Lua-Scripts - Templates für Drittsysteme

Unter Einstellungen > Schnittstellen > Datenübergabe > Lua Scripts können Sie beliebige Systeme anbinden, sofern diese über eine öffentlich erreichbare Schnittstelle verfügen.

Alamos FE2

Datenübergabe an den FE2, um die Zusatzalarmierung, Gebäudesteuerung und Einsatztablets zu versorgen

Eingabeparameter

```
[
  {
    "type": "password",
    "key": "apiKey",
    "label": "API Key *"
  },
  {
    "type": "text",
    "key": "fe2Id",
    "label": "FE2-Identifikation *"
  },
  {
    "type": "text",
    "key": "customerId",
    "label": "Kunden-Identifikation"
  },
  {
    "type": "password",
    "key": "password",
    "label": "Verschlüsselungspasswort * (min. 16 Zeichen)"
  },
  {
    "type": "text",
    "key": "partnerName",
    "label": "Partner Name"
  }
]
```

Script

```
local fetch = require "fetch"
local json = require "json"
local encoding = require "encoding"
local crypto = require "crypto"

local function isEmpty(value)
    return value == "" or value == nil or value == json.null
end

assert(event.model == "alarm", "Nur Alarm Events werden unterstützt")
assert(not isEmpty(configuration.fe2Id), "FE2-Identifikation muss gesetzt sein")
assert(not isEmpty(configuration.apiKey), "API Key muss gesetzt sein")
assert(not isEmpty(configuration.customerId), "Kunden-Identifikation")
assert(not isEmpty(configuration.password), "Verschlüsselungspasswort muss gesetzt sein")
assert(typeof(configuration.password) == "string" and configuration.password:len() >= 16,
"Verschlüsselungspasswort muss min. 16 Zeichen lang sein")
configuration.partnerName = configuration.partnerName or "DIVERA 24/7"


local alarm = event.payload
```

```
local externalId = `{alarm.id}`
if not isEmpty(alarm.foreign_id) then
    externalId = alarm.foreign_id
end

local payload = {
    type = "ALARM",
    timestamp = alarm.ts_update,
    sender = configuration.sender or `DIVERA Einheit {alarm.cluster_id}`,
    authorization = configuration.authorization,
    data = {
        externalId = externalId,
        message = { alarm.text },
        keyword = alarm.title,
        units = { },
        vehicles = { }
    }
}


payload.data.location = {}

if not isEmpty(alarm.lat) and not isEmpty(alarm.lng) then
    payload.data.location.coordinate = { alarm.lng, alarm.lat }
end

if not isEmpty(alarm.address) then
    payload.data.custom = { location_dest = alarm.address }
end

-- Gruppen zu units hinzufügen
if next(event.payload.groups) ~= "nil" then
    for key,value in event.payload.groups do
        table.insert(payload.data.units, {address = event.payload.groups[key].title})
    end
end

-- Fahrzeuge zu units hinzufügen
if next(event.payload.vehicles) ~= "nil" then
    for key,value in event.payload.vehicles do
        table.insert(payload.data.vehicles, {name = event.payload.vehicles[key].name})
    end
end

local password = crypto.subtle.importKey("raw", configuration.password, "PBKDF2", false)
local salt = crypto.getRandomValues(buffer.create(64))
local key = crypto.subtle.deriveKey({
    name = "PBKDF2",
    hash = "SHA-1",
    salt = salt,
    iterations = 5000
}, password, {
    name = "AES-CBC",
    length = 256
}, false)

local iv = crypto.getRandomValues(buffer.create(16))
local encryptedMessage = crypto.subtle.encrypt(
    {
        name = "AES-CBC",
        iv = iv
    },
    key,
    json.encode(payload)
)

local cloudPayload = {
    iv = encoding.encode("BASE64", iv),
    salt = encoding.encode("BASE64", salt),
    encryptedMessage = encoding.encode("BASE64", encryptedMessage),
    nameOfPartner = configuration.partnerName,
```

```
    apiKey = configuration.apiKey,
    topic = `{configuration.fe2Id}{configuration.customerId}`,
    messageIdentifier = crypto.randomUUID()
}

print(json.encode(cloudPayload))

local url =  "https://tpac.alamos-gmbh.bayern/thirdparty/alarm"
local response = fetch.post(url, {
    contentType = "application/json",
    data = json.encode(cloudPayload)
})

print(response.body)
```

VOMATEC FLOW

Datenübergabe an das Verwaltungssystem FLOW

Eingabeparameter

```
[
  {
    "type": "text",
    "key": "clientId",
    "label": "Client ID",
    "description": "Die Client-ID aus der Anleitung"
  },
  {
    "type": "text",
    "key": "clientSecret",
    "label": "Client Secret",
    "description": "Das Client-Secret aus der Anleitung"
  }
]
```

