International Transport Forum

International Transport Forum (ITF)

XML API Integration Guide - Haulier

Transport-company fleet systems

Document Version: D_ECMT_TLS_MXML-001 ● June 2, 2026

alfasoft
agile software company

ECMT MULTILATERAL QUOTA ELECTRONIC DATA

1. Overview

This guide is for haulier (transport company) systems integrating with the ECMT Transport Licences XML API. As a haulier you register your trucks and trailers, manage logbook trips for your licences, export your own licence/logbook data, download PDFs. You authenticate with the private key issued to your company.

All endpoints share the same transport, authentication and response conventions described in sections 2-4. Section 5 documents each endpoint available to the Haulier role.

Service base URL

Test: https://edi-test.itf-oecd.org/api/license/
Production: https://eds.itf-oecd.org/api/license/
These URLs have no visual presentation and cannot be opened in a browser - they are technical endpoints for system-to-system integration.

2. Authentication & signing

Every request body is an XML document digitally signed with the private key issued to your account by the ITF Secretariat. The public key used for verification is stored server-side, so the KeyInfo element may be omitted from the signature.

Signing parameters

Field	Type	Description
`CanonicalizationMethod`	C14N	Exclusive XML canonicalization (http://www.w3.org/2001/10/xml-exc-c14n#).
`SignatureMethod`	RSA-SHA512	http://www.w3.org/2001/04/xmldsig-more#rsa-sha512
`Reference Transform`	Enveloped	http://www.w3.org/2000/09/xmldsig#enveloped-signature, then exclusive C14N.
`DigestMethod`	SHA-256	http://www.w3.org/2001/04/xmlenc#sha256

The <User> element (your account email) MUST be the first child of the root element - the validator reads the account from DocumentElement.FirstChild. No extra whitespace or namespaces should be added to the signed XML.

A haulier signs with the private key issued to its own company. A full signed example and reference C# / PHP signing code are in Appendix A.

3. Transport

Method: HTTP POST

Content-Type: text/xml

Body: the signed XML document

Success HTTP status: 200 (PDF endpoints also use 200 for the binary file; 400 for errors)

4. Response format & status codes

Write endpoints (imports, logbook operations, add-control) return a <Result> envelope: a numeric <Status> (1 = success, 0 = error) and a <Message> that carries text only on error - it is empty on success. A successful write looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<Result>
  <Status>1</Status>
  <Message></Message>
</Result>
```

Status	Meaning
1	Success. The operation was applied (writes) or every row was imported. The <Message> is empty on success - it is populated only on error. Read endpoints return a data payload (<Licences> / <Logbooks> / <Controls>) instead of a <Result>.
0	Error - nothing saved. Covers invalid XML, authentication / signature failures ("Unauthorized access", "Incorrect DigestValue"), and imports where one or more rows failed. The <Message> carries the detail (for imports the "Imported: N, Errors: M." counters and per-row messages).

Read endpoints (the Export* endpoints and GetLogbookByUnicDocNumber) return a data payload (<Licences>, <Logbooks> or <Controls>) on success, or a <Result> with Status 0 on failure. PDF endpoints return the binary PDF (application/pdf, HTTP 200) on success, or a <Result> error with HTTP 400.

5. Endpoints

5.1 Register trucks

POST `/api/license/ImportTrucks`

Register the haulier's fleet trucks. Each <Truck> is validated (17-char VIN, valid Euro category, country code, ownership type) and checked for duplicates within the file and against the system.

Root element: `<ImportTrucksRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
Year	Integer	Registration year, e.g. 2026.
VinCode	String(17)	Vehicle identification number, exactly 17 characters.
PlateNumber	String	Registration plate.
OwnershipType	Integer	Ownership type id (1 = owned, etc.).
EUROCategory	Numeric	Truck Euro category: 5 = Euro-5, 6 = Euro-6.
CountryOfRegistration	String	3-letter ISO country code, e.g. SRB.
IsActive	Boolean	Whether the truck is active (true/1 = active). Defaults to active if omitted.
InactiveMessage	String(200)	Reason for inactivation (optional).

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<ImportTrucksRequest>
  <User>100000001@hau.com</User>
  <Year>2026</Year>
  <Trucks>
    <Truck>
      <VinCode>WMA06XZZXEM646755</VinCode>
      <PlateNumber>NS208MV</PlateNumber>
      <OwnershipType>1</OwnershipType>
      <EUROCategory>6</EUROCategory>
      <CountryOfRegistration>SRB</CountryOfRegistration>
      <IsActive>true</IsActive>
    </Truck>
  </Trucks>
</ImportTrucksRequest>
```

Returns a <Result>. Counters and per-row errors as in section 4.

