction	Směr hovoru (**incoming** – příchozí, **outgoing** – odchozí)
state	Stav hovoru (**connecting**, **ringing**, **connected**)
calls	Jednotlivé směřování hovorů probíhajícího volání. Například při volání telefonního čísla ve skupině se zástupcem kontaktu se vytvoří dva souběžné hovory na dvě různé destinace (**peer**).

Příklad:

```
GET /api/call/status
[
  {
    "calls": [
      {
        "id": 6,
        "peer": "sip:10.0.29.27",
        "state": "ringing"
      },
      {
        "id": 7,
        "peer": "sip:10.0.29.31",
        "state": "ringing"
      }
```

```
    ],
    "direction": "outgoing",
    "session": 4,
    "state": "ringing"
  }
]
```

5.7.2 api call dial

Funkce **/api/call/dial** umožňuje iniciovat nový odchozí hovor na zvolené telefonní číslo nebo sip uri pomocí parametru *number* nebo na jednoho či více uživatelů pomocí parametru *users*. Příkaz smí obsahovat pouze jeden z uvedených parametrů, v opačném případě bude vrácena odpověď s chybovým hlášením.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – řízení.**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
number	Povinný parametr specifikující cílové telefonní číslo nebo sip uri.
users	Seznam uuid (unikátní ID) uživatelů oddělenými čárkami.

Odpověď je ve formátu **application/json** a obsahuje informace o vytvořeném odchozím hovoru.

Parametr	Popis
session	Identifikátor hovoru, který lze použít např. pro sledování hovoru pomocí funkce **/api/call/status**, příp. pro ukončení hovoru funkcí **/api/call/hangup**.

Příklad:

```
GET /api/call/dial?number=sip:1234@10.0.23.194
{
  "success" : true,
  "result" : {
    "session" : 2
  }
}
```

5.7.3 api call answer

Funkce **/api/call/answer** umožňuje vyzvednout probíhající příchozí hovor (ve stavu **ringing**).

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – řízení.**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
session	Identifikátor probíhajícího příchozího hovoru.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/call/answer?session=3
{
  "success" : true
}
```

5.7.4 api call hangup

Funkce **/api/call/hangup** umožňuje ukončit probíhající příchozí nebo odchozí hovor.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – řízení.**

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
session	Identifikátor probíhajícího příchozího nebo odchozího hovoru.
reason	Důvod ukončení hovoru: **"normal"** – běžné ukončení hovoru (výchozí hodnota) **"rejected"** – signalizace odmítnutí hovoru **"busy"** – signalizace obsazení stanice

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/call/hangup?session=4
{
  "success" : true
}
```

5.8 api camera

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/camera**.

5.8.1 api camera caps

Funkce **/api/camera/caps** vrací seznam možných zdrojů videa a variant rozlišení JPEG snímků, které lze stahovat pomocí funkce **/api/camera/snapshot**.

Funkce je součástí služby **Camera** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Kamera – sledování**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje seznam podporovaných rozlišení JPEG snímků (pole **jpegResolution**) a seznam dostupných zdrojů obrazu (pole **sources**), které lze použít v parametrech funkce **/api/camera/snapshot**.

Parametr	Popis
width, height	Rozlišení snímku v pixelech
source	Identifikátor zdroje obrazu

Příklad:

```
GET /api/camera/caps
{
  "success" : true,
  "result" : {
    "jpegResolution" : [
      {
        "width" : 160,
        "height" : 120
      },
      {
        "width" : 176,
        "height" : 144
      },
      {
        "width" : 320,
        "height" : 240
      },
      {
        "width" : 352,
        "height" : 272
      },
      {
        "width" : 352,
        "height" : 288
      },
      {
        "width" : 640,
        "height" : 480
      }
    ],
    "sources" : [
      {
        "source" : "internal"
      },
      {
        "source" : "external"
      }
    ]
  }
}
```

5.8.2 api camera snapshot

Funkce **/api/camera/snapshot** umožňuje stažení obrázku z interní nebo externí IP kamery připojené k interkomu. Pomocí parametrů lze specifikovat zdroj obrázku, rozlišení apod.

Funkce je součástí služby **Camera** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Kamera – sledování**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
width	Povinný parametr specifikující horizontální rozlišení JPEG snímku v pixelech. Rozlišení musí odpovídat jedné z podporovaných variant (viz funkce **api/camera/caps**). Při zadání nepodporované hodnoty nebude požadavek proveden.
height	Povinný parametr specifikující vertikální rozlišení JPEG snímku v pixelech. Rozlišení snímku musí odpovídat jedné z podporovaných variant (viz funkce **api/camera/caps**). Při zadání nepodporované hodnoty nebude požadavek proveden.
source	Volitelný parametr definující zdroj videa (**internal** – interní kamera, **external** – externí IP kamera). Pokud parametr není uveden, je zvolen výchozí zdroj videa uvedený v konfiguračním webovém rozhraní v sekci Hardware / Kamera / Společné nastavení.
fps	Volitelný parametr definující snímkovou frekvenci. Pokud je parametr nastaven na hodnotu >= 1, interkom odesílá s nastavenou snímkovou frekvencí obrázky metodou **http server push.**
time	Volitelný parametr definující čas snímku v paměti vrátníku. Hodnoty time musí být v rozsahu paměti vrátníku, tedy <-30, 0> sekund. Pokud je tento parametr použit společně s parametrem **fps**, je parametr **fps** ignorován a funkce vrátí pouze jeden snímek o maximálním rozlišení 1280 x 960 px (width × height). Pokud je tento parametr použit, vrací funkce snímky o maximálním rozlišení 1280 x 960 px (width × height). Požadavky parametrů **width** a **height** na vyšší hodnoty budou ignorovány.

Odpověď je ve formátu **image/jpeg** příp. **multipart/x-mixed-replace** (pro fps >= 1). V případě chybných parametrů požadavku, funkce vrací informaci ve formátu **application/json**.

Příklad:

```
GET /api/camera/snapshot?width=640&height=480&source=internal

# následující příkaz vrátí snímek zachycený 5 sekund před voláním funkce
GET /api/camera/snapshot?width=640&height=480&source=internal&time=-5
```

5.9 api display

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/display**.

5.9.1 api display caps

Funkce **/api/display/caps** vrací seznam displejů v zařízení a jejich vlastnosti. Funkci lze použít pro detekci displeje a získání jeho rozlišení.

Funkce je součástí služby **Displej** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Displej – řízení**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje seznam dostupný displejů (pole **displays**).

Parametr	Popis
display	Identifikátor displeje
resolution	Rozlišení displeje v pixelech

Příklad:

```
GET /api/display/caps
{
  "success" : true,
  "result" : {
    "displays" : [
      {
        "display" : "internal",
        "resolution" : {
          "width" : 320,
          "height" : 240
        }
      }
    ]
  }
}
```

5.9.2 api display image

2N® IP Style

Funkce **/api/display/image** umožňuje modifikovat obsah zobrazovaný na displeji zařízení. Umožňuje nahrát obrázek, příp. nahraný obrázek z displeje odstranit. JPEG obrázky s progresivní kvalitou nejsou podporovány.

Funkce je součástí služby **Displej** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Displej (řízení)**.

Pro tuto funkci lze použít metody **PUT** nebo **DELETE**. Metoda **PUT** slouží k uploadu obrázku na displej. Metoda **DELETE** slouží k odstranění dříve uploadovaného obrázku z displeje.

Metoda PUT

Parametry požadavku:

Parametr	Popis
blob-image	Povinný parametr obsahující obrázek ve formátu JPEG, PNG s rozlišením daného displeje 1280 x 800 pixelů (viz funkce **/api/display/caps**). Parametr se uplatní pouze v případě metody **PUT**. Progresivní JPEG obrázky nejsou podporovány.
duration	Volitelný parametr. Doba zobrazení obrázku. Parametr se nastavuje v milisekundách.

Obrázek lze zobrazit dvěma způsoby jako notifikace nebo overlay. Notifikace se zobrazuje na předem definovanou dobu, po které automaticky zmizí. Overlay zůstává na displeji do té doby, než je nahrazena jiným obrázkem nebo uživatel sám obrázek odstraní.

Pokud HTTP požadavek neobsahuje volitelný parametr, jedná se o zobrazení v režimu overlay, tj. obrázek nahraný na neurčitou dobu. Pokud je volitelný parametr uveden, obrázek se zobrazí jako notifikace, která se ukončí po uplynutí nastavené doby. Zobrazení notifikace je možné předčasně ukončit dotykem displeje.

Nahrává-li se obrázek poprvé, přenáší se z hlavní jednotky na displej přes interní sběrnici (což může chvíli trvat). V paměti displeje se dá uložit i několik obrázků, a pokud se někdy takto uložené obrázky budou znovu posílat do zařízení, nebude je již třeba přenášet přes interní sběrnici. Místo toho je bude možno ihned zobrazit na displeji z paměti.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Parametry obrázku:

Model	Rozměry obrázku	Podporované formáty
2N® IP Style	1280 x 800 pixelů	normální JPEG (doporučeno), PNG

Upozornění

- Podporovaný JPEG formát je JPEG Baseline (neprogresivní kódování).

Příklad:

```
PUT api/display/image&duration=30000
{
  "success": true
}
```

Metoda DELETE

Parametr	Popis
display	Povinný identifikátor displeje. Hodnota je nastavena v parametru Hardware / Rozšiřující moduly / Jméno modulu. Případně lze zjistit hodnotu pomocí funkce **/api/display/caps**.

Příklad:

```
DELETE /api/display/image
{
  "success": true
}
```

2N® IP Verso

Funkce **/api/display/image** umožňuje modifikovat obsah zobrazovaný na displeji zařízení. Umožňuje nahrát, příp. nahraný obrázek z displeje odstranit. Progresivní JPEG obrázky nejsou podporovány.

Funkce je součástí služby **Displej** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Displej (řízení)**.

Pro tuto funkci lze použít metody **PUT** nebo **DELETE**. Metoda **PUT** slouží k uploadu obrázku na displej. Metoda **DELETE** slouží k odstranění dříve uploadovaného obrázku z displeje.

Metoda PUT

Parametry požadavku:

Parametr	Popis
display	Povinný identifikátor displeje. Hodnota je nastavena v parametru Hardware / Rozšiřující moduly / Jméno modulu. Případně lze hodnotu zjistit pomocí funkce **/api/display/caps**.
blob-image	Povinný parametr obsahující obrázek ve formátu JPEG, BMP, PNG s rozlišením daného displeje 320 x 214 pixelů (viz funkce **/api/display/caps**). Parametr se uplatní pouze v případě metody **PUT**. Požadavek smí obsahovat pouze jeden z parametrů, blob-image nebo blob-video. Progresivní JPEG obrázky nejsou podporovány.

Parametr	Popis
blob-video	Povinný parametr obsahující video ve formátu MPEG4 / H264, max. délka 60 s, fps max. 15 nahraný v rozlišení 320 x 214 pixelů.Požadavek smí obsahovat pouze jeden z parametrů, blob-image nebo blob-video.
duration	Volitelný parametr. Doba zobrazení obrázku/přehrávání videa. Parametr se nastavuje v milisekundách.
repeat	Volitelný parametr. Počet opakování přehrání videa. Parametr se týká pouze videa.

Obrázek lze zobrazit dvěma způsoby jako notifikace nebo overlay. Notifikace se zobrazuje na předem definovanou dobu, po které automaticky zmizí. Overlay zůstává na displeji do té doby, než je nahrazena jiným obrázkem nebo uživatel sám obrázek odstraní.

Pokud HTTP požadavek neobsahuje ani jeden z výše uvedených parametrů, jedná se o zobrazení v režimu overlay tj. obrázek nahraný na neurčitou dobu. Pokud jsou uvedeny oba volitelné parametry, zobrazení notifikace se ukončí událostí, která nastane dříve. Zobrazení notifikace je možné předčasně ukončit dotykem displeje.

Nahrává-li se obrázek poprvé, přenáší se z hlavní jednotky na displej přes interní sběrnici (což může chvíli trvat). V paměti displeje se dá uložit i několik obrázků, a pokud se někdy takto uložené obrázky budou znovu posílat do zařízení, nebude je již třeba přenášet přes interní sběrnici. Místo toho je bude možno ihned zobrazit na displeji z paměti.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Parametry obrázku:

Model	Rozměry obrázku	Podporované formáty
2N® IP Verso	320 x 214 pixelů	normální JPEG (doporučeno), BMP, PNG

Upozornění

- Podporovaný JPEG formát je JPEG Baseline (neprogresivní kódování).

Příklad:

```
  api/display/image?display=ext1&duration=30000
{
  "success" : true
}
```

Parametry videa:

Model	Rozměry videa	Podporované formáty
2N® IP Verso	320 x 214 pixelů	MPEG4 / H264: Baseline profil, level max. 5.2

Metoda DELETE

Parametr	Popis
display	Povinný identifikátor displeje. Hodnota je nastavena v parametru Hardware / Rozšiřující moduly / Jméno modulu. Případně lze zjistit hodnotu pomocí funkce **/api/display/caps**.

Příklad:

```
DELETE /api/display/image
{
  "success" : true
}
```

2N® IP Vario

Funkce **/api/display/image** umožňuje modifikovat obsah zobrazovaný na displeji zařízení. Umožňuje nahrát, příp. nahraný obrázek z displeje odstranit.

Funkce je součástí služby **Displej** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Displej (řízení)**.

Pro tuto funkci lze použít metody **PUT** nebo **DELETE**. Metoda **PUT** slouží k uploadu obrázku na displej. Metoda **DELETE** slouží k odstranění dříve uploadovaného obrázku z displeje.

Parametry požadavku:

Parametr	Popis
display	Povinný identifikátor displeje (**internal**)
blob-image	Povinný parametr obsahující obrázek v podporovaném formátu s rozlišením daného displeje (viz funkce **/api/display/caps**). Parametr se uplatní pouze v případě metody **PUT**.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Parametry obrázku:

Model	Rozměry obrázku	Podporované formáty
2N® IP Vario	320 x 240 pixelů	JPEG (doporučeno), GIF, BMP

Upozornění

Podporovaný JPEG formát je JPEG Baseline (neprogresivní kódování).

5.9.2.1 Příklady api display image

Uvedené příklady slouží jako vzor dat zasílaných z řídící aplikace na displej **2N® IP Verso** a **2N® IP Vario**.

Obrázek lze zobrazit dvěma způsoby jako notifikace nebo overlay. Oba způsoby jsou platné pouze pro model **2N® IP Verso**, u modelu **2N® IP Vario** lze obrázek zobrazit pouze přes notifikaci. Notifikace se zobrazuje na předem definovanou dobu, po které automaticky zmizí. Overlay zůstává na displeji do té doby, než je nahrazen jiným obrázkem, nebo uživatel sám obrázek odstraní.

Parameter *__duration__* udává dobu zobrazení obrázku/videa v ms.
Parametr *__repeat__* udává počet opakování videa a u obrázku se ignoruje.

Pokud HTTP požadavek neobsahuje ani jeden z výše uvedených parametrů, jedná se o zobrazení v režimu overlay. Tj. obrázek nahraný na neurčitou dobu. Pokud jsou uvedeny oba parametry, zobrazení se ukončí událostí, která nastane dříve. e

Nahrání obrázku na displej 2N® IP Verso a 2N® IP Vario

Poznámka

Každý z modelů podporuje jiné rozlišení obrázků.

Model	Rozměry obrázku	Podporované formáty
2N® IP Verso	214 x 240 pixelů	JPEG (doporučeno), BMP, PNG
2N® IP Vario	320 x 240 pixelů	JPEG (doporučeno), GIF, BMP

URL požadavku: https://10.27.24.15/api/display/image?display=ext1

- Metoda požadavku: PUT
- Vzdálená adresa: 10.27.24.15:443
- Stavový kód: 200 OK
- Verze: HTTP/1.1

Hlavičky odpovědi (95 B)

- Server: HIP2.22.0.31.1
- Content-Type: application/json
- Content-Length: 24

Hlavičky požadavku (494 B)

- Host: 10.27.24.15
- User-Agent: Mozilla/5.0 (Windows NT 6.1; W…) Gecko/20100101 Firefox/56.0
- Accept: */*
- Accept-Language: cs,en-US;q=0.7,en;q=0.3
- Accept-Encoding: gzip, deflate, br
- Referer: https://10.27.24.15/apitest.html
- Content-Length: 1325
- Content-Type: multipart/form-data; boundary=…----------258852674219952
- Cookie: _ga=GA1.1.375392382.1496656977…id=GA1.1.638680516.1507547865
- Connection: keep-alive

Řetězec dotazu

- display: ext1
- duration: 20
- repeat:3

Obsah požadavku

----------258852674219952

Content-Disposition: form-data; name="blob-image"; filename="picture.png"

Content-Type: image/png


```
‰PNG
IHDR   Ö   đ   ót
R   sRGB ®Îé   gAMA  ±Źüa       pHYs  Ă ĂÇo¨d  üIDATx^íŘŃqŰ8@Ń0•,\0Şq3)Ć»Ž4Y,$Ť'Ô
Í¬ÎůÔČ\ƒ˘ż}
@J,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$
HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$Hë:    ľż}ű íýţáIžy/ţ@,ŚIP,±×L�‘ ±Ó\–őŰ^b;    Žkż˙řůëÓ©í–
f)ËýBÎLđçŹď÷mŢł.jĚb¶v?˜m|;éJđBÎKđ0‹›a÷řţ/›dY¸Z$Á9+Áá[śCC/Ăů
´)éż6ĆŃŘĚ8[\quŽNď%ÁŘ©żoŚÂ˝ŚÍöu6–×mcšŤ%x!§żŽlĚ¶˙°¦I¸Ë5céw¬“`ě©    •}
š%¸Íi7še`»Îľ"Á9;ÁáÚ˜%¸ÍbwúHI_^Č“ś–ŕq|7Ëí–ŕ‹:+Áť ďĎÁŮög±;ő ţ?8)ÁÍŻâř‹    ÝćfřKŘ_'Á
Ř÷x9FÇYL¦Ó«ÝĚĆÇ÷â©ž�ŕâ|Z?ž–ŰśĹlşşââ(\M†KJđBÎú–848µ|x~–Á=r«ƒ%Śť•ŕĽŚ·÷Ĺd±˙ÇY<>ÝŘů‘x|
5žęĽ?Ń¸ďöWŢR‰fâ#ß”`ěÜŻJ,ń’    îĂ4^1Áá]Y,µ–Jpőš0
Ě˝R,;ú!
Ř
{
Ą
Wo
ĚŔ
kxÉ×®D,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$H
L,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$H
L,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$H
L,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$H
L,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,Ä$HL,¤>>
ţ+uKéż<:–    IEND®B`,
----------258852674219952--
```

Nahrání videa na displej 2N® IP Verso

Poznámka

Model	Rozměry videa	Podporované formáty
2N® IP Verso	214 x 240 pixelů	MPEG4 / H264: Baseline profil, level max. 5.2

URL požadavku: https://10.27.24.15/api/display/image?display=ext1&duration=20&repeat=3

- Metoda požadavku: PUT
- Vzdálená adresa: 10.27.24.15:443
- Stavový kód: 200 OK
- Verze: HTTP/1.1

Hlavičky odpovědi (95 B)

- Server: HIP2.22.0.31.1
- Content-Type: application/json
- Content-Length: 24

Hlavičky požadavku (516 B)

- Host: 10.27.24.15
- User-Agent: Mozilla/5.0 (Windows NT 6.1; W…) Gecko/20100101 Firefox/56.0
- Accept: */*
- Accept-Language: cs,en-US;q=0.7,en;q=0.3
- Accept-Encoding: gzip, deflate, br
- Referer: https://10.27.24.15/apitest.html
- Content-Length: 943815
- Content-Type: multipart/form-data; boundary=---------------------------14948718218673
- Cookie: _ga=GA1.1.375392382.1496656977…id=GA1.1.638680516.1507547865
- Connection: keep-alive

Obsah požadavku

----------14948718218673

Content-Disposition: form-data; name="blob-video"; filename="2N_intro.mp4"

Content-Type: video/mp4

```
 ftypmp42  isomiso2avc1mp41   free O!mdat  ·¨łÜEé˝ćŮH·-,Ř Ů#îďx264 - core 148 r2708
86b7198 - H.264/MPEG-4 AVC codec - Copyleft 2003-2016 - http://www.videolan.org/
x264.html - options: cabac=0 ref=2 deblock=0:0:0 analyse=0x1:0x111 me=hex subme=6
psy=1 psy_rd=1.00:0.00 mixed_ref=1 me_range=16 chroma_me=1 trellis=1 8x8dct=0 cqm=0
deadzone=21,11 fast_pskip=1 chroma_qp_offset=-2 threads=6 lookahead_threads=1
sliced_threads=0 nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0
bframes=0 weightp=0 keyint=150 keyint_min=15 scenecut=40 intra_refresh=0
rc_lookahead=30 rc=crf mbtree=1 crf=22.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4
vbv_maxrate=20000 vbv_bufsize=25000 crf_max=0.0 nal_hrd=none filler=0 ip_ratio=1.40
aq=1:1.00 €  őeˆ„    ®C‡Ü!q@ CzˇNbyRćŞŇÔ~ˆ$]´í·ŢFb0~ůh¨M=>?
„´mÍßŹ¨âµWjŻ«¸ĄzŹ”Ä¨OŹ”#ŚĎţŢÜ~éÂ6ŰËÓăďŞ üP•?¨n?é¦žš{–ü7ŢĘ«“b»ž¶Ť˜G
LZ¨¨.˜üß¨ÓřŽđý×\Ů’žjo¸…z˝x¨Ők&+ý¨ťřG¨kü¸3 ˆŔ|p«^ů¨¸˝UzÖż¨¨Xął€ž¤ŔŁŞ¨€pŔ!>lÍ�ŹůMfÓoďľ_„0„)
ćÎlŕő,ňË¨ţź?_?Ăţ|Ź¨ř{>¨řÎl&/Íźţ÷đť~»ŔżT<vW
Íź/}ţ Ď    e‡ŕ•ť¶ß¨Őy9čšziý–$
‰¨ů
_Á
ż
+
¨ř
{]
Ď
]u×]wÜ
Ŕ
"ý
ż9¨ĎČ§ű¨ü$lŮ7ęzë®şë®şë®úë®şë®şë®şë®şë®şë®ż¨ŐxxCîş‰®şë®şë®:O¨äőy¨ŻŽŠů{�Osóţ«ă‘u]u×S·,
\7ťĽ·p°Ó–X    ]`ű§Äńęľƒđüřţš§/Źó¨ Giż...
```

----------14948718218673--

5.10 api log

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/log**.