Script

Script

```
local fetch = require "fetch"
local json = require "json"

local function isEmpty(string)
  return string == nil or string == '' or string == json.null
end

local response = fetch({
  url = "https://id.vomatec.app/connect/token",
  method = "POST",
  headers = {
    ["Content-Type"] = "application/x-www-form-urlencoded;charset=UTF-8"
  },
  body = `client_id={configuration.clientId}&client_secret={configuration.clientSecret}
&grant_type=client_credentials`
})

print("Alarm: " .. json.encode(event.payload))

local decoded = json.decode(response.body)
local token = decoded.access_token
```

```
local einsatzmittelArray = {}

if event.payload.vehicles then
    for _, vehicle in ipairs(event.payload.vehicles) do
        local newVehicle = {
          fahrzeugBezeichnung = tostring(vehicle.name)
        }

        if not isEmpty(vehicle.foreignid) then
            newVehicle.fremdsystemId = tostring(vehicle.foreignid)
          else
            newVehicle.fremdsystemId = tostring(vehicle.name)
        end

        table.insert(einsatzmittelArray, newVehicle)
    end
end

local ts = event.payload.ts_create
local beginnDatum = if not isEmpty(ts) then string.sub(ts, 1, 10) else ""
local beginnUhrzeit = if not isEmpty(ts) then string.sub(ts, 12, 19) else ""

local ermitteltеEinsatznummer = if not isEmpty(event.payload.foreign_id) then event.payload.foreign_id else
event.payload.id

local payloadEinsatzdaten = {
  einsatznummer = ermitteltеEinsatznummer,
  stichwort = event.payload.title,
  meldebild = event.payload.text,
  einsatzleiter = "DIVERA247",
  istBeendet = if (event.payload.closed == "1") then true else false,
  istAktiv = if (event.payload.closed == "1") then false else true,
  beginnDatum = beginnDatum,
  beginnUhrzeit = beginnUhrzeit,

  einsatzort = {
    adresszusatz = event.payload.address,
    latitude = 0,
    longitude = 0
  },

  kurzbericht = event.payload.report,
  einsatzmittel = einsatzmittelArray
}

if not isEmpty(event.payload.lat) and not isEmpty(event.payload.lng) then
    payloadEinsatzdaten.einsatzort.latitude = tonumber(event.payload.lat) or 0
    payloadEinsatzdaten.einsatzort.longitude = tonumber(event.payload.lng) or 0
end

print("payloadEinsatzdaten: " .. json.encode(payloadEinsatzdaten))


local apiUrl = `https://api.flow.arigon.vomatec.app/ext/einsaetze/{ermitteltеEinsatznummer}?api-version=1.0-
extern`

local resultEinsatzdaten = fetch({
  url = apiUrl,
  method = "PUT",
  headers = {
    ["Content-Type"] = "application/json",
    ["Authorization"] = `Bearer {token}`
  },
  body = json.encode(payloadEinsatzdaten)
})
```

Firemanager

Datenübergabe an das Verwaltungssystem Firemanager

Eingabeparameter

```
[
  {
    "type": "password",
    "key": "bearer",
    "label": "Authentifizierung"
  }
]
```

Script

Script

```
local fetch = require "fetch"
local json = require "json"

local function isEmpty(value)
    return value == "" or value == nil or value == json.null
end

-- Einsatzdaten übergeben
local payload = {
    stempel = "Bereitgestellt durch Divera 24/7",
    einsatz = {
        externeid = event.payload.id,
        einsatznummer = if (isEmpty(event.payload.foreign_id)) then event.payload.id else event.payload.
foreign_id,
        stichwort = event.payload.title,
        bemerkung = event.payload.text,
        schlusswort = event.payload.report,
        sondersignal = event.payload.priority,
        meldernummer = event.payload.caller,
        einsatzort = {
            einsatzort_raw = event.payload.address,
            lat = event.payload.lat,
            lon = event.payload.lng,
            objekt = event.payload.scene_objekt
        }
    },
    fahrzeuge = {}
}

-- Zugeteilte Fahrzeuge hinzufügen
if next(event.payload.vehicles) ~= "nil" then
    for key,value in event.payload.vehicles do
        table.insert(payload.fahrzeuge, {funkrufname = event.payload.vehicles[key].name})
    end
end

local url =  "https://beta.firemanager.de/portal/eldiseinsatzes/divera365schnittstelle"

local response = fetch.post(url, {
    contentType = "application/json",
    data = json.encode(payload)
}, {
    headers = {
        Authorization = `Bearer {configuration.bearer}`
    }
})

print(response.body)
```

Zum Zeitpunkt der Veröffentlichung der einzelnen Scripts waren diese funktionstüchtig und mit dem jeweiligen Drittsystem kompatibel.

Wir geben jedoch keine Gewährleistung auf die uneingeschränkte und dauerhafte Funktionalität der vorgestellten Lua Scripts.

Verwandte Artikel

- Termine in einen externen Kalender übertragen
- iOS - Empfangen von Push Mitteilungen auf der Apple Watch aktivieren/ deaktivieren
- Android-App manuell installieren
- Newsletter 2026

- Monitor-App – Vollbildmodus beenden / Menüleiste einblenden