Example response

```
<?xml version="1.0" encoding="UTF-8"?>
<Result>
  <Status>1</Status>
  <Message></Message>
</Result>
```

5.2 Register trailers

POST `/api/license/ImportTrailers`

Register the haulier's fleet trailers. Same validation as trucks but without the EUROCategory field (trailers have no engine).

Root element: `<ImportTrailersRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
Year	Integer	Registration year.
VinCode	String(17)	Vehicle identification number, 17 characters.
PlateNumber	String	Registration plate.
OwnershipType	Integer	Ownership type id.
CountryOfRegistration	String	3-letter ISO country code.
IsActive	Boolean	Whether the trailer is active (true/1 = active). Defaults to active if omitted.
InactiveMessage	String(200)	Reason for inactivation (optional).

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<ImportTrailersRequest>
  <User>100000001@hau.com</User>
  <Year>2026</Year>
  <Trailers>
    <Trailer>
      <VinCode>WKESD000000833726</VinCode>
      <PlateNumber>KO1528AD</PlateNumber>
      <OwnershipType>3</OwnershipType>
      <CountryOfRegistration>NMK</CountryOfRegistration>
      <IsActive>true</IsActive>
    </Trailer>
  </Trailers>
</ImportTrailersRequest>
```

Returns a <Result> with import counters.

5.3 Export own licences

POST `/api/license/ExportLicences`

Retrieve the haulier's own licences for the given year.

Root element: `<ExportLicenceRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
Year	Integer	4-digit year, e.g. 2026.

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<ExportLicenceRequest>
```

```
  <User>100000001@hau.com</User>
  <Year>2026</Year>
</ExportLicenceRequest>
```

Returns a <Licences> payload scoped to the haulier.

5.4 Export own logbooks

POST `/api/license/ExportLogbooks`

Retrieve the haulier's logbooks (trips, steps, transit points) for the year.

Root element: `<ExportLogbooksRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
Year	Integer	4-digit year, e.g. 2026.

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<ExportLogbooksRequest>
  <User>100000001@hau.com</User>
  <Year>2026</Year>
</ExportLogbooksRequest>
```

Returns a <Logbooks> payload (see the NIA guide section for the full element structure).

5.5 Export logbook by licence number

POST `/api/license/ExportLogbookByNumber`

Retrieve one of the haulier's logbooks by licence number.

Root element: `<ExportLogbookByNumberRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<ExportLogbookByNumberRequest>
  <User>100000001@hau.com</User>
```

```
  <LicenceNumber>MD00026</LicenceNumber>
</ExportLogbookByNumberRequest>
```

Returns a <Logbooks> payload limited to the requested licence.

5.6 Create a logbook trip

POST `/api/license/CreateLogbookTrip`

Start a new trip on a licence's logbook (the trip number is assigned automatically).

Root element: `<CreateLogbookTripRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).
DepartureDate	Date	YYYY-MM-DD.
PlaceOfLoading	String(50)	Place of loading.
CountryOfLoading	String(2)	Two-letter country code of loading.
DepartureWeight	Decimal(2,3)	Gross weight at loading (tonnes, 3 decimals).
DepartureOdometer	Numeric	Odometer at departure.
PlaceOfUnloading	String(50)	Planned unloading location.
CountryOfUnloading	String(2)	Two-letter country code of unloading.
ArrivalWeight	Decimal(2,3)	Gross weight at unloading (tonnes, 3 decimals).
IsAssambledGoods	Boolean	1/true = assembled goods, 0 = not (doc spelling).
DepartureTruckPlateNumber	String(20)	Plate of a truck in the haulier's fleet.
DepartureTrailerPlateNumber	String(20)	Plate of a trailer in the haulier's fleet (optional).