- 5.10.1 api log caps
- 5.10.2 api log subscribe
- 5.10.3 api log unsubscribe
- 5.10.4 api log pull

5.10.1 api log caps

Funkce **/api/log/caps** vrací seznam typů podporovaných událostí, které se na daném zařízení zaznamenávají. Vrácený seznam je v závislosti na zařízení podmnožinou kompletního seznamu typů událostí uvedeného v následující tabulce:

Typ události	Popis	Parametry	
		Stálé	Podmíněné
AccessBlocked	Signalizuje blokování autentizace uživatele, zónového kódu, odchodového tlačítka REX a přístupu pomocí registrační značky vozidla na přístupovém bodu.	"ap, state"	
AccessLimited	Událost, která nastane po zadání 5 neúspěšných pokusů o autentizaci uživatele (karta, kód, otisk prstu). Přístupový modul bude zablokován po dobu 30 sekund i v případě, že následná autentizace by byla správná. Signalizuje odmítnutí zadaného uživatele.	"ap, type, state"	
AccessTaken	Při přiložení karty v Anti-passback oblasti.	"ap, session"	
ApiAccessRequested	Událost, kdy byl zaslán požadavek na /api/accesspoint/grantaccess s výsledkem "success" : true.	"ap, valid"	"session, uuid"
AudioLoopTest	Signalizuje provedení automatického audio loop testu a jeho výsledek.	"result"	
CallSessionStateChanged	Událost popisující směr, stav hovoru, adresu, číslo vytvořené session a kolikátý hovor se generoval.	"session, state"	"originator, info"
CallStateChanged	Při změně stavu hovoru (ringing, connected, terminated) indikuje i směr (příchozí, odchozí) a identifikaci protistrany nebo účtu SIP.	"direction, state, peer, session, call"	"reason, device, sipAccount, sipCallId "
CapabilitiesChanged	Signalizuje změnu dostupných funkcí.		

Typ události	Popis	Parametry	
		Stálé	Podmíněné
CardHeld	Signalizuje přiložení RFID karty ke čtečce delší než 4 s.	"reader, uid, valid"	"ap, session, direction, uuid"
CardEntered	Signalizuje přiložení RFID karty ke čtečce.	"reader, uid, valid"	"ap, session, direction, uuid"
CodeEntered	Signalizuje zadání kódu uživatelem pomocí numerické klávesnice.	"code, valid "	"ap, session, direction, input, type, uuid, reason"
ConfigurationChanged	Změna nastavení konfigurace zařízení.		
DeviceState	Systémová událost generovaná při změnách stavu zařízení.	"state"	
DisplayTouched	Signalizuje dotyk na displeji.	"x, y, dx, dy"	
DirectoryChanged	Změna v adresáři.	"series"	"timestamp"
DirectorySaved	Uložení změny v adresáři.	"series"	"timestamp"
DoorOpenTooLong	Signalizuje dlouhé otevření dveří, resp. nezavření dveří do nastavené doby.	"state"	
DoorStateChanged	Signalizuje změnu stavu dveří.	"state"	
DtmfEntered	Příjem DTMF kódu v hovoru nebo lokálně mimo hovor.	"code, call, valid"	"type, uuid"

Typ události	Popis	Parametry	
		Stálé	Podmíněné
DtmfPressed	Zadání DTMF kódu v hovoru nebo lokálně mimo hovor.	"code, call valid"	
DtmfSent	Odeslání FTMF kódu v hovoru nebo lokálně mimo hovor.	"code"	"call"
ExternalCameraStateChanged	Signalizuje změnu stavu připojené externí kamery.	"state"	"id, reason"
ErrorStateChanged	Signalizuje změnu chybného stavu LiftIP 2.0.		"in, state, reason"
FingerEntered	Signalizuje přiložení prstu k biometrické čtečce.	"valid"	"ap, session, direction, uuid"
FingerEnrollState	Přiložení prstu na čtečku pro nahrání otisku uživatele.	"session, state"	
HardwareChanged	Změna připojení rozšiřujících modulů.	"reason, class, id"	"info, config, state, categories"
CheckingCall	Zobrazuje detaily uskutečněného kontrolního volání.		"action"
InputChanged	Signalizuje změnu stavu logického vstupu.	"port, state"	
KeyPressed	Signalizuje stisk tlačítka rychlé volby, klávesy numerické klávesnice, dotyk na displeji nebo stisk tlačítka zahajujícího Bluetooth autentizaci.	"key"	
KeyReleased	Signalizuje uvolnění tlačítka rychlé volby nebo klávesy numerické klávesnice.	"key"	

Typ události	Popis	Parametry	
		Stálé	Podmíněné
LicensePlateRecognized	Signalizuje rozpoznání registrační značky vozidla s platnými právy pro přístup.	"ap, licensePlate, valid"	"session, uuid"
LiftConfigChanged	Změna nastavení řízení výtahu.	"hash"	
LiftErrorStateChanged			"in, state, reason"
LiftFloorsEnabled	Přístup na patro pomocí výtahu.	"floors"	"uuid, session"
LiftStatusChanged	Detekce připojení/odpojení Lift Control modulu.	"module, ready"	
LoginBlocked	Signalizuje dočasné zablokování přístupu k webovému rozhraní.	"address"	
MobKeyEntered	Signalizuje autorizaci pomocí bluetooth čtečky.	"action, authid, valid"	"ap, session, direction, uuid"
MotionDetected	Signalizuje detekci pohybu pomocí kamery.	ID = odpovídá číslu profilu detekce pohybu ve webovém rozhraní "state"	
NoiseDetected	Signalizuje detekci zvýšené hladiny hluku	pouze modely vybavené mikrofonem nebo mikrofonním vstupem "state"	

Typ události	Popis	Parametry	
		Stálé	Podmíněné
OutputChanged	Signalizuje změnu stavu logického výstupu.	"port, state"	
PairingStateChanged	Signalizuje párování s bluetooth rozhraním.	"state, authId"	
RescueStateChanged	Signalizuje změnu stavu režimu vyproštění.		"state, reason"
RegistrationStateChanged	Změna stavu registrace k SIP proxy.	"sipAccount, state"	"reason"
RexActivated	Signalizuje aktivaci odchodového tlačítka REX.	"ap, session, valid"	"reason"
SilentAlarm	Signalizuje aktivaci tichého alarmu.	"ap, session, name"	"uuid"
SwitchesBlocked	Signalizuje zablokování zámků tamper spínačem.	"state"	
SwitchOperationChanged	Změna fungování spínače (signalizuje stav uzamčení nebo přidržení spínače, nastartování i restartování časovače nebo jeho ukončení – přechodu do trvalého přidržení).	"switch"	"enabled, locked, held, hold_timeout, originator"
SwitchStateChanged	Signalizuje změnu stavu spínače 1–4	"switch, state"	"ap, session, originator, call, peer, device"
TamperSwitchActivated	Signalizuje aktivaci ochranného spínače.	"state"	

Typ události	Popis	Parametry	
		Stálé	Podmíněné
UnauthorizedDoorOpen	Signalizuje neautorizované otevření dveří.	pouze modely vybavené digitálními vstupy "state"	
UserActionActivated	Signalizuje změnu stavu vstupu, který je nakonfigurován na funkci Spouštěče uživatelské akce.	"id, state"	
UserAuthenticated	Signalizuje autentizaci uživatele a následné otevření dveří.	"ap, session, name"	"uuid, apbBroken"
UserRejected	Signalizuje odmítnutí autorizace uživatele.	"ap, session, name"	"uuid, reason"
VirtualInput	Změna virtuálního vstupu.	"port, state"	
VirtualOutput	Změna virtuálního výstupu.	"port, state"	
WaveKeyActivated	Aktivace Bluetooth autentizace.	"type"	

Funkce je součástí služby **Logging** a pro provedení funkce nejsou potřeba žádná zvláštní privilegia uživatele.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json**:

Parametr	Typ	Popis
events	array	Pole řetězců obsahující seznam podporovaných typů událostí.

Příklad:

```
GET /api/log/caps
{
  "success" : true,
  "result" : {
    "events" : [
      "KeyPressed",
      "KeyReleased",
      "InputChanged",
      "OutputChanged",
      "CardEntered",
      "CallStateChanged",
      "AudioLoopTest",
      "CodeEntered",
      "DeviceState",
      "RegistrationStateChanged"
    ]
  }
}
```

5.10.2 api log subscribe

Funkce **/api/log/subscribe** vytvoří kanál pro odběr událostí (subscription) a vrací unikátní identifikátor, který se použije při následném volání funkcí **/api/log/pull**, příp. **/api/log/unsubscribe**.

Každý kanál pro odběr událostí obsahuje vlastní frontu událostí. Do fronty kanálu jsou ukládány všechny nové události, které odpovídají filtru kanálu (parametr **filter**). Události z fronty kanálu lze číst pomocí funkce **/api/log/pull**.

Současně zařízení udržují v interní paměti frontu historie událostí (posledních 10000 událostí). Po restartu zařízení je tato fronta historie vždy prázdná.

Pomocí parametru **include** lze specifikovat, zda fronta kanálu bude na počátku prázdná (tj. budou do ní zapsány pouze nové události, které vzniknou po vytvoření kanálu), příp. zda má být jednorázově naplněna událostmi z části nebo celé zaznamenané historie událostí.

Pomocí parametru **duration** lze specifikovat životnost kanálu v případě, že se k němu nepřistupuje pomocí funkce **/api/log/pull**. Po nastavené době bude nepoužívaný kanál automaticky uzavřen, jako by byla použita funkce **/api/log/unsubscribe**.

Funkce je součástí služby **Logging** a v případě použití autentizace je nutné pro některé události nastavit privilegia uživatele podle tabulky níže. Do fronty kanálu nebudou zařazovány události, pro které autentizovaný uživatel nemá požadovaná privilegia.

Tabulka událostí:

Dostupné události	Vyžadovaná privilegia uživatele
KeyPressed	Sledování – Klávesnice
KeyReleased	Sledování – Klávesnice
CodeEntered	Sledování – Klávesnice
TamperSwitchActivated	žádná
UnauthorizedDoorOpen	žádná
DoorOpenTooLong	žádná
LoginBlocked	žádná
SilentAlarm	žádná
DoorStateChanged	žádná
DeviceState	žádná
AudioLoopTest	žádná
MotionDetected	žádná
NoiseDetected	žádná
HardwareChanged	žádná
FingerEnrollState	žádná
LiftStatusChanged	žádná
LiftFloorsEnabled	žádná
LiftConfigChanged	žádná
CapabilitiesChanged	žádná
ConfigurationChanged	žádná
ExtCameraStateChanged	žádná
RescueStateChanged	žádná

Dostupné události	Vyžadovaná privilegia uživatele
ErrorStateChanged	žádná
LiftCheckingCall	žádná
DtmfSent	žádná
RexActivated	žádná
AccessBlocked	žádná
AccessTaken	žádná
AccessLimited	žádná
DisplayTouched	žádná
DtmfPressed	žádná
SwitchesBlocked	žádná
CardEntered	Sledování – UID (karty/wiegand)
CardHeld	Sledování – UID (karty/wiegand)
DtmfEntered	Sledování – UID (karty/wiegand)
PairingStateChanged	Sledování – UID (karty/wiegand)
MobKeyEntered	Sledování – UID (karty/wiegand)
WaveKeyEntered	Sledování – UID (karty/wiegand)
FingerEntered	Sledování – UID (karty/wiegand)
UserAuthenticated	Sledování – UID (karty/wiegand)
UserRejected	Sledování – UID (karty/wiegand)
ApiAccessRequested	Sledování – Správa přístupu
LicensePlateRecognized	Sledování – Správa přístupu

Dostupné události	Vyžadovaná privilegia uživatele
CallStateChanged	Sledování – Telefon/Hovory
CallSessionStateChanged	Sledování – Telefon/Hovory
RegistrationStateChanged	Sledování – Telefon/Hovory
InputChanged	Sledování – Vstupy a výstupy
OutputChanged	Sledování – Vstupy a výstupy
VirtualInput	Sledování – vstupy a výstupy
VirtualOutput	Sledování – Vstupy a výstupy
SwitchStateChanged	Sledování – vstupy a výstupy
SwitchOperationChanged	Sledování – vstupy a výstupy
UserActionActivated	Sledování – Vstupy a výstupy
DirectoryChanged	Sledování – Systém
DirectorySaved	Sledování – Systém

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Typ	Povinný	Výchozí hodnota	Popis
include	string	Ne	new	Určuje, zda má být fronta událostí kanálu naplněna položkami z historie: **new** – pouze nové události, které vzniknou až po vytvoření kanálu **all** – všechny dosud zaznamenané událost včetně těch, které vzniknou až po vytvoření kanálu **–t** – všechny dosud zaznamenané události za posledních **t** sekund včetně těch, které vzniknou až po vytvoření kanálu (např. −10)
filter	list	Ne	bez filtru	Seznam typů požadovaných událostí (názvy typů událostí oddělené čárkou). Parametr je nepovinný, a pokud není uveden, pak se v rámci kanálu předávají všechny typy událostí daného zařízení, které nejsou defaultně skryté. Pro odběr defaultně skrytých událostí je nutné události v tomto parametru vyžádat. Defaultně skryté události jsou: • **FingerEnrollState** • **DirectorySaved** • **DirectoryChanged** • **HardwareChanged** • **DisplayTouched** • **PairingStateChanged** • **LiftConfigChanged** • **CapabilitiesChanged** • **ConfigurationChanged** • **ExtCameraStateChanged**

Parametr	Typ	Povinný	Výchozí hodnota	Popis
duration	uint32	Ne	90	Definuje dobu v sekundách, po které bude kanál automaticky uzavřen, pokud na něm nebudou probíhat žádné operace čtení pomocí **/api/log/pull**. Každým čtením z kanálu je automaticky životnost kanálu prodloužena o zde nastavenou hodnotu. Maximální možná hodnota je 3600 s.

Odpověď je ve formátu **application/json** a obsahuje pouze identifikátor vytvořeného subscription.

Parametr	Typ	Popis
id	uint32	Unikátní identifikátor vytvořeného subscription.

Příklad:

```
GET /api/log/subscribe?filter=KeyPressed,InputChanged
{
  "success" : true,
  "result" : {
    "id" : 2121013117
  }
}
```

5.10.3 api log unsubscribe

Funkce **/api/log/unsubscribe** uzavře kanál odběru událostí (subscription) s daným identifikátorem. Po provedení funkce nebude možné daný identifikátor použít, tj. následná volání funkce **/api/log/pull**, příp. **/api/log/unsubscribe** se stejným identifikátorem skončí chybou.

Funkce je součástí služby **Logging** a pro provedení funkce nejsou potřeba žádná zvláštní privilegia uživatele.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Typ	Povinný	Výchozí hodnota	Popis
id	uint32	Ano	–	Identifikátor existujícího kanálu získaný při předchozím volání funkce **/api/log/subscribe**.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/log/unsubscribe?id=21458715
{
  "success" : true,
}
```

5.10.4 api log pull

Funkce **/api/log/pull** provádí čtení položek z fronty kanálu (subscription) a vrací seznam dosud nevyčtených událostí, příp. prázdný seznam, pokud žádná nová událost není k dispozici. Větší množství událostí jsou pak stahovány po dávkách o 128 událostech.

Pomocí parametru **timeout** lze specifikovat maximální dobu, za jakou musí interkom vygenerovat odpověď. V případě, že ve frontě kanálu je alespoň jedna položka, odpověď je vygenerována okamžitě. V případě, že je fronta kanálu prázdná, interkom odloží odeslání odpovědi do doby, než vznikne nová událost, příp. vyprší nastavený timeout.

Funkce je součástí služby **Logging** a pro provedení funkce nejsou potřeba žádná zvláštní privilegia uživatele. Čtení událostí je podmíněno privilegii uživatele tyto události sledovat, viz tabulka událostí v 5.10.2 api log subscribe.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Typ	Povinný	Výchozí hodnota	Popis
id	uint32	Ano	–	Identifikátor existujícího kanálu vytvořeného předchozím voláním funkce **/api/log/subscribe**.

Parametr	Typ	Povinný	Výchozí hodnota	Popis
timeout	uint32	Ne	0	Specifikuje zpoždění odpovědi (v sekundách) v případě, že fronta kanálu je prázdná. Výchozí hodnota 0 znamená, že interkom odpovídá vždy okamžitě bez jakéhokoliv zpoždění.

Odpověď je ve formátu **application/json** a obsahuje seznam událostí.

Parametr	Typ	Popis
events	array	Pole objektu události. V případě, že během nastaveného timeoutu nenastala žádná událost, je pole prázdné.

Příklad:

```
GET /api/log/pull
{

  "success" : true,
  "result" : {
    "events" : [
      {
        "id" : 1,
        "tzShift" : 0,
        "utcTime" : 1437987102,
        "upTime" : 8,
        "event" : "DeviceState",
        "params" : {
          "state" : "startup"
        }
      },
      {
        "id" : 3,
        "tzShift" : 0,
        "utcTime" : 1437987105,
        "upTime" : 11,
        "event" : "RegistrationStateChanged",
        "params" : {
          "sipAccount" : 1,
          "state" : "registered"
        }
      }
    ]
  }
}
```

Události

Každá událost v poli **events** obsahuje následující informace, které jsou společné pro všechny typy událostí:

Parametr	Typ	Popis
id	uint32	Interní ID záznamu o události (32bit číslo, 1 po restartu interkomu, inkrementované s každou novou událostí)
utcTime	uint32	Absolutní čas vzniku události (Unix Time, UTC – koordinovaný světový čas).
upTime	uint32	Relativní čas vzniku události (počet sekund od restartu interkomu).
tzShift	int32	Rozdíl mezi místním časem a časem UTC v minutách. Přičtením této hodnoty k utcTime se získá místní čas vzniku události dle nastavení časové zóny v zařízení: localTime = utcTime + tzShift * 60
event	string	Typ události ("KeyPressed", "InputChanged", ...)
params	object	Specifické parametry události.

Událost DeviceState

Signalizuje změny stavu zařízení.

Parametry události:

Parametr	Typ	Popis
state	string	Signalizovaný stav zařízení: **"startup"** – generováno jednorázově po startu zařízení (vždy úplně první událost)

Příklad:

```
{
  "id" : 1,
  "tzShift" : 0,
  "utcTime" : 1437987102,
  "upTime" : 8,
  "event" : "DeviceState",
  "params" : {
    "state" : "startup"
  }
}
```

Událost AudioLoopTest

Signalizuje provedení automatického audio loop testu a jeho výsledek. Událost je signalizovaná vždy po provedení automatického testu (naplánovaného příp. manuálně spuštěného).

Parametr	Typ	Popis
result	string	Výsledek provedeného testu. **"passed"** – test byl úspěšně proveden a nebyl zjištěn žádný problém **"failed"** – test byl proveden, ale byl detekován problém s reproduktorem nebo mikrofonem v zařízení

Příklad:

```
{
  "id" : 26,
  "tzShift" : 0,
  "utcTime" : 1438073190,
  "upTime" : 9724,
  "event" : "AudioLoopTest",
  "params" : {
    "result" : "passed"
  }
}
```

Událost MotionDetected

Signalizuje detekci pohybu pomocí kamery. Událost je dostupná pouze na modelech vybavených kamerou. Událost se generuje pouze v případě, že v konfiguraci kamery interkomu je tato funkce povolena.

Parametry události:

Parametr	Typ	Popis
state	string	Stav detektoru pohybu: **"in"** – signalizuje začátek intervalu, ve kterém byl detekován pohyb **"out"** – signalizuje konec intervalu, ve kterém byl detekován pohyb

Příklad:

```
{
  "id" : 2,
  "tzShift" : 0,
  "utcTime" : 1441357589,
  "upTime" : 1,
  "event" : "MotionDetected",
  "params" : {
    "state" : "in"
  }
}
```

Událost NoiseDetected

Signalizuje zvýšenou hladinu zvuku detekovanou pomocí zabudovaného nebo externího mikrofonu. Událost se generuje pouze v případě, že v konfiguraci interkomu je tato funkce povolena.

Parametry události:

Parametr	Typ	Popis
state	string	Stav detektoru hluku: **"in"** – signalizuje začátek intervalu, ve kterém byl detekován hluk **"out"** – signalizuje konec intervalu, ve kterém byl detekován hluk

Příklad:

```
{
  "id" : 2,
  "tzShift" : 0,
  "utcTime" : 1441357589,
  "upTime" : 1,
  "event" : "NoiseDetected",
  "params" : {
    "state" : "in"
  }
}
```

Události KeyPressed a KeyReleased

Signalizuje stisk **(KeyPressed)** nebo uvolnění **(KeyReleased)** tlačítka nebo klávesy numerické klávesnice.

Parametry události:

Parametr	Typ	Popis
key	string	Kód stisknutého nebo uvolněného tlačítka: **"0"** až "**9"** – tlačítka numerické klávesnice **"%1"** – **"%150"** - tlačítka rychlé volby **"*"** – tlačítko se symbolem * příp. telefonu **"#"** – tlačítko se symbolem # příp. klíčku

Příklad:

```
{
  "id" : 4,
  "tzShift" : 0,
  "utcTime" : 1437987888,
  "upTime" : 794,
  "event" : "KeyPressed",
  "params" : {
    "key" : "5"
  }
}
```

Událost CodeEntered

Signalizuje zadání kódu uživatelem na numerické klávesnici interkomu. Událost se generuje pouze na zařízeních vybavených numerickou klávesnicí.

Parametry události:

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikáté zadání kódu se jedná.
direction	string	Směr průchodu: **"in"** – příchod **"out"** – odchod **"any"** – průchod *Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.*
code	string	Uživatelem zadaný kód, např. "1234". Kódy mají minimálně dvě číslice a kód **"00"** nelze použít.
type	string	
uuid	string	Unikátní ID uživatele.
valid	boolean	Platnost zadaného kódu (tj. zda je kód zaveden v konfiguraci interkomu jako platný kód uživatele nebo jako platný univerzální kód spínače). **false** – kód není platný **true** – kód je platný

Příklad:


```
{
  "id" : 128,
  "tzShift" : 0,
  "utcTime" : 1548078453,
  "upTime" : 1061,
  "event" : "CodeEntered",
  "params" : {
    "ap" : 0,
    "session" : 8,
    "direction" : "in",
    "code" : "1234",
    "type" : "user",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "valid" : true
  }
}
```

Událost CardEntered

Signalizuje přiložení RFID karty ke čtečce. Událost je dostupná pouze na modelech vybavených čtečkou RFID karet.

Parametry události:

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikáté přiložení karty se jedná.
direction	string	Směr průchodu RFID čtečky: **"in"** – příchod **"out"** – odchod **"any"** – průchod *Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.*
reader	string	Jméno RFID čtečky příp. Wiegand modulu, příp. jedna z následujících hodnot pro nemodulární modely interkomu: **"internal"** – interní čtečka (2N IP interkomy) **"external"** – externí čtečka připojená pomocí Wiegand interface *Pozn.: Jméno čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.*
uid	string	Jednoznačný identifikátor přiložené karty (číslo v hexadecimálním formátu, 6–16 znaků, podle typu karty).
uuid	string	Unikátní ID uživatele.
valid	boolean	Platnost přiložené RFID karty (tj. zda je uid karty přiřazeno jednomu z uživatelů v adresáři interkomu). **false** – karta není platná **true** – karta je platná

Příklad:


```
{
  "id" : 60,
  "tzShift" : 0,
  "utcTime" : 1548078014,
  "upTime" : 622,
  "event" : "CardEntered",
  "params" : {
    "ap" : 0,
    "session" : 5,
    "direction" : "in",
    "reader" : "ext2",
    "uid" : "4BD9E903",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "valid" : true
  }
}
```

Události InputChanged a OutputChanged

Signalizuje změnu stavu logického vstupu **(InputChanged)** nebo výstupu **(OutputChanged).** Aktuální seznam dostupných vstupů a výstupů lze zjistit pomocí funkce **/api/io/caps**.

Parametry události:

Parametr	Typ	Popis
port	string	Název I/O portu.
state	boolean	Aktuální logický stav I/O portu: **false** – neaktivní, log.0 **true** – aktivní, log.1

Příklad:

```
{
  "id" : 2,
  "tzShift" : 0,
  "utcTime" : 1437987103,
  "upTime" : 9,
  "event" : "OutputChanged",
    "params" : {
      "port" : "led_secured",
      "state" : false
  }
}
```

Událost SwitchStateChanged

Signalizuje změnu stavu spínače (viz konfigurace interkomu Hardware | Spínače).

Parametry události:

Parametr	Typ	Popis
switch	uint32	Číslo spínače 1..4
state	boolean	Aktuální logický stav spínače: **false** – neaktivní, log.0 **true** – aktivní, log.1
originator		Informuje, jakým způsobem byl switch aktivován. **profile** – vyvoláno přechodem na nastavený aktivní časový profil. **api** – vyvoláno pomocí http api (api/switch/ctrl). **ap** – autentizace uživatele na přístupovém bodu. Událost pak doplňuje číslo app a session. **rex** – stisk odchodového tlačítka (tlačítko, které otevírá dveře po definovanou dobu, kdy osoba opouští místnost). **idt** – vyvoláno pomocí http api (api/switch/ctrl), pokud se použila speciální autentizace pro 2N® Indoor Touch 2.0, 1.0. **dtmf** – použití dtmf kódu v hovoru. **auth** – autorizace pomocí uživatelského, universálního nebo zónového kódu. **uni** – autorizace pomocí univerzálního kódu. **zone** – autorizace pomocí zónového kódu. **automation** – vyvoláno pomocí akce automation.

Příklad:

```
{
  "id" : 2,
  "tzShift" : 0,
  "utcTime" : 1437987103,
  "upTime" : 9,
  "event" : "SwitchStateChanged",
    "params" : {
        "switch" : 1,
        "state" : true
  }
}
```

Událost CallStateChanged

Signalizuje vytvoření, ukončení příp. jinou změnu stavu probíhajícího hovoru.

Parametry události:

Parametr	Typ	Popis
	string	Specifikuje, zda se jedná o příchozí nebo odchozí hovor: **"incoming"** – příchozí hovor **"outgoing"** – odchozí hovor
	string	Aktuální stav automatu hovoru: **"connecting"** – probíhá sestavování hovoru (pouze odchozí hovory) **"ringing"** – vyzvánění **"connected"** – hovor spojen **"terminated"** – hovor ukončen
peer	string	SIP URI volajícího (příchozí hovory) nebo volaného (odchozí hovory) účastníka.
session	uint32	Jednoznačný identifikátor hovoru. Identifikátor hovoru lze dále použít ve funkcích **/api/call/answer**, **/api/call/hangup** a **/api/call/status**.
call	uint32	TBD

Příklad:


```
{
  "id" : 5,
  "tzShift" : 0,
  "utcTime" : 1438064126,
  "upTime" : 660,
  "event" : "CallChanged",
  "params" : {
    "direction" : "incoming",
    "" : "ringing",
    "peer" : "sip:2229@10.0.97.150:5062;user=phone",
    "session" : 1,
    "call" : 1
  }
}
```

Událost RegistrationChanged

Signalizuje změnu stavu registrace všech SIP účtů k SIP serveru.

Parametry události:

Parametr	Typ	Popis
sipAccount	uint32	Číslo SIP účtu, na kterém proběhla změna stavu: **1** – první SIP účet **2** – druhý SIP účet
	string	Udává nový stav automatu registrace SIP účtu: **"registered"** – účet byl úspěšně zaregistrován **"unregistered"** – účet byl odregistrován **"registering"** – probíhá registrace **"unregistering"** – probíhá odregistrování účtu

Příklad:

```
{
  "id" : 3,
  "tzShift" : 0,
  "utcTime" : 1437987105,
  "upTime" : 11,
  "event" : "RegistrationChanged",
  "params" : {
    "sipAccount" : 1,
    "" : "registered"
  }
}
```

Událost TamperSwitchActivated

Signalizuje aktivaci ochranného spínače – otevření krytu zařízení. Funkce ochranného spínače musí být nakonfigurována v menu Digitální vstupy / Ochranný spínač.

Parametry události:

Parametr	Typ	Popis
	string	Stav ochranného spínače: **"in"** – signalizuje aktivaci ochranného spínače (tj. otevření krytu zařízení) **"out"** – signalizuje deaktivaci ochranného spínače (tj. zavření krytu zařízení)

Příklad:

```
{
  "id" : 54,
  "tzShift" : 0,
  "utcTime" : 1441357589,
  "upTime" : 158,
  "event" : "TamperSwitchActivated",
  "params" : {
    "state" : "in"
  }
}
```

Událost UnauthorizedDoorOpen

Signalizuje neautorizované otevření dveří. Vyžaduje připojení snímače otevřených dveří na jeden z digitálních vstupů a příslušné nastavení v menu Digitální vstupy | Stav dveří.

Parametry události:

Parametr	Typ	Popis
state	string	Stav neautorizovaného otevření dveří: **"in"** – signalizuje zahájení stavu neautorizovaného otevření **"out"** – signalizuje ukončení stavu neautorizovaného otevření dveří

Příklad:

```
{
  "id" : 80,
  "tzShift" : 0,
  "utcTime" : 1441367842,
  "upTime" : 231,
  "event" : "UnauthorizedDoorOpen",
  "params" : {
    "state" : "in"
  }
}
```

Událost DoorOpenTooLong

Signalizuje dlouhé otevření dveří, resp. nezavření dveří do nastavené doby. Vyžaduje připojení snímače otevřených dveří na jeden z digitálních vstupů a příslušné nastavení v menu Digitální vstupy | Stav dveří.

Parametry události:

Parametr	Typ	Popis
state	string	Stav dlouhého otevření dveří: **"in"** – signalizuje zahájení stavu dlouhého otevření dveří. **"out"** – signalizuje ukončení stavu dlouhého otevření dveří.

Příklad:

```
{
  "id" : 96,
  "tzShift" : 0,
  "utcTime" : 1441369745,
  "upTime" : 275,
  "event" : "DoorOpenTooLong",
  "params" : {
    "state" : "out"
  }
}
```

Událost LoginBlocked

Signalizuje dočasné zablokování přístupu k webovému rozhraní z důvodu opakovaného zadání neplatného přihlašovacího jména nebo hesla.

Parametry události:

Parametr	Typ	Popis
address	string	IP adresa, ze které bylo provedeno opakované přihlášení s neplatnými údaji.

Příklad:

```
{
  "id" : 5,
  "tzShift" : 0,
  "utcTime" : 1441369745,
  "upTime" : 275,
  "event" : "LoginBlocked",
  "params" : {
    "address" : "10.0.23.32"
  }
}
```

Událost UserAuthenticated

Signalizuje autentizaci uživatele a následné otevření dveří.

Parametry události:

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikátou autorizaci uživatele se jedná.
name	string	Specifikuje jméno uživatele z telefonního seznamu
uuid	string	Unikátní ID uživatele.
apbBroken	string	Platnost přiložené karty v Anti-passback oblasti. **false** – oblast soft APB není aktivní **true** – oblast soft ABP je aktivní a je porušena

Příklad:


```
{
  "success" : true,
  "result" : {
    "events" : [
      {
        "id" : 65,
        "tzShift" : 0,
        "utcTime" : 1593606655,
        "upTime" : 7951,
        "event" : "UserAuthenticated",
        "params" : {
          "ap" : 0,
          "session" : 6,
          "name" : "Alice Gruberov\u00E1",
          "uuid" : "8fa29ebc-2fe8-4a8c-9a3b-d8b0351fb6f8",
          "apbBroken" : true
        }
      }
    ]
  }
}
```

Událost CardHeld

Signalizuje přiložení RFID karty ke čtečce delší než 4 s.

Parametry události:

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikáté přiložení karty se jedná.
direction	string	Směr průchodu RFID čtečky: **"in"** – příchod **"out"** – odchod **"any"** – průchod *Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.*
reader	string	Identifikace čtečky, která kartu načetla.
uid	string	Identifikátor uživatele.
valid	string	Platnos přiložené karty (tj. zda je karty zavedena v konfiguraci interkomu jako platná pro uživatele.). **false** – karta není platná **true** – karta je platná

Příklad:


```
{
  "id" : 9,
  "tzShift" : 0,
  "utcTime" : 1516893493,
  "upTime" : 354,
  "event" : "CardHeld",
  "params" : {
    "ap" : 1,
    "session" : 4,
    "direction" : "out",
    "reader" : "ext2",
    "uid" : "3F00F318E7",
    "valid" : true
   }
}
```

Událost SilentAlarm

Signalizuje aktivaci tichého alarmu.

Parametry události:

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikátý SilentAlarm se jedná.
name	string	Specifikuje jméno uživatele z telefonního seznamu.
uuid	string	Unikátní ID uživatele.

Příklad:

```
{
  "id" : 200,
  "tzShift" : 0,
  "utcTime" : 1548079445,
  "upTime" : 2053,
  "event" : "SilentAlarm",
  "params" : {
    "ap" : 0,
    "session" : 17,
    "name" : "Joseph",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c"
  }
}
```

Událost AccessLimited

Signalizuje odmítnutí zadaného uživatele.

Parametry události:

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
type	string	card, code, finger
state	string	Stav, možné hodnoty: in = aktivní, out = neaktivní.

Příklad:


```
{
    "id" : 408,
    "tzShift" : 0,
    "utcTime" : 1517302112,
    "upTime" : 408951,
    "event" : "AccessLimited",
    "params" : {
      "ap" : 0,
      "type" : "card",
      "state" : "in"
    }
}
```

Událost PairingStateChange

Signalizuje párování s bluetooth rozhraním.

Parametry události:

Parametr	Typ	Popis
state	string	pending
authId	string	Autorizační ID

Příklad:

```
{
        "id" : 197,
        "tzShift" : 0,
        "utcTime" : 1516894499,
        "upTime" : 1360,
        "event" : "PairingStateChanged",
        "params" : {
          "state" : "pending",
          "authId" : "F2CAE955C9B4E81CD00E3A096E52543B"
        }
      }
```

Událost SwitchesBlocked

Signalizuje zablokování zámků ochranným spínačem. Pokud je funkce povolena, po aktivaci ochranného spínače dojde k zablokování všech spínačů po dobu 30 minut. Blokování bude aktivní i po restartu zařízení.

Parametry události:

Parametr	Typ	Popis
state	string	in, out

Příklad:

```
{
        "id" : 205,
        "tzShift" : 0, "utcTime" : 1516894667,
        "upTime" : 1528,
        "event" : "SwitchesBlocked",
        "params" : {
          "state" : "in"
        }
    }
```

Událost FingerEntered

Signalizuje přiložení prstu ke čtečce otisků prstů.

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikáté přiložení prstu se jedná.
direction	string	Směr průchodu čtečky otisků prstů: **"in"** – příchod **"out"** – odchod **"any"** – průchod *Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.*
uuid	string	Unikátní ID uživatele.
valid	string	Platnost otisku prstu (tj. zda je otisk prstu zaveden v konfiguraci interkomu jako platný otisk prstu uživatele). **false** – otisk prstu není platný **true** – otisk prstu je platný

Příklad: Načtení prstu zadaného uživatele


```
{
  "id" : 18,
  "tzShift" : 0,
  "utcTime" : 1548077595,
  "upTime" : 203,
  "event" : "FingerEntered",
  "params" : {
    "ap" : 0,
    "session" : 2,
    "direction" : "in",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "valid" : true
  }
}
```

Neúspěšné zadání: Načtení prstu nezadaného uživatele


```
{
  "id" : 1368,
  "tzShift" : 0,
  "utcTime" : 1548145535,
  "upTime" : 62598,
  "event" : "FingerEntered",
  "params" : {
    "ap" : 0,
    "session" : 1,
    "direction" : "in",
    "valid" : false
  }
}
```

Událost MobKeyEntered

Signalizuje autorizaci pomocí bluetooth čtečky.

Parametry události:

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikátou autorizaci pomocí Mobile KEY se jedná.
direction	string	Směr průchodu: **"in"** – příchod **"out"** – odchod **"any"** – průchod *Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.*
authid	string	Mobile Key ID.
uuid	string	Unikátní ID uživatele.
valid	string	Platnost Mobile Key (tj. zda je Mobile Key zaveden v konfiguraci interkomu jako platný Mobile Key pro uživatele). **false** – Mobile Key není platný **true** – Mobile Key je platný

Příklad:


```
{
  "id" : 161,
  "tzShift" : 0,
  "utcTime" : 1548079174,
  "upTime" : 1782,
  "event" : "MobKeyEntered",
  "params" : {
    "ap" : 0,
    "session" : 9,
    "direction" : "in",
    "authid" : "48c48155eed7ea1dbb0b4d534b7459b9",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "valid" : true
  }
}
```

Událost DoorStateChanged

Signalizuje změnu stavu dveří.

Parametry události:

Parametr	Typ	Popis
state	string	opened, closed

Příklad:

```
{
        "id" : 240,
        "tzShift" : 0,
        "utcTime" : 1516895295,
        "upTime" : 2156,
        "event" : "DoorStateChanged",
        "params" : {
          "state" : "opened"
        }
}
```

Událost UserRejected

Signalizuje odmítnutí autorizace uživatele.

Parametry události:

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikáté odmítnutí autorizace se jedná.
name	string	Jméno uživatele.
uuid	string	Unikátní ID uživatele.
reason	string	accessBlocked, switchLocked, invalidTime, invalidProfile, invalidSequence, invalidCredential, authInterupted, timeout, switchDisabled

Příklad:


```
{
  "id" : 173,
  "tzShift" : 0,
  "utcTime" : 1548079274,
  "upTime" : 1882,
  "event" : "UserRejected",
  "params" : {
    "ap" : 0,
    "session" : 10,
    "name" : "Joseph",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "reason" : "invalidCredential"
  }
}
```

Událost DisplayTouched

Signalizuje dotyk na displeji.

Parametry události:

Parametr	Typ	Popis
x	string	Souřadnice bodu dotyku na displeji. Maximální hodnota je dána rozlišením displeje.
y	string	Souřadnice bodu dotyku na displeji.
dx	string	Změna souřadnic při posunu na displeji, může nabývat i záporné hodnoty. Maximální hodnota je dána rozlišením displeje.
dy	string	Změna souřadnic při posunu na displeji.

Příklad:


```
{
    "id" : 337,
    "tzShift" : 0,
    "utcTime" : 1517301424,
    "upTime" : 408263,
    "event" : "DisplayTouched",
    "params" : {
      "x" : 89,
      "y" : 100,
      "dx" : 0,
      "dy" : 0
    }
}
```

Událost DtmfEntered

Signalizuje DTMF kód v hovoru.

```
{
    "id" : 86,
    "tzShift" : 0,
    "utcTime" : 1558522871,
    "upTime" : 3531,
    "event" : "DtmfEntered",
    "params" : {
      "code" : "00",
      "type" : "uni",
      "call" : 3,
      "valid" : true
 }
}
```

Parametr	Typ	Popis
code	string	Znění zadaného kódu.
type	string	Informuje, jaký typ kódu byl použit. **uni** – univerzální kód spínače **user** – uživatelský kód
call	string	ID hovoru.
valid	string	Platnost zadaného kódu (tj. platný univerzální kód spínače nebo platný uživatelský kód). **false** – kód není platný **true** – kód je platný

Událost AccessTaken

Signalizuje přiložení karty v Anti-passback oblasti.