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<CreateLogbookTripRequest>
  <User>100000001@hau.com</User>
  <LicenceNumber>MD00026</LicenceNumber>
  <DepartureDate>2026-02-01</DepartureDate>
  <PlaceOfLoading>ISTANBUL</PlaceOfLoading>
  <CountryOfLoading>TR</CountryOfLoading>
  <DepartureWeight>20.000</DepartureWeight>
  <DepartureOdometer>100000</DepartureOdometer>
  <PlaceOfUnloading>WARSAW</PlaceOfUnloading>
  <CountryOfUnloading>PL</CountryOfUnloading>
  <ArrivalWeight>20.000</ArrivalWeight>
  <IsAssambledGoods>0</IsAssambledGoods>
  <DepartureTruckPlateNumber>NS208MV</DepartureTruckPlateNumber>
  <DepartureTrailerPlateNumber>KO1528AD</DepartureTrailerPlateNumber>
```

```
</CreateLogbookTripRequest>
```

Returns a <Result> (Status 1 on success).

5.7 Add a loading/unloading step

POST `/api/license/AddStepToLogbookTrip`

Add a loading or unloading step (weight change) to the current trip of a licence.

Root element: `<AddStepToLogbookTripRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).
Date	Date	YYYY-MM-DD of the step.
Place	String	Step location.
WeightSign	String	'+' for loading, '-' for unloading.
Weight	Decimal	Weight delta (tonnes).
Odometer	Integer	Odometer at the step.
TruckPlateNumber	String	Optional truck plate.
TrailerPlateNumber	String	Optional trailer plate.

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<AddStepToLogbookTripRequest>
  <User>100000001@hau.com</User>
  <LicenceNumber>MD00026</LicenceNumber>
  <Date>2026-02-03</Date>
  <Place>SOFIA</Place>
  <WeightSign>-</WeightSign>
  <Weight>5.000</Weight>
  <Odometer>100850</Odometer>
</AddStepToLogbookTripRequest>
```

Returns a <Result>.

5.8 Change a trip

POST `/api/license/ChangeLogbookTrip`

Update the current trip's attributes. Accepts the same fields as CreateLogbookTrip (section 5.6).

Root element: `<ChangeLogbookTripRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).
Date	Date	YYYY-MM-DD of the step.
Place	String	Step location.
WeightSign	String	'+' for loading, '-' for unloading.
Weight	Decimal	Weight delta (tonnes).
Odometer	Integer	Odometer at the step.
TruckPlateNumber	String	Optional truck plate.
TrailerPlateNumber	String	Optional trailer plate.

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<ChangeLogbookTripRequest>
  <User>100000001@hau.com</User>
  <LicenceNumber>MD00026</LicenceNumber>
  <DepartureDate>2026-02-01</DepartureDate>
  <PlaceOfLoading>ISTANBUL</PlaceOfLoading>
  <CountryOfLoading>TR</CountryOfLoading>
  <DepartureWeight>21.000</DepartureWeight>
  <DepartureOdometer>100000</DepartureOdometer>
  <PlaceOfUnloading>WARSAW</PlaceOfUnloading>
  <CountryOfUnloading>PL</CountryOfUnloading>
  <ArrivalWeight>21.000</ArrivalWeight>
  <IsAssambledGoods>0</IsAssambledGoods>
</ChangeLogbookTripRequest>
```

Returns a <Result>.

5.9 Cancel a trip

POST `/api/license/CancelLogbookTrip`

Cancel the current trip of a licence.

Root element: `<CancelLogbookTripRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).
Reason	String	Cancellation reason.

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<CancelLogbookTripRequest>
  <User>100000001@hau.com</User>
  <LicenceNumber>MD00026</LicenceNumber>
  <Reason>Cargo cancelled by client</Reason>
</CancelLogbookTripRequest>
```

Returns a <Result>.

5.10 Add a transit point

POST `/api/license/AddTransitpointLogbookTrip`

Record a transit (border crossing) point on the current trip.

Root element: `<AddTransitpointLogbookTripRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).
Date	Date	YYYY-MM-DD of the crossing.
Place	String	Transit/border location.
CountryOfTransit	String	3-letter country code crossed.
Odometer	Integer	Odometer at the transit point.
TruckPlateNumber	String	Optional truck plate.