```
{
  "success" : true,
  "result" : {
    "events" : [

    ]
  }
}
```

Událost ApLockStateChanged

Signalizuje změnu stavu (zapnuto/vypnuto) nouzového uzamčení.

```
{
    "id" : 35,
    "tzShift" : 0,
    "utcTime" : 1558522465,
    "upTime" : 3125,
    "event" : "ApLockStateChanged",
    "params" : {
      "ap" : 0,
      "state" : "in"
    }
}
```

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
state	string	Stav změny stavu. **"in"** – signalizuje začátek intervalu nouzového uzamčení **"out"** – signalizuje konec intervalu nouzového uzamčení

Událost RexActivated

Signalizuje aktivaci vstupu, která je nastavena na tlačítko REX.

```
{
    "id" : 29,
    "tzShift" : 0,
    "utcTime" : 1558522162,
    "upTime" : 2822,
    "event" : "RexActivated",
    "params" : {
      "ap" : 1,
      "session" : 1
    }
}
```

Parametr	Typ	Popis
ap	string	Access Point, možné stavy: 0 = entry, 1 = exit.
session	string	Informuje, o kolikátou aktivaci REX tlačítka se jedná.

Událost LiftStatusChanged

Signalizuje detekci připojení/odpojení Lift Control modulu.

```
 {
        "id" : 2871,
        "tzShift" : 0,
        "utcTime" : 1561540370,
        "upTime" : 73822,
        "event" : "LiftStatusChanged",
        "params" : {
          "module" : 0,
          "ready" : true
        }

},
```

Parametr	Typ	Popis
module	string	Informuje o ID modulu.
ready	string	Informuje o stavu modulu. **false** – odpojeno **true** – připojeno

Událost LiftFloorsEnabled

Signalizuje permanentní povolení přístupu na patro nebo trvalý přístup uživatele


```
{
        "id" : 2850,
        "tzShift" : 0,
        "utcTime" : 1561540011,
        "upTime" : 73463,
        "event" : "LiftFloorsEnabled",
        "params" : {
          "type" : "user"
          "floors" : [
            0, 1, 2, 3, 4
          ],
          "uuid" : "621a5a49-1f8b-d34c-9a8b-881055864deb",
        }
      },
```



```
{
        "id" : 2855,
        "tzShift" : 0,
        "utcTime" : 1561540016,
        "upTime" : 73468,
        "event" : "LiftFloorsEnabled",
        "params" : {
          "type" : "public"
          "floors" : [
            1, 4
          ],
        }
      },
```

Parametr	Typ	Popis
type	string	Informuje, o jaký typ přístupu se jedná. **public** – změna veřejného přístupu **user** – při autentizaci uživatele
floors	string	Informuje o přístupných patrech.

Událost LiftConfigChanged

Signalizuje změnu nastavení řízení výtahu.

```
      {
       "id" : 2860,
       "tzShift" : 0,
       "utcTime" : 1561540163,
       "upTime" : 73615,
       "event" : "LiftConfigChanged",
       "params" : {
         "hash" : 11
       }
     },
```

Parametr	Typ	Popis
hash	string	Unikátní číslo změny konfigurace.

Událost CapabilitiesChanged

Signalizuje změnu dostupných funkcí.


```
{
  "success":true,
  "result":{
   "events":[
     {
      "id":21,
      "tzShift":0,
      "utcTime":1585037151,
      "upTime":256,
      "event":"CapabilitiesChanged",
      "params":{

      }

     }


   ]

  }

}
```

Parametr	Typ	Popis
id	string	Pořadové číslo události.
tzShift	uint32	Rozdíl mezi místním časem a časem UTC v minutách. Přičtením této hodnoty k utcTime se získá místní čas vzniku události dle nastavení časové zóny v zařízení: localTime = utcTime + tzShift * 60
utcTime	uint32	Absolutní čas vzniku události (Unix Time, UTC – koordinovaný světový čas).
upTime	uint32	Relativní čas vzniku události (počet sekund od restartu interkomu).
event	string	Typ události CapabilitiesChanged.
params	object	Specifické parametry události.

5.11 api audio

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/audio**.

5.11.1 api audio test

Funkce **/api/audio/test** spustí automatický test zabudovaného mikrofonu a reproduktoru interkomu. Po provedení testu je výsledek zapsán do logu v zařízení jako událost **AudioLoopTest**.

Funkce je součástí služby **Audio** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Audio (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/audio/test
{
  "success" : true
}
```

5.12 api email

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/email**.

5.12.1 api email send

Funkce **/api/email/send** odešle ze zařízení e-mail na uvedenou adresu. Pro správnou funkci je potřeba nakonfigurovat službu SMTP v zařízení (tj. nastavit adresu SMTP serveru, správné přihlašovací údaje apod.)

Funkce je součástí služby **E-mail** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **E-mail (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
to	Povinný parametr obsahující e-mailovou adresu pro doručení.
subject	Povinný parametr specifikující předmět zprávy.
body	Nepovinný parametr specifikující obsah zprávy (může obsahovat html značky). Pokud parametr není uveden, zpráva bude doručena bez obsahu.
pictureCount	Nepovinný parametr specifikující počet přiložených obrázků z kamery. V případě, že není uveden, obrázky nejsou přiloženy. Parametr může nabývat hodnot <0, 5>.
timeSpan	Časový rozsah snímků přiložených k emailu v sekundách. Parametr je nepovinný, implicitní hodnota 0.
width	Nepovinné parametry nastavující rozlišení přiložených obrázků z kamery. Výška a šířka snímku musí odpovídat jedné z podporovaných variant (viz funkce **api/camera/caps**).
height	

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/email/send?to=somebody@email.com&subject=Hello&body=Hello
{
  "success" : true
}
```

5.13 api pcap

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/pcap**.

5.13.1 api pcap

Funkce **/api/pcap** slouží ke stažení zaznamenaného provozu na síťovém rozhraní zařízení (pcap soubor). Záznam síťového provozu lze také řídit pomocí funkcí **/api/pcap/restart** a **/api/pcap/stop**.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST.**

Funkce nemá žádné parametry.

Odpověď ve formátu **application/pcap** a stažený soubor lze přímo otevřít např. v programu Wireshark.

Příklad:

```
GET /api/pcap
```

5.13.2 api pcap restart

Funkce **/api/pcap/restart** provede odstranění všech zaznamenaných paketů a znovu spustí záznam provozu na síťovém rozhraní zařízení.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/pcap/restart
{
  "success" : true
}
```

5.13.3 api pcap stop

Funkce **/api/pcap/stop** zastaví zaznamenávání provozu na síťovém rozhraní zařízení.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Příklad:

```
GET /api/pcap/restart
{
  "success" : true
}
```

5.13.4 api pcap live

Funkce **api/pcap/live** slouží ke spuštění zachytávání chunk paketů.

Skupiny služeb privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- GET
- POST

Požadavek

Požadavek obsahuje parametry ve formátu URL (nebo **application/x-www-form-urlencoded** při použití metody POST).

Tabulka 1. Parametry požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
duration	Ne	Celé číslo, které definuje délku stahování v sekundách	doba neurčitá	Definuje dobu trvání zachytávání paketů. Je-li parametr vynechán nebo roven 0, doba trvání je neurčitá (tj. dokud není stahování zastaveno funkcí **api/pcap/live/stop** nebo ukončeno cílovým zařízením).

Příklad požadavku

```
URL: https://192.168.1.1/api/pcap/live?duration=10
```

Odpověď

Zařízení začne streamovat chunky po úspěšném požadavku.

Příklad použití jazyka Python pro stažení zachycených paketů

```
command = requests.post( "https://" + address + "/api/pcap/live?duration=120",
verify=False, stream=True, auth=HTTPBasicAuth("admin", "pass") ) with
open("trace.pcap", 'wb') as f: for chunk in command.iter_content(chunk_size=None):
f.write(chunk)
```

Pokud jedno zachytávání paketů už probíhá, není možné spustit další.

5.13.5 api pcap live stop

Funkce **api/pcap/live/stop** slouží k zastavení zachytávání chunk paketů.

Skupiny služeb privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- GET
- POST

Požadavek

Požadavek nemá žádné parametry.

Příklad požadavku

```
URL: https://192.168.1.1/api/pcap/live/stop
```

Odpověď

Zařízení přestane streamovat chunky po úspěšném požadavku. Požadavek slouží k zastavení zachytávání bez zadané doby trvání nebo předčasnému zastavení zachytávání s danou dobou trvání.

Zařízení zašle v odpovědi **`success`** : **`true`**, i když neprobíhá žádné zachytávání. Pro tento koncový bod nejsou žádné konkrétní chybové kódy.

5.13.6 api pcap live stats

Funkce **api/pcap/live/stats** slouží k získání statusu zachytávání chunk paketů.

Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- GET
- POST

Požadavek

Požadavek nemá žádné parametry.

Příklad požadavku

```
URL: https://192.168.1.1/api/pcap/live/stats
```

Odpověď

Odpověď je ve formátu **application/json**. Obsahuje klíče **success** a **result**. Hodnota klíče **result** obsahuje detailní informace o stavu zachytávání paketů.

Tabulka 1. Klíče JSON odpovědi

Klíč	Typické vrácené hodnoty	Popis
running	`true` nebo `false`	Oznamuje, zda zachytávání chunk paketů právě probíhá, nebo neprobíhá.
bytesSent	Celé číslo	Obsahuje počet bytů poslaných v právě probíhajícím zachytávání otevřených paketů. Jestliže zachytávání paketů právě neprobíhá, hodnota je 0.
packetsSent	Celé číslo	Obsahuje počet paketů poslaných v právě probíhajícím zachytávání otevřených paketů. Jestliže zachytávání paketů právě neprobíhá, hodnota je 0.

Příklad odpovědi

```
{ "success": true, "result": { "running": true, "bytesSent": 11261, "packetsSent": 90
} }
```

5.14 api dir

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/dir**.

5.14.1 api dir template

Funkce **/api/dir/template** umožňuje vyhledávat šablonu záznamu v adresáři.

Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- GET
- POST

Požadavek

Tabulka 1. Parametry požadavku

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
není	-	-	-	-

Příklad požadavku

```
https://192.168.1.1/api/dir/template
```

Odpověď

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole s jedním objektem (Šablona záznamu), který obsahuje všechny dostupné klíče záznamu v adresáři včetně výchozích hodnot pro dané zařízení.

Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Note

- Dostupnost klíčů závisí na modelu, typu a hardwarové konfiguraci zařízení (např. klíč photo je dostupný pouze u zařízení, která jsou vybavena displejem a ukládají obrázky do adresáře).

Tabulka 2. Klíče odpovědi JSON v poli **users**

Klíč	Typické vrácené hodnoty	Popis
uuid	Prázdný	Unikátní uživatelský identifikátor. Když je pomocí funkce **api/dir/create** vytvořen nový záznam v adresáři, je jeho uuid buď ve formě parametru požadavku, nebo je automaticky generován zařízením. Formát uuid je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány pro prázdný uuid.
deleted	`false`	Indikuje, jestli byl záznam v adresáři vymazán, nebo ne. Vymazané záznamy zůstávají v adresáři, dokud není dosažen maximální počet záznamů. Uložené vymazané záznamy si ponechávají uuid jen z důvodů ukládání do logů. Jsou dvě platné hodnoty: `false`, `true`.
owner	Prázdný	Informace o tom, zda je záznam v adresáři vzdáleně spravován aplikací 2N® My2N nebo 2N® Access Commander nebo jiným systémem pro vzdálenou správu. Tato hodnota je určena pro interní potřeby 2N® TELEKOMUNIKACE a.s., případně pro vymazání skupiny uživatelů (viz **api/dir/delete**).
name	Prázdný	Název záznamu v adresáři (název uživatele nebo zařízení). Očekává se řetězec o maximální délce 63 znaků. Název může zůstat nevyplněn (v takovém případě je záznam v logu, e-mailu apod. identifikován pomocí uuid).

Klíč	Typické vrácené hodnoty	Popis
photo	Prázdný	Obrázek záznamu v adresáři (např. fotografie uživatele nebo logo společnosti). Ukládá se jako jpeg soubor šifrovaný pomocí base64 s touto syntaxí: **PŘÍKLAD:** ***_DATA_IN_BASE64***
email	Prázdný	E-mailová adresa záznamu v adresáři. Očekávaný formát je namestructure@domainhierarchy.top. Je možno zadat i více adres a oddělit je čárkou (uložit je jako řetězec).

Klíč	Typické vrácené hodnoty	Popis
treepath	/	Definice pozic záznamu v adresáři na displeji. • Výchozí pozice je kořenová složka. Tuto pozici zajistíte pouhým stiskem jednoho lomítka. **PŘÍKLAD: /** ukázka záznamu v kořenové složce • Záznam je možno umístit na několik pozic a ty oddělit středníkem (**;**). **PŘÍKLAD: */Folder1/;/Folder2/*** ukazuje záznam ve složce Folder1 i Folder2 • Záznamu může být přidělena volací skupina, která slouží i jako náhradní název v určité pozici na displeji, když se vynechá lomítko na konci definice pozice. **PŘÍKLAD: */Folder1/Calling Group*** ukazuje záznam ve složce Folder1 pod názvem "Calling Group" • Záznamu je v každé pozici možno individuálně přiřadit prioritu (prioritní záznam se pak zobrazuje nad ostatními záznamy) přidáním **:1** na konec definice pozice. **PŘÍKLAD: */Folder1/:1;Folder2/Calling Group:1*** ukazuje prioritní záznam ve Folder1 pod jeho názvem a prioritní záznam ve Folder2 pod názvem "Calling Group" • Jedné volací skupině je možno přiřadit více záznamů (výběr této volací skupiny na displeji vede ke skupinovému volání na všechny záznamy ve volací skupině). **PŘÍKLAD:** *Entry1:* ***/Calling Group*** *Entry2:* ***/Calling Group*** ukazuje oba záznamy v kořenové složce na jednom řádku s názvem "Calling Group" (při výběru tohoto řádku se volá na oba záznamy)
virtNumber	Prázdný	Virtuální číslo záznamu v adresáři. Virtuální čísla se dají vytáčet na numerické klávesnici (pokud je k dispozici). Maximální délka je 7 znaků. První a poslední znak se dá vybrat z rozmezí A až Z nebo 0 až 9. Zbylé znaky mohou být mezi 0 a 9.

Klíč	Typické vrácené hodnoty	Popis
deputy	Prázdný	Uuid zástupného záznamu, který je volán, když původně volaný uživatel není k dispozici nebo neodpovídá. Jestliže zástupce nemá nastaven zástupný uuid, zůstane klíč prázdný.
buttons	Prázdný	Tlačítka přiřazená tomuto záznamu v adresáři. Pole celých čísel (přiřazovaných podle pozice tlačítka počínaje 1) oddělených čárkami.

Klíč	Typické vrácené hodnoty	Popis

callPos	Pole	Informace o volání pro záznam v adresáři. Zadává se jako pole až tří objektů, tj. tří sad informací o volání. Každý z těchto tří objektů může obsahovat následující klíče:
		• peer - telefonní číslo záznamu v adresáři
		• profiles - časové profily, pokud je toto telefonní číslo platné (číslo se nevytočí, když je neplatné)
		  • P=X,..,Y
		  kde X,..,Y představují pole předdefinovaných časových profilů oddělených čárkou od 0 (časový profil 1) do 19 (časový profil 20)
		  PŘÍKLAD: *P=1,3,5* znamená, že předdefinované profily 2, 4 a 6 definují dobu platnosti telefonního čísla
		  • D=A,..,B@H:MM-H:MM
		  kde A,..,B představují pole dnů v týdnu oddělených čárkou (0 až 6 pro neděle až sobota, 7 je svátek), H znamená denní hodinu (od 0 do 23), MM znamená minuty (od 00 do 59) - dvě hodnoty definující interval. Jednotlivé definice se dají kombinovat a oddělovat středníkem.
		  PŘÍKLAD: *D=7@0:15-13:15;D=5,7@13:15-15:15;D=7@15:15-23:30* znamená, že telefonní číslo je platné ve svátek od 0:15 do 13:15, v pátek a ve svátek od 13:15 to 15:15 a ve svátek od 15:15 do 23:30.
		  PŘÍKLAD: *D=5,7* znamená, že telefonní číslo je platné v pátek a ve svátek celý den.
		• grouped - definuje, zda se číslo vytočí ve skupinovém volání spolu s předchozím číslem (třetí číslo se vytočí se zástupcem). Může nabývat hodnot `true` nebo `false`.
		• ipEye - definuje IP adresu počítače, na němž běží aplikace 2N® IP Eye (pro příjem videa, jestliže telefon nemá displej).

Klíč	Typické vrácené hodnoty	Popis

access	JSON objekt	Informace o kontrole přístupu pro záznam v adresáři. Obsahuje tyto klíče:
		• validFrom - definice začátku doby platnosti pověření záznamu v adresáři. Zadává se ve formátu Unix Time. Je-li pole prázdné, začíná doba platnosti na začátku časového období (tj. 00:00:00 UTC 1.1. 1970).
		• validTo - definice konce doby platnosti pověření záznamu v adresáři. Zadává se ve formátu Unix Time. Je-li pole prázdné, končí doba platnosti na konci časového období (tj. 03:14:07 UTC 19.1. 2038).
		• accessPoints - obsahuje pole přístupových bodů (dva přístupové body, vstup a výstup). Každá položka pole je reprezentována JSON objektem s těmito klíči:
		  • enabled - definuje, zda je obecně možné tento přístupový bod použít (tj. zda je možné se na tomto konkrétním přístupovém bodu autentizovat). Jsou dvě platné hodnoty: `true`, `false`.
		  • profiles - definuje, zda je aktuálně možné tento přístupový bod použít (tj. zda je možné se v tuto chvíli na tomto přístupovém bodě autentizovat). Syntax definice profilu je stejná jako u callPos.
		• pairingExpired - určuje, zda párování záznamu v adresáři pomocí Bluetooth vypršelo, nebo ne. Jsou dvě platné hodnoty: `true`, `false`.
		• virtCard - definuje virtuální kartu záznamu v adresáři, která je přenášena na rozhraní Wiegand a do dalších systémů třetích stan, pokud je autentizace záznamu v adresáři úspěšná. Očekává se 6 až 32 hexadecimálních znaků.
		• card - definuje až dvě karty záznamu v adresáři, které se používají pro autentizaci. Čísla karet jsou zapsána jako pole řetězců. Očekává se 6 až 32 hexadecimálních znaků.
		• mobkey - definuje ID Bluetoothové autentizace záznamu v adresáři. Očekává se 32 hexadecimálních znaků.

Klíč	Typické vrácené hodnoty	Popis
		• fpt - Šablony otisků prstů záznamu v adresáři. Očekává se zašifrovaný otisk prstu (pro více informací kontaktujte Technickou podporu). • pin - definuje PIN záznamu v adresáři. Očekává se 2 až 15 číslic. • apbException - definuje, zda má záznam v adresáři výjimku ze systému anti-passback (např. pokud ji má, mohou se jeho pověření použít vícenásobně pro vstup bez výstupu). Jsou dvě platné hodnoty: `true`, `false`. • code - definuje až tři samostatné kódy spínačů. Ty jsou zapsány v poli řetězců (2 až 15 číslic). První pozice v poli definuje kód prvního spínače. Při vložení prázdného řetězce dojde k přeskočení kódu pro daný spínač. • liftFloors - definuje konfiguraci přístupných pater výtahu po autentizaci. Očekává se následující formát: F=P,..,R@PROFILE1_DEFINITION| F=X,..,Z@PROFILE2_DEFINITION kde P,..,R a X,..,Z jsou pole podlaží oddělených čárkou (0 až 63). io_1_1 se zadává jako 0, io_1_5 jako 4, io_2_2 jako 9 a tak dále. Pole pater aktivních v určitých profilech se oddělují |. Dají se použít předem definované profily nebo ad hoc definice nového profilu (viz definice syntaxe v callPos/profily). Definici profilu lze vynechat, když není použit žádný profil. **PŘÍKLAD:** ***F=2,4*** definuje patra bez časového profilu (uživateli jsou vždy přístupná). **PŘÍKLAD:** ***F=5@P=0,7*** definuje, že šesté patro (F=5) je přístupné v pondělí a ve svátek celý den. **PŘÍKLAD:** ***F=0@P=7,13*****|** ***F=0@D=5@9:15-11:45;D=4@11:45-13:30*** definuje, že první patro (F=0) je přístupné podle předdefinovaných profilů 8 a 14 (P=7,13) a podle časového profilu definovaného v řetězci definice.

Klíč	Typické vrácené hodnoty	Popis
		• licensePlates - definuje sadu registračních značek přiřazených uživateli (pro otevření dveří na základě rozpoznání registrační značky). Jednotlivé registrační značky se oddělují čárkou. Bílé znaky se ignorují. Maximální povolený počet znaků je 255 (včetně bílých znaků). Maximální počet registračních značek v poli je 20 a každá registrační značka může mít maximálně 10 znaků jiných než bílých.
timestamp	0	Časová značka změn provedených v adresáři. Časové značky se v adresáři generují automaticky ve vzestupném pořadí. Více informací o použití časových značek najdete v kapitole **api/dir/query**.

Příklad odpovědi


```
{
    "success": true,
    "result": {
        "series": "5247939846841727056",
        "users": [
            {
                "uuid": "",
                "deleted": false,
                "owner": "",
                "name": "",
                "photo": "",
                "email": "",
                "treepath": "\/",
                "virtNumber": "",
                "deputy": "",
                "buttons": "",
                "callPos": [
                    {
                        "peer": "",
                        "profiles": "",
                        "grouped": false,
                        "ipEye": ""
                    },
                    {
                        "peer": "",
                        "profiles": "",
                        "grouped": false,
                        "ipEye": ""
                    },
                    {
                        "peer": "",
                        "profiles": "",
                        "grouped": false,
                        "ipEye": ""
                    }
                ],
                "access": {
                    "validFrom": "0",
                    "validTo": "0",
                    "accessPoints": [
                        {
                            "enabled": true,
                            "profiles": ""
                        },
                        {
                            "enabled": true,
                            "profiles": ""
```

```
                        }
                    ],
                    "pairingExpired": false,
                    "virtCard": "",
                    "card": [
                        "",
                        ""
                    ],
                    "mobkey": "",
                    "fpt": "",
                    "pin": "",
                    "apbException": false,
                    "code": [
                        "",
                        "",
                        "",
                        ""
                    ],
                    "licensePlates": "",
                    "liftFloors": ""
                },
                "timestamp": 0
            }
        ]
    }
}
```


5.14.2 api dir create

Funkce **/api/dir/create** umožňuje vytvářet (nebo přepisovat) pole záznamů v adresáři a nastavovat jejich vybraná pole.

Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- PUT

Požadavky

Požadavek obsahuje parametr **query** a parametry ve formátu **application/json**.

parametr	typ	očekávané hodnoty	výchozí hodnota	popis
`force`	query	`1 (True)` `0 (False)`	`0`	Tento klíč vybírá, zda bude záznam v adresáři identifikovaný daným uuid (viz níže) přepsán novou sadou dat. Když bude hodnota tohoto klíče nastavena na `True`, přepíší se existující data novou sadou dat a zbylá pole budou nastavena na výchozí hodnoty. Když bude nastavena na `False` nebo vynechána (omitted) a v adresáři existuje záznam s daným uuid, zařízení odpoví chybovým kódem `EDIR_UUID_ALREADY_EXISTS` a změny konfigurace nebudou provedeny. Tento klíč nijak neovlivňuje vytvoření nového záznamu v adresáři.

parametr	typ	očekávané hodnoty	výchozí hodnota	popis
.users	application/json	pole JSON objektů definujících parametry záznamu v adresáři	n/a	Přehled všech dostupných klíčů ve JSON definici záznamu v adresáři najdete v kapitole **api/dir/template**. Do pole je možné zadávat prázdné objekty. Zařízení vytvoří pro každý prázdný objekt prázdný záznam v adresáři (pouze s přiřazeným identifikátorem uuid). Jestliže nějaký objekt v poli obsahuje klíč uuid, vytvoří se nebo změní záznam s daným uuid nebo zařízení odpoví chybovým kódem. Jestliže nějaký objekt v poli neobsahuje uuid, zařízení vytvoří nový záznam a přiřadí mu nový identifikátor uuid. Parametry záznamu jsou nastaveny na hodnoty podle klíčů definovaných v JSON struktuře požadavku. Viz příklad dole. Jestliže je klíč vynechán nebo jeho hodnota je prázdné pole, odpověď zařízení obsahuje pouze klíč **series** (nevytvoří se nové záznamy v adresáři).

Příklady požadavku

Požadavek 1: Vytvoření nového adresáře

```
URL: https://192.168.1.1/api/dir/create?force=1
JSON:
{
  "users": [
    {
      "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF",
      "name": "Julius Thompson",
      "email": "Julius@snlsa.com",
      "access": {
        "pin": "1234"
      }
    },
    {
      "name": "Carlos Inpiers",
      "owner": "My2N",
      "email": "Carlos_Inpieros@emailos.cz"
    },
    {
      "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF",
      "name": "Fridrich Dairy",
      "email": "mycountry",
      "access": {
        "pin": "5678"
      },
      "test": "something",
      "albert": "einstein"
    },
    {},
    {}
  ]
}
```

1. První záznam: Pokud v adresáři není žádný záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení vytvoří záznam s tímto identifikátorem a nastaví název jeho parametrů, e-mail a přístup na zadané hodnoty. Je-li v adresáři záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení přepíše název jeho parametrů, e-mail a přístup na zadané hodnoty a všechny jeho ostatní parametry nastaví na výchozí hodnoty (protože parametr klíče force je nastaven na `True`).
2. Zařízení vytvoří druhý záznam, přiřadí mu náhodný identifikátor uuid, nastaví jeho název, vlastníka a e-mail na zadané hodnoty a zbylým parametrům ponechá výchozí hodnoty.
3. Třetí záznam nepřepíše stávající záznam se stejným uuid, protože je tam několik chyb (e-mail má nesprávný formát, jsou tam neexistující pole s názvem **test** a **albert**).

4. Vytvoření čtvrtého a pátého záznamu proběhlo úspěšně s náhodně přiřazeným identifikátorem uuid a všechna pole byla nastavena na výchozí hodnoty.

Požadavek 2: Vytvoření 3 nových uživatelů

```
URL: https://192.168.1.1/api/dir/create
JSON:
{
   "users":[
      {
         "name":"user1",
         "email":"user1gmail.com",
         "callPos":[
            {
               "peer":"2246",
               "profiles":"",
               "grouped":false
            }
         ]
      },
      {
         "name":"user2",
         "email":"user2@gmail.com",
         "callPos":[
            {
               "peer":"2246",
               "profiles":"",
               "grouped":false
            }
         ]
      },
      {
         "name":"user3",
         "email":"user3@gmail.com",
         "callPos":[
            {
               "peer":"234324234",
               "profiles":"",
               "grouped":false
            }
         ]
      }
   ]
}
```

Požadavek se snaží vytvořit tři nové záznamy v adresáři. Jednotlivým záznamům se snaží přiřadit náhodný identifikátor uuid, nastavit jejich jméno, e-mail a informace o volání (CallPos). Zbylým parametrům ponechá výchozí hodnoty.

Odpověď

Odpověď je ve formátu **application/json**. **Výsledek (result)** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole objektů, které obsahují klíče a hodnoty z výsledku požadavku (viz následující tabulka).

Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče JSON odpovědi v poli **users**

Klíč	Typické vrácené hodnoty	Popis
uuid	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) vytvořeného, upraveného nebo nezměněného záznamu.
timestamp	celé číslo	Časová značka změn provedených v adresáři. Více informací o použití časové značky najdete v kapitole **api/dir/query**. Časová značka je uvedena, jen když je změna v adresáři provedena úspěšně.

Klíč	Typické vrácené hodnoty	Popis
errors	pole chybových objektů	Chybový objekt obsahující pole všech chyb, které nastaly. Objekt **errors** je uveden, jen když se změnu v adresáři nepodařilo provést. V hodnotě klíče **code** je obsažen chybový kód, který ukazuje důvod neúspěšného provedení změny v adresáři. Další zpřesnění důvodu chyby může být v hodnotě klíče **field** pro chybové kódy `EDIR_FIELD_NAME_UNKNOWN` a `EDIR_FIELD_VALUE_ERROR`, identifikující klíč nebo hodnotu nesplňující validační pravidla. V odpovědi se mohou vrátit následující chybové kódy: • `EDIR_UUID_DOES_NOT_EXIST` – uživatel s daným uuid neexistuje. • `EDIR_UUID_IS_MISSING` – chybí povinný parameter uuid. • `EDIR_UUID_INVALID_FORMAT` – uuid nemá platný formát, který je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány pro prázdný uuid. • `EDIR_UUID_ALREADY_EXISTS` – v adresáři existuje záznam se zadaným identifikátorem uuid a klíč **force** je nastaven na 0 (False) nebo omitted. Požadovaná změna proto nemůže být provedena. • `EDIR_FIELD_NAME_UNKNOWN` – neznámý klíč. Seznam dostupných klíčů pro dané zařízení najdete v kapitole **/api/dir/template**.

Klíč	Typické vrácené hodnoty	Popis
		• `EDIR_FIELD_NOT_AVAILABLE` – zadaný klíč není platný pro daný model zařízení. Seznam dostupných klíčů pro dané zařízení najdete v kapitole **/api/dir/template**. • `EDIR_FIELD_VALUE_ERROR` – zadaná hodnota nesplňuje validační kritéria (hodnota není platná). • `EDIRLIM_USER` – adresář je plný. • `EDIRLIM_PHOTO` – byl dosažen maximální počet fotografií v úložišti. Nové záznamy mohou být vytvořeny jen bez fotografií. • `EDIRLIM_FPT` – byl dosažen maximální počet šablon otisků prstů v úložišti. Nové záznamy mohou být vytvořeny jen bez otisků prstů. • `EINCONSISTENT` – v hodnotách klíčů **validFrom** a **validTo** je nesrovnalost (**validFrom** musí být menší než **validTo**).

Příklady odpovědí

Odpověď na požadavek 1: Vytvoření nového adresáře


```
{
  "success": true,
  "result": {
    "series": "6423407687606431951",
    "users": [
      {
        "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF",
        "timestamp": 34
      },
      {
        "uuid": "044197A7-54AD-7577-6EEA-787A6097263E",
        "timestamp": 35
      },
      {
        "errors": [
          {
            "code": "EDIR_FIELD_VALUE_ERROR",
            "field": "email"
          },
          {
            "code": "EDIR_FIELD_NAME_UNKNOWN",
            "field": "test"
          },
          {
            "code": "EDIR_FIELD_NAME_UNKNOWN",
            "field": "albert"
          }
        ]
      },
      {
        "uuid": "41970B83-21C8-45DD-8FFC-787A6097263E",
        "timestamp": 36
      },
      {
        "uuid": "0447BBA7-6E7c-420C-A654-466D43D6A067",
        "timestamp": 37
      }
    ]
  }
}
```


1. První záznam je vytvořen se zadaným identifikátorem uuid a zadanými poli (nezadaná pole jsou nastavena na výchozí hodnoty). Záznam se vytvoří bez ohledu na existenci záznamu se stejným uuid, protože klíč **force** je v požadavku nastaven na `true`. Vrátí se časová značka změny.

2. Druhý záznam je vytvořen s náhodným uuid a jsou vyplněna zadaná pole (nezadaná pole jsou nastavena na výchozí hodnoty). Vrátí se časová značka změny.
3. Třetí objekt v požadavku obsahoval neplatný formát e-mailové adresy. Navíc klíče **test** a **albert** odkazovaly k neexistujícím polím.
4. Vytvoření čtvrtého a pátého záznamu proběhlo úspěšně s náhodně přiřazeným identifikátorem uuid a všechna pole byla nastavena na výchozí hodnoty. Časová značka v zařízení se tudíž dvakrát aktualizovala. Vrátí se časové značky změn.

Odpověď na požadavek 2: Vytvoření 3 nových uživatelů


```
{
   "success":true,
   "result":{
      "series":"20680937807286928 47",
      "users":[
         {
            "uuid":"01234567-89AB-CDEF-0123-456789ABCDCC",
            "errors":[
               {
                  "code":"EDIR_FIELD_VALUE_ERROR",
                  "field":"email"
               }
            ]
         },
         {
            "uuid":"01234567-89AB-CDEF-0123-456789ABCDCB",
            "timestamp":15207
         },
         {
            "uuid":"01234567-89AB-CDEF-0123-456789ABCDCA",
            "timestamp":15208
         }
      ]
   }
}
```


E-mail záznamu pro uživatele user1 nebyl platný, záznam tedy nebyl vytvořen. Zbylí dva uživatelé byli vytvořeni úspěšně.

5.14.3 api dir update

Funkce **/api/dir/update** umožňuje aktualizaci pole záznamů v adresáři a nastavení jejich vybraných polí.

Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- PUT

Požadavek

Požadavek obsahuje parametry ve formátu **application/json**. Více informací o jednotlivých parametrech záznamu v adresáři a jejich zobrazení najdete v kapitole **api/dir/template**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
users	Ano	pole JSON objektů definujících parametry záznamu v adresáři	-	Pole musí obsahovat nejméně jeden objekt s klíčem **uuid**, který definuje, který záznam se má aktualizovat. Přehled dostupných klíčů v JSON definici záznamu v adresáři najdete v kapitole **api/dir/template**.

Příklad požadavku

```
URL: https://192.168.1.1/api/dir/update JSON { "users": [ { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD", "email": "abcd@def.cz", "access": { "pin": "1234" } }, { "uuid": "76543210-68FF-18CA-3210-FEDCBA987654", "name": "ABCD2", "owner": "My2N", "email": "abcd2@def.cz" }, { "uuid": "01234567-89A-CDEF-0123-456789ABCDEF", "name": "ABCD3", "owner": "My2N", "email": "abcd3@def.cz" }, { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD4", "owner": "My2N", "email": "abcd4@def.cz", "albert": "einstein" }, { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD4", "owner": "My2N", "email": "abcd4@def.cz", "access.pin": "hello" } ] }
```

Pokud není v adresáři žádný záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení odpoví chybovým kódem (viz dále). To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Jestliže je v adresáři záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, budou jeho parametry aktualizovány podle hodnot zadaných pro jednotlivé klíče. To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Třetí záznam nebude aktualizován (uuid má špatný formát).

Čtvrtý záznam nebude aktualizován (neznámý název pole).

Pátý záznam nebude aktualizován (nesprávný formát přístupového PINu).

Odpověď

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole objektů, které obsahují klíče a hodnoty z výsledku požadavku (viz následující tabulka).

Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče JSON odpovědi v poli **users**

Klíč	Typické vrácené hodnoty	Popis
uuid	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) upraveného nebo nezměněného zápisu.
timestamp	celé číslo	Časová značka změn provedených v adresáři. Více informací o použití časové značky najdete v kapitole **api/dir/query**. Časová značka je uvedena, jen když je změna v adresáři provedena úspěšně.

Klíč	Typické vrácené hodnoty	Popis
errors	pole chybových objektů	Chybový objekt obsahující pole všech chyb, které nastaly. Objekt **errors** je uveden, jen když se změnu v adresáři nepodařilo provést. V hodnotě klíče **code** je obsažen chybový kód, který ukazuje důvod neúspěšného provedení změny v adresáři. Další zpřesnění důvodu chyby může být v hodnotě klíče **field** pro chybové kódy `EDIR_FIELD_NAME_UNKNOWN` a `EDIR_FIELD_VALUE_ERROR`, identifikující klíč nebo hodnotu nesplňující validační pravidla. V odpovědi se mohou vrátit následující chybové kódy: • `EDIR_UUID_DOES_NOT_EXIST` - záznam s daným uuid neexistuje (tj. nedá se aktualizovat). • `EDIR_UUID_IS_MISSING` - chybí povinný klíč **uuid**. • `EDIR_UUID_INVALID_FORMAT` - uuid nemá platný formát, který je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány pro prázdný uuid. • `EDIR_FIELD_NAME_UNKNOWN` - neznámý klíč. Seznam dostupných klíčů pro dané zařízení najdete v kapitole **/api/dir/template**. • `EDIR_FIELD_VALUE_ERROR` - zadaná hodnota nesplňuje validační kritéria (hodnota není platná). • `EDIRLIM_PHOTO` - byl dosažen maximální počet fotografií v úložišti. Další fotografie se nedají přidávat. • `EDIRLIM_FPT` - byl dosažen maximální počet šablon otisků prstů v úložišti. Další otisky prstů se nedají přidávat. • `EINCONSISTENT` - v hodnotách klíčů **validFrom** a **validTo** je nesrovnalost (**validFrom** musí být menší než **validTo**).

Příklad odpovědi


```
{ "success": true, "result": { "series": "6423407687606431951", "users": [ { "uuid":
"01234567-89AB-CDEF-0123-456789ABCDEF", "timestamp": 39 }, { "uuid":
"76543210-68FF-18CA-3210-FEDCBA987654", "errors": [ { "code":
"EDIR_UUID_DOES_NOT_EXIST" } ] }, { "uuid": "01234567-89A-CDEF-0123-456789ABCDEF",
"errors": [ { "code": "EDIR_UUID_INVALID_FORMAT" } ] }, { "uuid": "01234567-89AB-
CDEF-0123-456789ABCDEF", "errors": [ { "code": "EDIR_FIELD_NAME_UNKNOWN", "field":
"albert" } ] }, { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "errors": [ {
"code": "EDIR_FIELD_VALUE_ERROR", "field": "access.pin" } ] } ] } }
```


První záznam v adresáři je úspěšně aktualizován, vrátí se jeho **uuid** a **timestamp** změny.

Druhý záznam neexistuje (není záznam s tímto **uuid**).

Třetí objekt má špatný formát **uuid**.

Ve čtvrtém objektu je uveden neznámý klíč **albert**.

V pátém objektu je zadána nesprávná hodnota PIN kódu.

5.14.4 api dir delete

Funkce **/api/dir/delete** slouží k vymazávání pole záznamů v adresáři.

Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- PUT

Požadavek

Požadavek obsahuje parametry ve formátu **application/json**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
owner	Ano, pokud users je omitted	řetězec	-	Všechny záznamy v adresáři se zadaným vlastníkem budou vymazány.
users	Ano, pokud owner je omitted	pole JSON objektů definujících identifikátory uuid	-	Pole musí obsahovat alespoň jeden objekt s klíčem **uuid**, který definuje, které pole má být vymazáno.

Příklad požadavku

```
URL: https://192.168.1.1/api/dir/delete JSON (owner specified) { "owner": "My2N" }
JSON (uuid specified) { "users": [ { "uuid": "01234567-89AB-
CDEF-0123-456789ABCDEF" }, { "uuid": "76543210-68FF-18CA-3210-FEDCBA987654" },
{ "uuid": "76543210-68FF-18-3210-FEDCBA987654" } ] }
```

Pokud není v adresáři žádný záznam se zadaným vlastníkem, vrátí se prázdné pole.

Pokud není v adresáři žádné pole se zadaným uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení odpoví chybovým kódem (viz dále). To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Jestliže je v adresáři záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, bude vymazán. To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Třetí uuid má nesprávný formát a vrátí se chyba.

Odpověď

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole objektů, které obsahují klíče **uuid** a **timestamp**.

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče odpovědi JSON v poli **users**

Klíč	Typické vrácené hodnoty	Popis
uuid	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) vymazaného nebo nezměněného záznamu.
timestamp	celé číslo	Časová značka změn provedených v adresáři. Více informací o použití časové značky najdete v kapitole **api/dir/query**. Časová značka je uvedena, jen když je změna v adresáři provedena úspěšně.
errors	pole chybových objektů	Chybový objekt obsahující pole všech chyb, které nastaly. Objekt **errors** je uveden, jen když se změnu v adresáři nepodařilo provést. V hodnotě klíče **code** je obsažen chybový kód, který ukazuje důvod neúspěšného provedení změny v adresáři. V odpovědi se mohou vrátit následující chybové kódy: • `EDIR_UUID_DOES_NOT_EXIST` - záznam se zadaným identifikátorem uuid neexistuje (tj. nedá se vymazat). • `EDIR_UUID_INVALID_FORMAT` - uuid nemá platný formát, který je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány jako prázdný uuid.

Příklad odpovědi

```
{ "success": true, "result": { "series": "6423407687606431951", "users": [ { "uuid":
"01234567-89AB-CDEF-0123-456789ABCDEF", "timestamp": 39 }, { "uuid":
"76543210-68FF-18CA-3210-FEDCBA987654", "errors": [ { "code":
"EDIR_UUID_DOES_NOT_EXIST" } ] }, { "uuid": "76543210-68FF-18-3210-FEDCBA987654",
"errors": [ { "code": "EDIR_UUID_INVALID_FORMAT" } ] } ] } }
```

První záznam v adresáři je úspěšně vymazán, vrátí se jeho **uuid** a **timestamp** změny.

Druhý záznam neexistuje (neexistuje záznam s daným **uuid**).

Třetí objekt má špatný formát **uuid**.

5.14.5 api dir get

Funkce **/api/dir/get** slouží k vyhledávání pole záznamů v adresáři a jejich zadaných polí.

Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- POST

Požadavek
Požadavek obsahuje parametry ve formátu **application/json**. Více informací o jednotlivých parametrech záznamu v adresáři a jejich objektovém zobrazení najdete v kapitole **api/dir/template**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
fields	Ne	pole řetězců	všechna pole s jinými hodnotami než výchozími	V odpovědi musí být specifikována jména požadovaných polí, pokud ne, vrátí se všechna pole s nevýchozími hodnotami, pokud se zadá prázdné pole, vrátí se všechna dostupná pole. Přehled všech dostupných klíčů v JSON definici záznamu v adresáři najdete v kapitole **api/dir/template**. Neznámé názvy polí budou ignorovány.
users	Ne	pole JSON objektů definujících uuid	-	Pole musí obsahovat alespoň jeden objekt s klíčem **uuid**, který definuje záznam, jehož pole se mají vrátit. Pokud klíč chybí nebo je zadáno prázdné pole, vrátí se prázdné pole.

Příklad požadavku

```
URL: https://192.168.1.1/api/dir/get JSON { "fields": [ "name", "email",
"callPos.peer", "callPos[1].grouped" ], "users": [ { "uuid": "01234567-89AB-
CDEF-0123-456789ABCDEF" }, { "uuid": "76543210-68FF-18CA-3210-FEDCBA987654" },
{ "uuid": "76543210-68FF-18-3210-FEDCBA987654" } ] }
```

Pokud není v adresáři žádné pole se zadaným uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení odpoví chybovým kódem (viz dále). To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654

Jestliže je v adresáři záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, jeho zadaná pole (v příkladu název, e-mail, telefonní čísla všech volajících stanic a pro druhou volající stanici i skupina) budou vrácena. To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Identifikátor uuid 76543210-68FF-18-3210-FEDCBA987654 má nesprávný formát.

Odpověď
Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole objektů, které obsahují klíče a hodnoty z výsledku požadavku (viz následující tabulka).

Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče JSON odpovědi v poli **users**

Název klíče	Typické vrácené hodnoty	Popis
uuid	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) nalezeného záznamu.

Název klíče	Typické vrácené hodnoty	Popis
Various keys	různé	Zadaná pole záznamu v adresáři, která jsou vrácena. Viz **api/dir/template**.
timestamp	celé číslo;	Časová značka změn naposledy provedených pro každý vrácený záznam v adresáři. Více informací o použití časové značky najdete v kapitole **api/dir/query**. Časová značka je uvedena, jen když je záznam v adresáři vrácen.
errors	pole chybových objektů	Chybový objekt obsahující pole všech chyb, které nastaly. Objekt **errors** je uveden, jen když se změnu v adresáři nepodařilo provést. V hodnotě klíče **code** je obsažen chybový kód, který ukazuje důvod neúspěšného provedení změny v adresáři. V odpovědi se mohou vrátit následující chybové kódy: • `EDIR_UUID_DOES_NOT_EXIST` - záznam se zadaným identifikátorem uuid neexistuje (tj. nedá se aktualizovat). • `EDIR_UUID_INVALID_FORMAT` - uuid nemá platný formát, který je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány jako prázdný uuid.

Příklad odpovědi


```
{ "success": true, "result": { "series": "6423407687606431951", "users": [ { "uuid":
"01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD", "email": "abcd@abcd.cz",
"callPos": [ { "peer": "" }, { "peer": "", "grouped": "false" }, { "peer": "" } ],
"timestamp": 39 }, { "uuid": "76543210-68FF-18CA-3210-FEDCBA987654", "errors": [ {
"code": "EDIR_UUID_DOES_NOT_EXIST" } ] }, { "uuid": "76543210-68FF-18-3210-
FEDCBA987654", "errors": [ { "code": "EDIR_UUID_INVALID_FORMAT" } ] } ] } }
```


První záznam v adresáři je úspěšně vrácen, jeho **uuid** a **timestamp** jsou vráceny.

Druhý záznam neexistuje (neexistuje záznam s daným **uuid**).

Třetí objekt má špatný formát **uuid**.

5.14.6 api dir query

Funkce **/api/dir/query** slouží k vyhledávání pole záznamů v adresáři podle iterátoru časových značek a jejich zadaných polí.

Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

Metody

- POST

Požadavek

Požadavek obsahuje parametry ve formátu **application/json**. Více informací o jednotlivých parametrech záznamu v adresáři a jejich objektovém zobrazení najdete v kapitole **api/dir/template**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
series	Ne	řetězec	aktuální série zařízení	Řetězec představuje počet sérií časových značek v zařízení. Není-li klíč zadán, je uvažována současná série zařízení. Jestliže se zadaná série liší od současné série zařízení, zařízení vrátí svou sérii, nejvyšší hodnotu časové značky a neplatnou časovou značku.
fields	Ne	pole řetězců	všechna pole s jinými než výchozími hodnotami	V odpovědi musejí být specifikovány názvy požadovaných polí, pokud ne, vrátí se všechna pole s nevýchozími hodnotami, pokud se zadá prázdné pole, vrátí se všechna dostupná pole. Přehled všech dostupných klíčů v JSON definici záznamu v adresáři najdete v kapitole **api/dir/template**. Neznámé názvy polí budou ignorovány.
iterator	Ne	objekt JSON	{"timestamp": 0}	Klíč definuje iterátor pro dotaz (je podporován iterátor timestamp). Iterátor timestamp má celočíselnou hodnotu. Pokud je zadán, jsou vráceny pouze novější záznamy. Poslední timestamp je vždy vrácen kteroukoli z funkcí **api/dir/create**, **api/dir/update** nebo **api/dir/delete** a může být použito jako „poslední známá“ změna. Pokud má iterátor timestamp hodnotu nula, jsou vráceny všechny záznamy adresáře. Pokud je hodnota iterátoru timestamp vyšší nebo rovna aktuální hodnotě časového razítka v zařízení, zařízení vrátí svou řadu a nejvyšší hodnotu timestamp.

Příklad požadavku

```
URL: https://192.168.1.1/api/dir/query JSON { "series": "2229480630597592840",
"fields": [ "name", "email", "callPos.peer", "callPos[1].grouped" ], "iterator":
{ "timestamp": 6 } }
```

Nesouhlasí-li klíč series s aktuálním klíčem series v zařízení, zařízení vrátí svou sérii, nejvyšší hodnotu časové značky a neplatnou časovou značku.

Je-li zadaná časová značka nižší než aktuální nejvyšší časová značka, vrátí se všechny vyšší časové značky.

Zařízení je schopno zvládnout až 10000 unikátních uživatelských identifikátorů. Když se tento počet zvýší, zařízení vrátí klíč **invalid**, který znamená, že adresář má neznámou historii (některé záznamy byly vymazány a nejsou již uloženy v zařízení).

Je-li zadaná časová značka nižší než neplatná časová značka, zařízení vrátí svou aktuální sérii, nejvyšší hodnotu časové značky a neplatnou časovou značku.

Odpověď

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Klíč **users** obsahuje pole objektů, které obsahují klíče a hodnoty z výsledku požadavku (viz následující tabulka).

Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče JSON odpovědi v poli **users**

Název klíče	Typické vrácené hodnoty	Popis
uuid	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) nalezeného záznamu.
Various keys	různé	Zadaná pole záznamu v adresáři, která jsou vrácena. Viz **api/dir/template**.
timestamp	celé číslo	Časová značka změn naposledy provedených pro každý vrácený záznam v adresáři. Časová značka je uvedena, jen když je záznam v adresáři vrácen.

Příklad odpovědi


```
{ "success": true, "result": { "series": "2229480630597592840", "users": [ { "uuid":
"01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD", "email": "abcd@abcd.cz",
"callPos": [ { "peer": "" }, { "peer": "", "grouped": "false" }, { "peer": "" } ],
"timestamp": 7 }, { "uuid": "A6543210-68FF-18CA-3210-FEDCBA987654", "name": "DEFG",
"email": "defgd@defg.cz", "callPos": [ { "peer": "" }, { "peer": "", "grouped":
"false" }, { "peer": "" } ], "timestamp": 9 }, { "uuid":
"044197A7-54AD-7577-6EEA-787A6097263E", "name": "HIJK", "email": "hijk@hijk.cz",
"callPos": [ { "peer": "" }, { "peer": "", "grouped": "false" }, { "peer": "" } ],
"timestamp": 10 } ] } }}
```


Vrátí se tři záznamy v adresáři, které mají časovou značku vyšší než 6 (v tomto případě je maximální časová značka v adresáři 10).

5.15 api mobilekey

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/mobilekey**.

5.15.1 api mobilekey config

Funkce **api/mobilekey/config** slouží ke čtení a zapisování identifikátorů lokace a šifrovacích klíčů pro autentizaci přes Bluetooth.

Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu (sledování).

Metody

- **GET** – čtení identifikátorů lokace a šifrovacích klíčů
- **PUT** – zapisování identifikátorů lokace a šifrovacích klíčů

Požadavek

Pro metodu **GET** se nepoužívají žádné parametry.

Parametry požadavku **PUT** se ukládají ve formátu ***application/json***.

Tabulka1. Klíče JSON požadavku PUT

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
location	Ne	Řetězec o maximální délce 127 znaků	–	***location*** definuje lokaci zařízení pro účely autentizace přes Bluetooth. Akceptován je každý řetězec, který danou lokaci jednoznačně definuje. Lokace je pak přenášena zařízeními 2N a slouží Bluetoothovému autentizačnímu zařízení k vybrání relevantních autentizačních parametrů.

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
keys	Ne	Pole objektů obsahující šifrovací klíče	–	***keys*** obsahuje šifrovací klíče k zabezpečení komunikace mezi zařízením 2N a zařízením určeným k provedení autentizace přes Bluetooth. Objekty v poli mají tyto klíče: • ***type*** – algoritmus, v současnosti je podporován algoritmus RSA, tento klíč je nepovinný, • ***key*** – data šifrovacího klíče (formát DER kódovaný v Base64), používají se 1024bitové šifrovací klíče, tento klíč je povinný, • ***ctime*** – čas vytvoření ve tvaru Unixového času, 32bitové celé číslo bez znaménka, tento klíč není povinný.

Příklad požadavku PUT


```
URL: https://192.168.1.1/api/mobilekey/config JSON: { "location": "LocationUniqueID",
"keys": [ { "type": "rsa", "key":
"MIICXAIBAAKBgQCXmIX1U7wGFW3FiDdhq7BIktIc4lg7X2IMxLE83I75S3BRPL/
7LCAefnMUJL0uyyFdeMpRoOVhVs/iPfnYPNf4AiQO4lIQh8tSKDeat5IfXSMy9zXMyHeBOBg19R+/
uShyJsnLoJoB5MJDowwkOuSMIskK+dA17+3E/
Y+ujhhpCQIDAQABAoGAfzHOVAUp4cDhFbgxH5Y6lun5uZqAhXCGiEgQngxBOhJ97uuV+V0QpgVa8S/
SPAzbtd2/g7YIQB/
i1OVDWJfUbEiuBhr6ZHwk5jfCfF0KkmTQtEBd4bbCz+FWyoy19DUXdsLNMf8GW4eWhooX+NCqc2sfloO4Nz+S
pXmqpsMIc/ECQQDk63xVnRqPCgG3fqpLVGWkQl9wmYAIUP8MrdOAfRYfSC/
LrjX55lCRj4mAnSzRQfNclSEz2mITkoaCcjn1lTmXAkEAqYdjMEhIrg4LpYzqZDOF6v6w/
sUcLkepktTBYCFFV+YgOrlPr7akR8XtED8X/6QHwWciphp/5OBoJ/
KRAWWGXwJAZ0YNe5o6pxk+mQed0AotKKOA5w15A0d3KMMqzaag2k/
4sAzR8QGEi4aT4+AEngsAvV3R8tCsumO6JxNdLnu51QJAO6abzB1jFXtztajDwMYwVOORO9P3eoFUtYmPEVgj
oi4jIQabd2R4oZiPNaw9sYHyCKdVlcS1Q7+CZqv/
QdKLWQJBAKeoGxqcpDHvMtzgcSj3lZz0Z8dWmgtTF7QO5boHhxtZ1SEo3MvlicVue8U1tV2XjUR6r7YueuusM
9GxhBqr5YI=", "ctime": 1608047606 }, { "type": "rsa", "key":
"MIICXQIBAAKBgQCfyMHsTjPKf3Dv00gwMrQAR5UZrpt3tBy3kBvPv4R4o9H7Zzse7+yKwfPTddKJQOL1IrCX
06Zo8SZAMotjjpMy1M9K27ZB95YtAYiGLLRWeLAJUkL4gixgkHeS+T8uQxLW7/
etqwU00uPmd94ZEZy226CHdKQW3zge2WEtuQ5oCwIDAQABAoGAZCp6RyUPGpahufZ9fpmKddJqCduH4paqmfh
hNu8coHQyIqQoT9CgPKwxqhJmlVxz6rCAe+1WmNrz27LT5uluJKViU0XnLV7FHG2smagjQ3rPepgOGcayphui
IlHikaBCafxnCRV/
E1Ifg08d1xK4cK858yMjpoEgDdEJi0R2qmECQQDXqtwGiXYSRnZzR90eCjrip6IIQqJuARE91LOLyOhkPzCiP
Pf2IrT1JQsw6Tu0ZTm3NJzZ0VSEdZU6s2NcKHsnAkEAvap5GacBi9EZ9lsiaQj/
dVA6LbUnBCo7qwRj7SUYw6ikCvmcLjdpjR80twj3FTAXB0sTeWgyT42HmBpPX2dKfQJBAMc5Ml9nhAaFyM3dS
MmDMbpGmEuBIoLzwXWYkvNB+EsChG6aW4Snsvnx6lCYY2rVR2eR1oLv+F8UL3I2XEa5rmkCQBZXhnxnF9+Iei
5y/dKxpKYFFVVdCYOMFgtHMR42SHyD2Q8R6Dvpex2Ml4EYJULxr0TEqz6Z75M/
cMGSF9d9K2ECQQDEffSJoyjYwY2rGbPX8N5d9yrp3HLRbH4RjFGR0zCbSaA+PTQwxu2q1Asd8g7LN95Umyvli
ddJgayDIwnJSGse", "ctime": 1608044538 } ] }
```


Zařízení 2N umožňují použití maximálně 4 šifrovacích klíčů najednou. První šifrovací klíč v poli se považuje za primární šifrovací klíč a ostatní klíče jsou sekundární. Jestliže se Bluetoothové zařízení autentizuje jakýmkoli sekundárním šifrovacím klíčem, zařízení 2N je vyzve k nahrazení tohoto šifrovacího klíče klíčem primárním. Proto by se vždy měl na začátek pole přidávat nejnovější šifrovací klíč.

Je-li předloženo pole kratší než 4, jsou chybějící šifrovací klíče vymazány (nahrazeny prázdným objektem).

Klíč **type** není povinný. Je-li vynechán algoritmus, zařízení 2N automaticky předpokládá použití algoritmu RSA (***rsa***).

Parametr klíče ***ctime*** není povinný. Je-li čas vytvoření vynechán nebo neplatný, zařízení 2N zobrazí v konfiguračním webu datum 1. ledna 1970 00:00:00 a pro tento šifrovací klíč nevrátí ***ctime***.

Odpověď

Odpověď na požadavek **GET** je ve formátu ***application/json***. **Výsledný objekt** obsahuje klíče ***location*** a ***keys***.

Odpověď na požadavek **PUT** neobsahuje žádné podrobnosti. Např. když je uvedena neplatná hodnota šifrovacího klíče, klíč nebude zapsán bez notifikace.

Tabulka 2. Odpověď na klíče JSON požadavku GET

Klíč	Typické vrácené hodnoty	Popis
location	Řetězec	Identifikátor lokace zařízení 2N. Podrobnosti najdete v sekci Požadavek.
keys	Pole objektů obsahující šifrovací klíče	Délka pole je vždy 4 (za chybějící klíče se vracejí prázdné objekty). Podrobnosti a struktura objektů v poli jsou popsány v sekci Požadavek.

Příklad odpovědi na požadavek GET


```
{ "success": true, "result": { "location": "54-1046-0745", "keys": [ { "type": "rsa",
"key": "MIICXAIBAAKBgQCXmIX1U7wGFW3FiDdhq7BIktIc4lg7X2IMxLE83I75S3BRPL/
7LCAefnMUJL0uyyFdeMpRoOVhVs/iPfnYPNf4AiQO4lIQh8tSKDeat5IfXSMy9zXMyHeBOBg19R+/
uShyJsnLoJoB5MJDowwkOuSMIskK+dA17+3E/
Y+ujhhpCQIDAQABAoGAfzHOVAUp4cDhFbgxH5Y6lun5uZqAhXCGiEgQngxBOhJ97uuV+V0QpgVa8S/
SPAzbtd2/g7YIQB/
i1OVDWJfUbEiuBhr6ZHwk5jfCfF0KkmTQtEBd4bbCz+FWyoy19DUXdsLNMf8GW4eWhooX+NCqc2sfloO4Nz+S
pXmqpsMIc/ECQQDk63xVnRqPCgG3fqpLVGWkQl9wmYAIUP8MrdOAfRYfSC/
LrjX55lCRj4mAnSzRQfNclSEz2mITkoaCcjn1lTmXAkEAqYdjMEhIrg4LpYzqZDOF6v6w/
sUcLkepktTBYCFFV+YgOrlPr7akR8XtED8X/6QHwWciphp/5OBoJ/
KRAWWGXwJAZ0YNe5o6pxk+mQed0AotKKOA5w15A0d3KMMqzaag2k/
4sAzR8QGEi4aT4+AEngsAvV3R8tCsumO6JxNdLnu51QJAO6abzB1jFXtztajDwMYwVOORO9P3eoFUtYmPEVgj
oi4jIQabd2R4oZiPNaw9sYHyCKdVlcS1Q7+CZqv/
QdKLWQJBAKeoGxqcpDHvMtzgcSj3lZz0Z8dWmgtTF7QO5boHhxtZ1SEo3MvlicVue8U1tV2XjUR6r7YueuusM
9GxhBqr5YI=", "ctime": 1608047754 }, { "type": "rsa", "key":
"MIICXQIBAAKBgQCfyMHsTjPKf3Dv00gwMrQAR5UZrpt3tBy3kBvPv4R4o9H7Zzse7+yKwfPTddKJQOL1IrCX
06Zo8SZAMotjjpMy1M9K27ZB95YtAYiGLLRWeLAJUkL4gixgkHeS+T8uQxLW7/
etqwU00uPmd94ZEZy226CHdKQW3zge2WEtuQ5oCwIDAQABAoGAZCp6RyUPGpahufZ9fpmKddJqCduH4paqmfh
hNu8coHQyIqQoT9CgPKwxqhJmlVxz6rCAe+1WmNrz27LT5uluJKViU0XnLV7FHG2smagjQ3rPepgOGcayphui
IlHikaBCafxnCRV/
E1Ifg08d1xK4cK858yMjpoEgDdEJi0R2qmECQQDXqtwGiXYSRnZzR90eCjrip6IIQqJuARE91LOLyOhkPzCiP
Pf2IrT1JQsw6Tu0ZTm3NJzZ0VSEdZU6s2NcKHsnAkEAvap5GacBi9EZ9lsiaQj/
dVA6LbUnBCo7qwRj7SUYw6ikCvmcLjdpjR80twj3FTAXB0sTeWgyT42HmBpPX2dKfQJBAMc5Ml9nhAaFyM3dS
MmDMbpGmEuBIoLzwXWYkvNB+EsChG6aW4Snsvnx6lCYY2rVR2eR1oLv+F8UL3I2XEa5rmkCQBZXhnxnF9+Iei
5y/dKxpKYFFVVdCYOMFgtHMR42SHyD2Q8R6Dvpex2Ml4EYJULxr0TEqz6Z75M/
cMGSF9d9K2ECQQDEffSJoyjYwY2rGbPX8N5d9yrp3HLRbH4RjFGR0zCbSaA+PTQwxu2q1Asd8g7LN95Umyvli
ddJgayDIwnJSGse", "ctime": 1608046389 }, { "type": "rsa", "key":
"MIICXQIBAAKBgQCwWXVu2CNcUFgoqQBQ5NjaLJVEWuAFryK/
h9jfNe+qDufFS+itWsWfvyvMkUhhiidPCpgoOgqEipkYa0q3maPKPS4CJXZBFo++JSzsgw6a/
VxH0n8joHfJf6nIEcCGcuMAa/
HOEoOZq7uL7n2jTsyVnnDbYClXENh4Np9izSX23QIDAQABAoGBAI5iDFDMrfAw5p0dpqWpv/
SXnoUsIkg0mYeu9UlzUOgrVLKVkW22Jm3OelyWyKwIUaid0zBXfHp7NRTkO9V1dSnS5CnuO73tye9MV5TeLqj
MSBVCSPZWJK//
hu1VaRAL9UTZc+1e277lOB8c1Fup4uxR4b757brrclNKjT1U4Hh5AkEA4mFz+IrgTtdiLNLQdww5B3ZELmaOl
+lkYgC50hvqy2TDNGGiKGPqYmEd/4ySHBmaGnoh9ZFxnC/
ItrNEXBGdawJBAMdsOd2qDdbOSie2TpsGJs5eEUrLX6yW/
w+siO4SXczCTnJXckZyj79eEOcnrTRK+SuDsN+8+wmO3b9CZqxOxtcCQQCUkukOAfddRzaDvIhc2YTERPZSjb
SgNulO+LL8Fp5uht8mjb1jTNATaTHK+nMaRiNBpU6MYLxziVjtr5H56wWpAkA0fXYVtEcEPTQjk8bI4yufsf7
XMwSxxuTH2WAJWeg6lwJS8lVv2YOgmT/VuAnM89b17ynFGlbxQxt21iF0RR/
tAkA0nJMkbD3daWYAutdOQjzemae3O0O9r90NZ/
Khj5tQpv8gLe6EEpyFUNQNqodNoTmkIJJmPLlBjyx+zTspdE+C" }, { } ] } }
```


Výchozí hodnota parametru ***location*** je sériové číslo zařízení 2N. Chcete-li přidat do jedné lokace více zařízení, patřičně tuto hodnotu změňte.

5.16 api lpr

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/lpr**.

5.16.1 api lpr licenseplate

Funkce **api/lpr/licenseplate** slouží k řízení přístupu pomocí rozpoznávání registračních značek vozidel (SPZ).

Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu (řízení).

Metody

- POST

Požadavek

Požadavek obsahuje parametry ve formátu **application/json**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
lprUuid	Ano	uuid řetězec	–	Identifikátor uuid události rozpoznání registračních značek generovaný systémem Rozpoznávaní registračních značek.
lprID	Ano	ID (číslo)	–	Interně určený identifikátor označující jednu registrační značku v rámci jednoho zaznamenaného příjezdu vozidla. V rámci jednoho příjezdu může být tatáž registrační značka rozpoznávána vícekrát. V případě, že kamera získá z dalšího rozpoznání přesnější data, upřesní se plateText, zatímco lprID zůstává stejné. Každá jednotlivá událost rozpoznání generuje vlastní lprUuid.

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
accessPoint	Ano	0 nebo 1	–	Signalizuje, že vozidlo s detekovanou registrační značkou vjíždí (0) do objektu nebo z něj vyjíždí (1) – podle toho se pro událost v zařízení 2N použijí nastavená přístupová pravidla.
plateText	Ano	řetězec registrační značky	–	Text rozpoznané registrační značky, který slouží k identifikaci uživatele v seznamu uloženém v zařízení 2N.
lprDir	Ne	0 až 3	–	Určuje detekovaný směr pohybu vozidla/registrační značky dle nastavení směru na kameře. Možné směry: • 0 = Unknown • 1 = Undefined (In nebo Out) • 2 = In • 3 = Out
plateImage	Ne	obrázek šifrovaný pomocí base64	Bez obrázku	Obrázek, na němž byla rozpoznána registrační značka. Maximální velikost obrázku je 256 kB.

Příklad požadavku

```
URL: https://192.168.1.1/api/lpr/licenseplate
JSON:
{
    "lprUuid": "bc4baad9-d2cd-4706-986f-b942963bfa9f",
    "lprID": 143289,
    "accessPoint": 0,
    "plateText": "ABC123456",
    "plateImage": "/9j/4AAQSkZJRgABAQEASABIAAD//gATQ3JlYXRlZCB3aXRoIEdJTVD/
4gKwSUNDX1BST0ZJTEUAAQEAAAKgbGNtcwQwAABtbnRyUkdCIFhZWiAH5AAMABIACgAHAA5hY3NwTVNGVAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA9tYAAQAAAADTLWxjbXMAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA1kZXNjAAABIAAAAEBjcHJ0AAABYAAAADZ3dHB0AAABmAAAABRja
GFkAAABrAAAACxyWFlaAAAB2AAAABRiWFlaAAAB7AAAABRnWFlaAAACAAAAABRyVFJDAAACFAAAACBnVFJD
AAACFAAAACBiVFJDAAACFAAAACBjaHJtAAACNAAAACRkbW5kAAACWAAAACRkbWRkAAACfAAAACRtbHVjAAA
AAAAAAAEAAAAMZW5VUwAAACQAAAAcAEcASQBNAFAAIABiAHUAaQBsAHQALQBpAG4AIABzAFIARwBCbWx1Yw
```

```
AAAAAAAAABAAAADGVuVVMAAAAaAAAAHABQAHUAYgBsAGkAYwAgAEQAbwBtAGEAaQBuAABYWVogAAAAAAAA9
tYAAQAAAADTLXNmMzIAAAAAAAEMQgAABd7///MlAAAHkwAA/ZD///uh///
9ogAAA9wAAMBuWFlaIAAAAAAAAG+gAAA49QAAA5BYWVogAAAAAAAAJJ8AAA+EAAC2xFhZWiAAAAAAAABilw
AAt4cAABjZcGFyYQAAAAAAAwAAAAJmZgAA8qcAAA1ZAAAT0AAACltjaHJtAAAAAAADAAAAAKPXAABUfAAAT
M0AAJmaAAAmZwAAD1xtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAAgAAAAcAEcASQBNAFBtbHVjAAAAAAAAAAEA
AAAMZW5VUwAAAAgAAAAcAHMAUgBHAEL/
2wBDAAMCAgMCAgMDAwMEAwMEBQgFBQQEBQoHBwYIDAoMDAsKCwsNDhIQDQ4RDgsLEBYQERMUFRUVDA8XGBY
UGBIUFRT/
2wBDAQMEBAUEBQkFBQkUDQsNFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQ
UFBQUFBT/wgARCABMAIADAREAAhEBAxEB/8QAGwAAAgMBAQEAAAAAAAAAAAAAAAAcFBggDBAL/
xAAaAQACAwEBAAAAAAAAAAAAAAAAABQMEBgIB/
9oADAMBAAIQAxAAAAHUx4kXairWqwAAAAAAAAAAFlrWHgkcKhqtzRps56eO5KPsAD5CGnh689S0Unk75j5I
/rz2Wilh5odHZvQxU0KGfJNf4/WS0Unk65An4Js6aJDfaF5qKmPP3zGezycd3xsbH6vFe1x7yRO+/
ghnyS207dds1/
J3x35707l9HUrlS+0LufdAk0VnXyfbqlkzXbFx+rxXtce8kTvv4IZ8kutG601bK60bluqWoCxBmnS55/
IHeetAjbalp8niGfJdi4/
V4r2uPeSJ338EM+SbLxmup1yosGi6j3acZNCBrTJajPWgR3Cpaa6lplnU5rYmP1eK9rj3kid9/
BDPkm0MXr1g0XVW1VAAqVurorOvs9aBH9nut8lp1C3VsddfxbtMg8kTv2cdIHQImqpa3ijcAAAWzJdZK1lZ
M10NPA7Ujmdrzr5hQWjRa9kLtsKWmRddlouaEAAAAAAAAAACRil1vktRa6tnh75FSxgAAAAAAAAAASkUno8
9/8QAKhAAAQMDAwIGAgMAAAAAAAAABQADBAIGBwEVFzE2EhMWMzQ1EBEUMDL/
2gAIAQEAAQUCrr0bpL5Kpad5KKLkoouSii5KKLkoouSii5KKLkoouSiiE5L8bjblLzeSCVUU
WmY7smvZCK2QitjIqoMQbpTbdb1eyEVIgSomi001q1rDz2qFjUlXIg5S6q0ILMIBKPDoT3qkSvVIlRpTM1n
IwtmEQx2MaYDVOU06vx25TJ4btBeD824Poli738pdVHusrFYlSnZshNMuP1Y/
GyRwnKP8Aqw+17vrquSy5tU63MlRvKNQfnXB9EsXe/
lLqgNpzT+kTGMalRLMDxE2y1GomXLgK67h9QkLD7Xu7uTGMnxQMnRvFBgfOuD6JYu9/
KXVWw1Szb1w37SGnTMglpKlk5c782H2vd3cmNJWkO45UaizVq1rQvs9aBq1bVtX04FjcnQ1ydDX
J0NcnQ1d9ys3HVbt9Rgog1PpKFdNf1qLyVWyxydDVy3s+dZWLqNfMuELQeGERkkVI/
sgj5BORbQOkAMTzDcijYhq2IatiGrYhq2IatiGpmO1GpX/
xAAsEQABAwAJAgcBAQEAAAAAAAABAAIDBAUQERMVM1FSEkIhMTJBBcYGRIhQw/9oACAEDAQE/
AVLWAaboxeswl2CzCXZzCXYLMJdgswl2CzCXYLMJdgswl2CzCXYLMJdgoqwBN0gTH0j3qQ58J+3myr5C5S
Sm5XOOp0L1iGoIqoMQbpTbdb1eyEVIgSomi001q1rDz2qFjUlXIg5S6q0ILMIBKPDoT3qkSvVIlRpTM1n
ORX+iLkF/
oi5BBwcLjqwJzmswZ6DGrxS0vqzn8UBwqFRNEy5Ek0bnZpGbQYe5JwJmLQRrdZRmvJCxWAHsmrzTpBVcv
YqkUwzDpb4CyrR6JP7YeMB6+4+UQbTXmfQIdAwBtt+TtnXkaunD+6t9BtN+lZjmCWuUNgOaCrtFJwPkvXnFP
13KiO6ZmKxm0oDHUONtOUsAnaP+zxP3U/oWDvg7PRwSz0T8ww7Bn17fpQ1rz6j00DE0CqPS6FpGP9Ah3t
/xAAvEQABAwEFBgYDAQEAAAAAAAABAAIDBAUQERMUEhUhMUEgIjJCBsQgJDEzUv/aAAgBAgEBPwA
AeKhs0uGMhwWWxalZbFqVlsWpWWxalZbFqVlsWpWWxalZbFqVlsWpUtm4Dm4DGIojDwDP0xqPi8/
kLj7X0c1vi4/O+tfyqw7rfxW73EhoxK38RlQ/CpEwf8Adm/VT6a5LfnEu9xJlCXZwHq0Y/tWNz3+
Q2+u1bGvOS1jA2N8wHIEtLvvHtVw9NhHrP1ntuVTXi2x9iHw/kpUcdt7UHCuj1xHPuMA+qUWlc/Eyo0V
y7rYhFYGvOIdoE6Ww3zT2dLZ9cpdzDiZ6V+M2D+Hmk7DFTIa8xgYgZt6fYzHT6b3e7L8b8yBXqO/
kjlmqkY6TpmcygRkUr8ZLzyMTwcNPz0W/55jHeT89bJLiBTGt1pR6dWBd9kB0jSRVnzDSn6Vn9Qd3BpxUtK8Y
W3Ut1dHh8fY5IU/u2N5ZhzZkrr5tB1y1kM1N5jDSRVESybJJzK1HSDpdf2yaD5DrdLTSEsc8y0uyK4hOXxwh
Zpi+QGzFE/LzHt2vg5VF57YxlGDjjFLXXN8qHH5xO2yiB6v9lxNkjbtxg00MD+HQoG4wZyjXe7dZrCK2hk
1D5AqeYwP2wo5WSjaxcfo5LzqPFuc5/KbLlvLMsuqbj+3/aQ2+T2b7KbQ0L7h6wa2VHbiwQ6S/XWeLRvz
9XeOtSWiRJ2vBu3fMIbt9rdt+SWJnYiq6ZTUJTxDuGVKGLbNwFHkG3YZKaZUN2ev8Mu7AsXfOeh7T7vM0tLU
LzHhM/9aPz2aDB0rA7HvgiRvh0rMgzBQQOnt+bDo9m91GZYx6HEGVMdQPk
EAEIQAAECAwEKCAwFBQAAAAAAAIBAwAEERIFECEzNUFRFRkpREyIxNHORsbIUMmFxcnSHRNBDJDxEngOUMYBhwcI9iGo7Zih8T
```

```
BSVGPw/9oACAEBAAY/AlIlQRTCqrBN3PYR1E/
Vd5F9kYqV1C3xipXULfGKldQt8YqV1C3xipXULfGKldQt8YqV1C3xipXULfGKldQt8YqV1C3xipXULfGKld
Qt8CF0JdAFf1Wc3sgXAJDAkqhJnhqWBbKzBcb0U/
5L1lls3S0ANY5hNbEoyfNbEoyfNbEoUikZkRTOrJXkBsVM15BFKrHMJrYlFX5Z1lNLgKN6iYVWCM5KZERSq
krRUS9MShrXgFRQ8y5oub8T7b0orYIhOgjhlnVVhWn5xppxOUVLCkZQY1oygxrQjrDgvNryEC1SGJhoUDwh
FtImlM/
zhJyyivvkvG0Ii0pFFJEXyrBNOgjjZJRRLPEzKp4oFxfNypEv0g9sXR9Xc7q3roeiH1i5vxPtvAy1OEDYJZ
EaJgSDffPhHTwkS571loCcLQKVh1JlsmlcctCBctKRc34n2xKe/
wB5YnrSqtDpEqRlaMagqr5Fhp7M6180X+Il+kHti6Pq7ndW9dD0Q+sXN+J9t5XGrLTCLThXPpH4mccc8jY2
d8YJMXF0u8btijYA0Cf2pRIXhZ1q0n5QW0vyhHAFQYbSy2i8vniU9/vLE/0kTkvXCDiH1p+0ScxnBxQ60/
aJfpB7Yuj6u53VvXQ9EPrFzfifbeuegpROBEuvDDsm3Kq843SpEVE5KxxDblk/1hvj8RMuveQzVb8p7/
eWJ/pIfZzONV9qL/
MTVOUKH1LEt0g9sXR9Xc7q3roeiH1i5vxPtvXO6AOyHpuXdZsO0wOKqKmCmiMZLa67oxktrrujGS2uu6GUm
SbLha04Na8n8xKe/wB5Yn+kiSNVoKlYX2pSHWHEq24KgXmht4p0jbA7Vjg6KvtrF0VX/
HPsvXQ9EPrFznPyopp2brySr7PhDA+KqLQhjmj/AFpHNH+tI5o/8o5o/
wDKJVWmja4K1W35aboZlHJd1wgrxhpTCtYmZoBUBdKqIsVTAsC3Oy/
DEKU4UFoq+yOZv9aR4O234PLVwpWqleugdOLQEr1w5LEtk/
GA9BQrMy0rZpp5F839VGZZonXF0ZoFitp1eM4Wlb1l1sXB0GlYyfK7EYyfK7EYyfK7EYyfK7EYyfK7EYyfK
7EYyfK7EYyfK7EYyfK7EYyfK7EYyfK7EYyfK7EYsstg0OgBpe//
xAAnEAABAgQGAgMBAQAAAAAAAABABEhMVHwEEFhkaHBcYEwsdHxIP/
aAAgBAQABPyEF8SSYAVUJJrxHiCLex8tGjRo0aNGjRo0aIIaYXG8jct4PpDxGOuAMiET+TgzZJG5wZ6pfHs
FdHSvLpWt0o4OCEA4whgccD0r66QSZ4AEO4wGQEhgBMowFgWAJklsDHBKjk/
8AYHfE7RpIQR4nhSl9t7BjlOePtBGNBY7AmHJmyiQAmIsgNiUDFaQYTW6yOATjiYsax9BCvFGKiwVxnBJuB
DJCSOgIzAsNd38tgmOSZYGA5GUkcLNkctOUapzIMGCJOHLHMQDhkUEIWOoB4S4UYqLBXGcJDkELE5gM0ykn
RDykRr9PGDhUnQAJDa4cXctJA9sy1PQXLTstAhnJGaNKNqmHR5bdRiosFcZxBiENUHclVCZPwgDEmeicwc6
x3ciZL6XYG0seWnZaBGJz/oG4KHanb/Q/DqxUYqLBXGcRQLwuKjphJoAMhPL/
ABPnyykoTf5Du4FC5adtoE+bs3mDkhMRjAqDFBs+FpgXAP4IjDBncxhYKqEAjs6l4+2BDxRhYpGYV89q+e1
/e/a/v/
tNk6DgjQZDHTeRx3N5WcELYgiCIQFwRkj64Qfkln7Vw9ogf7pdyc00wiU3AoAjzoSpH7HtEfZBkGpZj5cue
gw1E5BNjGt5FpoJYFxnyobFX90r+6V/dK/ulf3Sv7pX90r+6V/dK/ulf3Sv7pafpg7DD//
aAAwDAQACAAMAAAAQ5222222dFT9dTNQqNhSVn9+sNdo2ZCesNp6Jf3+sNzOtfOesNwSWfGJsVSW3WR3rWt
ttttt9iSSSSSSU/8QAJxEAAQIEBAcBAQAAAAAAAAAAAQARMWGx0RAhkfFBUXGBocHw4TD/
2gAIAQMBAT8QJADlGnWDD98LaDdbQbraDdbQbraDdbQbraDdbQbraDdbQboKYTENEAARBGGTMvHH
1gVYhMg63pZbkst6WTMdZbA4y5ktyWQFyBMEYAEmEURCQAJG2HRQdCoO/
1g1WIBPUhMiAzK3YLdgntuJIHzFz9muhwRm7wWTgI6A4KEHgIdDmvIFV5qmEX1xUHf6wGWwGQyFkTkcnA2x
CZB0agWJLt2C9vpVlSiJcMGoFGShpkicCIof1eUKrzVMIvrioO/
1gGblzH1zUYz0yuoeB651QABgGTmwCOHHSKGEyl1WVKpKBZjZg1H4UA8cC2o/F5AqvNUwi+uKg7/
WAAxyURK5I7CDzWRN6BdP7575aQxrKlUlAmOeD6H9Rhhwz0LryBVeaphF9cVB3+sPDURwFnzfk3IrfDZb4b
LfDZZ2g5oS7DmqypVJQJ/
EsDlqLsiuEckFOkAuze0TFmphF9cUc8OCRq1sCAdBA8V8wuvmF18wuvmF1ksRmjNrIzIkPBuZPNAEYGyBJO
E2OSOI49l8wum5VDgSVl7R12jyKyvqHp/
XNuP0UJjmYkzwADA6lNApTQKU0ClNApTQKU0ClNApTQKU0CAsDYf/
8QAJxEAAQICCgMBAQAAAAAAAAAAAQAR0fEQITFBUWFxobHwgZHB4TD/
2gAIAQIBAT8QAJMEAaYLfc1PBBTwQU8EFPBBTwQU8EFPBBTwQU8EEcjnIxRDIGIQyrO5s+0BW
AGZZSZFSZFSZFAxYepGh8LDNSZFFGCWRBoJADlEgSJzEaNcZqJrs80P1sJA0CeEIxZSxSxFhgOBRR3qNoXq
2R6ZsKtQ7oAmwIaOxCdK0ivUVFbc8LqMRR0NF2eaCd0TWazFA4mCgWyAzqQ0SdgxPkqz0vXE4IQbF31BQYG
v2HQ7wB5E1tzwuoxFHQ0XZ5oO22AfcFYkNa4K2CGlXCIOTlMTjG+weynQj1hguJwW3+lFxUN6P6jjGh/R/
VsjwuoxFHQ0XZ5oMccWxZATUHJza2SrcTqMGViY0H2nicFs/
pWSx5H4hnG9x7EVsjwuoxFHQ0XZ5o3jlH7O6xsXxClgipYIqWCKqkIqW5+TguJwWz+lHAAcgP6L8K2JBceE
BAAkM7/
P1A7GDmjoaIIvZY+njQUOi3ArvGC7xgusYLrGCCsDUsyeKGnALWvcAEQxyEQCGKCdmDcbvK7xgviQigDXlf
xBdYLxiEJuBuNf6lrAdsREdQsAyoKubFTJFTJFTJFTJFTJFTJFTJFTJFTJFTJFTJFTJFHnSTnR//
xAAkEAEAAQMEAgIDAQAAAAAAAABEQAhMRBBUYFhoSBxMJHxsf/
aAAgBAQABPxAvDyhiVLYAvNJ7hnAN3C8mTj8vnz58+fPnz58+fKctkm85Z5Cj5UduBYqUGRGhsipogkzZlc
gm+lqVYS8wLq3lj1MqCQxuVYmiuZkSeACvWht3UyOOBKXQF7B5RsAGWgosTWlCIAKrY0YHfM5keB1fQ0DBP
```

```
gQxORiQcAULChPSAnBIjfZK/jq/
n6kgwAoGEk3GyUe9dkpQGETcssrR6KISwQ2FZyt8EYTYon6GmNqOqZEaLkx1TSu7GXkau+DBJ56GgmuFQBA
K2wb0Gi4QAAlADAYND7Rcc+haSit9fMbzVZhtiiurHTxOlcwIXOYYjgOKRJDRLxW61mZB0Zfaj4rRJ56GhH
w/gbACs3wbTNEntl+hZmfu1Q+6En7E9CoIEx4+gAo4HsgPF8P3FSOcg5JkphItLB5NF3talYIF7YDt/
vzRjaqHY9+9GR+DEnnoaB/
VgRJ1+1XuoKZAASIIAMzlNHPYwFDwu+yKFxWCL4bDo+C72tSoAM8q0Rkd69iV/
et3wGJPPQ0AVC+OQH2VALyIviwDfMmYi0/
BQbv25MCRwi2InfRd7GqYDYMDAnx6VM6aFCqgdmHNEQRAApKMhKCTEULN5LyMdqGrxgLwXMnZ+nROGk4Sra
SZUFElukB8DBh4kHcnG3o3HEs80g20tA0SmwTqpYMLE4EMW2o/
4k4UYR2aHLyzhYSR5IJ4NEgtSnXdIiACDAyErpZAE8WZKDr/AEqKgE4iMnhFI2W9LL4JIH1fMdw/
lYQIX4dIuVCg674kgSWwA+pyumSPjX0J+UGDBgwYMGDBgwYMiFYNegNP/9k="
}
```

Odpověď

Odpověď je ve formátu **application/json**.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
success	`true`, `false`	Hodnota je `true`, když je požadavek úspěšně zpracován. Dojde-li k chybě, hodnota je `false` a v popisu chyby `error` jsou k dispozici další informace.

Příklad odpovědi

```
{ "success": true }
```

Může dojít k různým chybám (např. chybějící povinný parametr). Chybový kód 13 (příliš velká data parametru) znamená, že odpověď nebyla zpracována a je nutné zaslat požadavek znovu s menším obrázkem nebo bez obrázku.

Následně obdržené duplicitní platné požadavky jsou ignorovány (v paměti se uloží jen posledních deset úspěšných požadavků). Je možné pokusit se poslat požadavek znovu, když nepřijde od zařízení 2N žádná odpověď, bez riskování otevření duplicitní bariéry nebo duplicitního uložení události.

5.16.2 api lpr image

Funkce **api/lpr/image** umožňuje získávat obrázky ze systému rozpoznávání registračních značek (SPZ).

Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu (sledování).

Metody

- GET
- POST

Požadavek

Požadavek obsahuje parametry ve formátu URL.

Tabulka 1. Parametry požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
plateText	Ano	Řetězec s textem registrační značky	-	Text rozpoznané registrační značky, který slouží k identifikaci obrázku, který má zaslán (dá se uložit až pět obrázků). Patří-li ke stejnému textu registrační značky dva nebo více obrázků, vrátí se ten nejnovější.

Příklad požadavku

```
URL: https://192.168.1.1/api/lpr/image?plateText=ABC123456
```

Odpověď

Úspěšná odpověď je ve formátu **image/jpeg**.

Může dojít k různým chybám (např. chybějící povinný parametr). Chyby se vracejí ve formátu json s odpovědním kódem 200. Není-li k předloženému textu registrační značky přiřazen žádný obrázek, vrátí se chyba 15 - "data nejsou k dispozici".

5.17 api accesspoint

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/accesspoint**.

5.17.1 api accesspoint blocking ctrl

Funkce **api/accesspoint/blocking/ctrl** slouží ke kontrole blokování přístupu pro jednotlivé přístupové body.

Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu – řízení.

Metody

- GET
- POST

Požadavek

Požadavek obsahuje parametry ve formátu URL.

Tabulka 1. Parametry URL požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
id	Ano	Celé číslo (0, 1)	-	Určuje identifikátor přístupového bodu, který se má kontrolovat (0 pro Příchod a 1 pro Odchod).
action	Ano	Řetězec (on, off)	-	Určuje, zda má být blokování daného přístupového bodu zapnuto / vypnuto.

Příklad požadavku

```
URL: https://192.168.1.1/api/accesspoint/blocking/ctrl?id=0&action=on
```

Odpověď
Odpověď je ve formátu **application/json**.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
success	`true`, `false`	Hodnota je `true`, když je požadavek úspěšně zpracován (tj. blokování přístupu je v požadovaném stavu bez ohledu na nastalou změnu).

Příklad odpovědi

```
{ "success": true }
```

Může dojít k různým chybám (např. chybějící povinný parametr). Chybový kód 18 (přístupový bod nepovolen) znamená, že požadavek nebyl zpracován, protože daný přístupový bod byl v té době nepovolen.

5.17.2 api accesspoint blocking status

Funkce **api/accesspoint/blocking/status** vrací stav blokování přístupu pro jednotlivé přístupové body.

Skupiny služeb a privilegií
- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu – sledování.

Metody
- GET
- POST

Požadavek
Požadavek obsahuje parametry ve formátu URL.

Tabulka 1. Parametry požadavku URL

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
id	Ne	Celé číslo (0, 1)	Všechny	Určuje, pro který přístupový bod se má vrátit stav blokování přístupu. Jestliže je tento parametr vynechán, stav blokování přístupu se vrátí pro všechny přístupové body.

Příklad požadavku

```
URL: https://192.168.1.1/api/accesspoint/blocking/status?id=0
```

Odpověď

Odpověď je ve formátu **application/json**. Hodnota klíče `result` obsahuje jeden klíč, `accessPoints`, který obsahuje pole s objektem pro každý přístupový bod (pole má délku 1, jestliže byl v požadavku přístupový bod specifikován). Objekty v poli obsahují následující klíče.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
id	Celé číslo (0, 1)	Identifikuje přístupový bod (0 pro příchod, 1 pro odchod).
blocked	Boolean (`true`, `false`)	Obsahuje aktuální stav blokování přístupového bodu (`true`, když je přístupový bod blokován, `false`, když není blokován).

Příklad odpovědi

```
{ "success": true, "result": { "accessPoints": [ { "id": 0, "blocked": true }, { "id":
 1, "blocked": false } ] } }
```

Může dojít k různým chybám (např. nedostatečná privilegia).

5.17.3 api accesspoint grantaccess

Funkce **api/accesspoint/grantaccess** slouží k povolení vzdáleného přístupu příslušnému uživateli (konkrétního zaměstnance/uživatele nebo uživatele zahrnutého do obecné skupiny, např. návštěvník). Povolení vzdáleného přístupu lze udělit také pod přihlášeným účtem, kdy je ze záznamu zřejmé, který uživatel přístup umožnil.

Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu – řízení.

Metody

- GET
- POST

Požadavek

Požadavek obsahuje parametry ve formátu URL.

Tabulka 1. Parametry URL požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
id	Ano	Celé číslo (0, 1)	–	Určuje identifikátor přístupového bodu, který se má kontrolovat (0 pro Příchod a 1 pro Odchod).
user	Ano	uuid	–	Identifikace uživatele, jehož jménem má být otevřeno (a jehož přístupová nastavení mají být vzata v potaz).

Odpověď

Odpověď je ve formátu **application/json**.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
success	`true`, `false`	Hodnota je `true`, když je požadavek úspěšně zpracován (tj. blokování přístupu je v požadovaném stavu bez ohledu na nastalou změnu).

Klíč	Typické vrácené hodnoty	Popis
reason	`invalidAp`, `invalidCredential`, `accessBlocked`	Klíč se zobrazuje v případě odpovědi accessGranted:false. V případě úspěšného přístupu není tento klíč zobrazen.

Příklad odpovědi

```
{
  "success" : true,
  "result" : {
    "accessGranted" : false,
    "reason" : "invalidAp"
  }
}
```

Může dojít k různým chybám (např. chybějící povinný parametr). Chybový kód 18 (přístupový bod nepovolen) znamená, že požadavek nebyl zpracován, protože daný přístupový bod nebyl v té době povolen.

5.18 api lift

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/lift.**

5.18.1 api lift grantaccess

Funkce api/lift/grantaccess slouží k aktivaci pater výtahu na základě autorizace v jiném zařízení.

Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu – řízení.

Metody

- GET
- POST

Požadavek

Požadavek obsahuje parametry ve formátu URL (nebo ve formátu application/x-www-form-urlencoded při použití metody POST).

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnoty	Popis
uuid	Ano	uuid	-	Uuid uživatele, kterému má být udělen přístup do jeho pater (podle nastavení Přístupových práv).
duration	NE	1 .. 600	doba trvání nastavená v cílovém zařízení	Definuje dobu aktivace pater. Je-li tento parametr vynechán, použije se výchozí hodnota uvedená v parametru ***Switch-On Duration***.

Příklad požadavku

```
URL:
https://192.168.1.1/api/lift/grantaccess?user=09ebfd7d-24e4-4d58-
ad02-804ad69938a6&duration=180
```

Odpověď

Odpověď je ve formátu **application/json**.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
success	`true, false`	Když je požadavek úspěšně zpracován, je hodnota `true`. Dojde-li k chybě, je hodnota `false` a v klíči `error` jsou další informace.

Příklad odpovědi

```
{
    "success": true
}
```

Může dojít k různým chybám (např. chybějící povinný parametr). Vrátí-li se chybový kód 12, param: user (Uživatel nenalezen), požadavek nebyl zpracován, protože v adresáři cílového zařízení nebyl tento identifikátor uuid nalezen. Pokud zadaný uuid chybí nebo má nesprávný formát, zařízení odpoví chybovým kódem 11 (chybějící povinný parametr) nebo chybovým kódem 12 (neplatná hodnota parametru) v tomto pořadí.

Když dojde k aktivaci pater, která jsou již aktivní, bude použita delší doba trvání (zbývající čas versus nově požadovaná doba trvání).

Tento koncový bod API se dá použít k ovládání výtahových pater spolu se standardním řízením přístupu z jiného zařízení (např. posílat aktivační požadavek pro výtahová patra z interkomu na přístupovou jednotku, která je ve výtahu a komunikuje přímo s reléovými deskami).

5.19 api automation

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/automation**.

- 5.19.1 api automation trigger

5.19.1 api automation trigger

Funkce **/api/automation/trigger** slouží pro aktivaci funkce automatizace události HttpTrigger.

Funkce je součástí služby **Automation API** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Přístup k automatizaci**.

Pro tuto funkci lze použít metody **GET**.

Funkce je nastavena dle triggerId s parametry požadavku.

Odpověď je ve formátu **application/json** a obsahuje souhrn informací o zařízení:

Příklad:

```
{
  "success" : true
}
```

5.20 api cert

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/cert**.

5.20.1 api cert ca

Funkce **/api/phone/config** slouží ke správě CA certifikátů.

Funkce je součástí služby **Systém API** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém – řízení.**

U této funkce lze použít metodu **GET, PUT** nebo **DELETE**. Metoda **PUT** vrací informace o certifikátech. v zařízení. Metoda **PUT** slouží k nahrání certifikátů do zařízení. Metoda **DELETE** slouží ke smazání certifikátu ze zařízení.

Metoda GET

Parametry požadavku **GET**:

Parametr	Popis
id	Volitelný parametr (string) identifikující CA certifikát. Hodnota parametru **id** je přednastavené id nebo otisk (hash) certifikátu. Pokud není **id** zadáno, odpověď obsahuje úplný seznam všech CA certifikátů v zařízení.

Odpověď je ve formátu **application/json** a může obsahovat následující parametry:

Parametr	Popis
fingerprint	Otisk (hash) certifikátu
subject,issuer	Informace o subjektu nebo vydavateli: Common Name (CN), Organization (O), Organization Unit (OU), Location (L), State (S), Country (C)
id	Hodnota (*řetězec*) dříve nastavené identifikace certifikátu.
startDate	Datum začátku platnosti certifikátu

Parametr	Popis
endDate	Datum konce platnosti certifikátu
protected	Hodnota určující, že certifikát nemůže být smazán (`true`/`false`). Interní certifikáty s **id** začínajícím znakem # nemohou být smazány.
systemUseOnly	Hodnota určující, zda může daný certifikát uživatel zvolit jako certifikát pro jakoukoli službu. Pokud je hodnota `true`, certifikát se v seznamu pro výběr nezobrazí.

Příklad 1: Získání dat všech certifikátů

```
GET /api/cert/ca                          //požadavek
{                                         //odpověď
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint" : "4deea7060d80babf1643b4e0f0104c82995075b7",
        "subject" : {
          "CN" : "Thawte RSA CA 2018",
          "O" : "DigiCert Inc",
          "OU" : "www.digicert.com",
          "C" : "US"
        },
        "issuer" : {
          "CN" : "DigiCert Global Root CA",
          "O" : "DigiCert Inc",
          "OU" : "www.digicert.com",
          "C" : "US"
        },
        "startDate" : "2017-11-06T12:23:52Z",
        "endDate" : "2027-11-06T12:23:52Z",
        "allowRemove" : true
      },
      {
        "fingerprint" : "a8985d3a65e5e5c4b2d7d66d40c6dd2fb19c5436",
        "subject" : {
          "CN" : "DigiCert Global Root CA",
          "O" : "DigiCert Inc",
```

```
          "OU" : "www.digicert.com",
          "C" : "US"
        },
        "issuer" : {
          "CN" : "DigiCert Global Root CA",
          "O" : "DigiCert Inc",
          "OU" : "www.digicert.com",
          "C" : "US"
        },
        "startDate" : "2006-11-10T00:00:00Z",
        "endDate" : "2031-11-10T00:00:00Z",
        "protected" : false,
        "id" : "#my2n-utility",
        "systemUseOnly" : true
      }
    ]
  }
}
```

Příklad 2: Získání informací pro certifikát zadaného **id**

```
GET /api/cert/ca?id=#my2n-utility                    //požadavek
{                                                    //odpověď
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint" : "a8985d3a65e5e5c4b2d7d66d40c6dd2fb19c5436",
        ...
        "id" : "#my2n-utility",
        ...
      }
    ]
  }
}
```

Metoda PUT

Při opakovaném nahrání stejného certifikátu se původní certifikát přepíše. Je možné nahrát více certifikátů v jednom souboru ve formátu PEM. Soubor může obsahovat libovolné bloky, zpracovávají se pouze certifikáty. Pokud se některý z obsažených certifikátů nepodaří nahrát, žádný se neuloží a vrátí se kód chyby.

Parametry požadavku **PUT**:

Parametr	Popis
blob-cert	Povinný parametr obsahující soubor s certifikátem ve formátu DER nebo PEM.

Parametr	Popis
id	Volitelný parametr (string) identifikující CA certifikát. Pokud je nahrán certifikát pod stejným **id**, původní certifikát s tímto **id** se přepíše. Při nahrávání souboru s více certifikáty se **id** nezadává.

Odpověď je ve formátu **application/json** a obsahuje otisk nahraného certifikátu

Parametr	Popis
fingerprint	Otisk (hash) nahraného certifikátu
replaced	Otisk přepsaného certifikátu

Příklad:

```
PUT /api/cert/ca                        //požadavek
{                                       //odpověď
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint": "9623fa35e414aa930ed22348a22d04a4c4fda26b"
      },
      {
        "fingerprint": "9623fa35e414aa930ed22348a22d04a4c4fda26b"
        "replaced": "9623fa26e414aa930ed22348a22d04a4c4fda26c"
      }
    ]
  }
}
----
{                                       //odpověď
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "blob-cert",
    "description" : "invalid certificate",
    "data" : "invalid_cert"
  }
}
```

Metoda DELETE

Parametry požadavku **DELETE**:

Parametr	Popis
id	Povinný parametr (string) identifikující CA certifikát. Parametr **id** je přednastavené id nebo otisk (hash) certifikátu.

Odpověď je ve formátu **application/json**.

Příklad:

```
DELETE /api/cert/ca?
fingerprint=4deea7060d80babf1643b4e0f0104c82995075b8          //požadavek
{
//odpověď
  "success" : true
}
---
{
//odpověď
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "id",
    "description" : "certificate not found",
    "data": "cert_not_found"
  }
}
```

5.20.2 api cert user

Funkce **/api/cert/user** slouží ke správě uživatelských certifikátů.

Funkce je součástí služby **Systém API** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém – řízení.**

U této funkce lze použít metodu **GET, PUT** nebo **DELETE**. Metoda **PUT** vrací informace o certifikátech. v zařízení. Metoda **PUT** slouží k nahrání certifikátů do zařízení.
Metoda **DELETE** slouží ke smazání certifikátu ze zařízení.

Metoda GET

Parametry požadavku **GET**:

Parametr	Popis
id	Volitelný parametr (string) identifikující uživatelský certifikát. Hodnota parametru **id** je přednastavené id nebo otisk (hash) certifikátu. Pokud není **id** zadáno, odpověď obsahuje úplný seznam všech uživatelských certifikátů v zařízení.

Odpověď je ve formátu **application/json** a může obsahovat následující parametry:

Parametr	Popis
fingerprint	Otisk (hash) certifikátu
subjet, issuer	Informace o subjektu nebo vydavateli: Common Name (CN), Organization (O), Organization Unit (OU), Location (L), State (S), Country (C).
id	Hodnota (*řetězec*) dříve nastavené identifikace certifikátu.
startDate	Datum začátku platnosti certifikátu
endDate	Datum konce platnosti certifikátu
protected	Hodnota určující, že certifikát nemůže být smazán (`true`/`false`). Interní certifikáty s **id** začínajícím znakem # nemohou být smazány.
systemUseOnly	Hodnota určující, zda může daný certifikát uživatel zvolit jako certifikát pro jakoukoli službu. Pokud je hodnota `true`, certifikát se v seznamu pro výběr nezobrazí.

Příklad 2: Získání informací pro určitý certifikát identifikovaný otiskem

```
GET /api/cert/user?id=a164b11215a30f08603fd85c314327e274772b00   //požadavek
{
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint" : "a164b11215a30f08603fd85c314327e274772b00",
        "subject" : {
```

```
            "CN" : "00-0001-0205",
            "O" : "2N TELEKOMUNIKACE a.s.",
            "S" : "Czech Republic",
            "C" : "CZ"
          },
          "issuer" : {
            "CN" : "My2N Device Utility Certificate Authority",
            "O" : "2N TELEKOMUNIKACE a.s.",
            "S" : "Czech Republic",
            "C" : "CZ"
          },
          "startDate" : "2021-11-08T07:50:36Z",
          "endDate" : "2022-02-06T07:50:36Z",
          "protected" : false,
          "id" : "#my2n-utility",
          "systemUseOnly" : true
        }
      ]
    }
}
```


Metoda PUT

Při opakovaném nahrání stejného certifikátu se původní certifikát přepíše.

Parametry požadavku **PUT**:

Parametr	Popis
blob-cert	Povinný parametr obsahující soubor s certifikátem ve formátu DER nebo PEM.
blob-pk	Povinný parametr soukromého klíče ve formátu DER nebo PEM
password	Nepovinný parametr obsahující heslo k nahraného klíče.
id	Volitelný parametr (string) identifikující uživatelský certifikát. Pokud je nahrán certifikát pod stejným **id**, původní certifikát s tímto **id** se přepíše.

Odpověď je ve formátu **application/json** a obsahuje otisk nahraného certifikátu

Parametr	Popis
fingerprint	Otisk (hash) nahraného certifikátu

Parametr	Popis
replaced	Otisk přepsaného certifikátu

Příklad

```
PUT /api/cert/ca         //požadavek
{
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint": "9623fa25e414aa930ed22348a22d04a4c4fda26b"
      },
      {
        "fingerprint": "9623fa25e414aa930ed22348a22d04a4c4fda26b"
        "replaced": "9623fa25e414aa930ed22348a22d04a4c4fda26c"
      }
    ]
  }
}
Response
{
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "blob-cert",
    "description" : "invalid certificate",
    "data" : "invalid_cert"
  }
}
```

Metoda DELETE

Parametry požadavku:

Parametr	Popis
id	Povinný parametr (string) identifikující uživatelský certifikát. Parametr **id** je přednastavené id nebo otisk (hash) certifikátu.

Odpověď je ve formátu **application/json**.

Příklad

```
DELETE /api/cert/ca?fingerprint=4deea7060d80babf1643b4e0f0104c82995075b7
{
  "success": true
}
```