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<AddTransitpointLogbookTripRequest>
  <User>100000001@hau.com</User>
  <LicenceNumber>MD00026</LicenceNumber>
  <Date>2026-02-02</Date>
  <Place>KAPIKULE BORDER</Place>
  <CountryOfTransit>BGR</CountryOfTransit>
  <Odometer>100400</Odometer>
</AddTransitpointLogbookTripRequest>
```

Returns a <Result>.

5.11 Finish a trip

POST `/api/license/FinishLogbookTrip`

Close the current trip with arrival details.

Root element: `<FinishLogbookTripRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).
ArrivalDate	Date	YYYY-MM-DD of arrival.
PlaceOfUnloading	String	Actual unloading location.
CountryOfUnloading	String	3-letter country code of unloading.
ArrivalOdometer	Integer	Odometer at arrival.

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<FinishLogbookTripRequest>
  <User>100000001@hau.com</User>
  <LicenceNumber>MD00026</LicenceNumber>
  <ArrivalDate>2026-02-05</ArrivalDate>
  <PlaceOfUnloading>WARSAW</PlaceOfUnloading>
  <CountryOfUnloading>POL</CountryOfUnloading>
  <ArrivalOdometer>101200</ArrivalOdometer>
</FinishLogbookTripRequest>
```

Returns a <Result>.

5.12 Licence PDF

POST `/api/license/GetLicencePDF`

Download the PDF of one of the haulier's licences.

Root element: `<LicencePDFRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<LicencePDFRequest>
  <User>100000001@hau.com</User>
  <LicenceNumber>MD00026</LicenceNumber>
```

```
</LicencePDFRequest>
```

On success: HTTP 200, application/pdf. On failure: HTTP 400 + <Result>.

5.13 Logbook PDF

POST `/api/license/GetLogbookPDF`

Download the PDF of one of the haulier's logbooks.

Root element: `<LogbookPDFRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
LicenceNumber	String	Full licence number incl. country code, e.g. MD00026 (or just the numeric part).

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<LogbookPDFRequest>
  <User>100000001@hau.com</User>
  <LicenceNumber>MD00026</LicenceNumber>
</LogbookPDFRequest>
```

On success: HTTP 200, application/pdf. On failure: HTTP 400 + <Result>.

5.14 Export own controls

POST `/api/license/ExportControlsByHaulier`

Retrieve roadside controls recorded against the haulier's licences. The caller must own the haulier identified by HaulierCode.

Root element: `<ExportControlsByHaulierRequest>`

Request fields

Field	Type	Description
User	String	Account email (haulier). MUST be the first child of the root element - the signature validator reads it from there.
HaulierCode	String	Haulier local registration code (LocalId), e.g. 100000001.

Request example (unsigned body - sign before posting, see Appendix A)

```
<?xml version="1.0" encoding="UTF-8"?>
<ExportControlsByHaulierRequest>
```

```
  <User>100000001@hau.com</User>
  <HaulierCode>100000001</HaulierCode>
</ExportControlsByHaulierRequest>
```

Returns a <Controls> payload (see the Control Authority guide for the <Control> element structure). A foreign or unknown HaulierCode yields an empty <Controls> payload - the endpoint only returns the caller's own team data.

Appendix A. Signing reference

A.1 Signed XML example

The body below is a ImportTrucksRequest document after signing - a representative request a haulier sends. Every endpoint follows the same pattern: the payload root (with <User> as its first child), then a <Signature> element appended as the last child of the root.

```
<?xml version="1.0" encoding="UTF-8"?>
<ImportTrucksRequest>
  <User>100000001@hau.com</User>
  <Year>2026</Year>
  <Trucks>...</Trucks>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    <SignedInfo>
      <CanonicalizationMethod
Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
      <SignatureMethod
Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha512"/>
      <Reference URI="">
        <Transforms>
          <Transform
Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"/>
          <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"/>
        </Transforms>
        <DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
        <DigestValue>JXv3Auvi...=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>g1yF+MDpigG7...==</SignatureValue>
  </Signature>
</ImportTrucksRequest>
```

A.2 Signing in C# (.NET)

```
  XmlDocument doc = new XmlDocument();
  doc.LoadXml(xml);
var rsa = new RSACryptoServiceProvider();
  var keyDoc = new XmlDocument();
  keyDoc.Load("private_key.xml");
rsa.FromXmlString(keyDoc.InnerXml);

SignedXml signedXml = new SignedXml(doc);
  signedXml.SignedInfo.CanonicalizationMethod = "http://www.w3.org/2001/10/xml-exc-
c14n#";
signedXml.SignedInfo.SignatureMethod = "http://www.w3.org/2001/04/xmldsig-more#rsa-
sha512";
signedXml.SigningKey = rsa;
```

```
Reference reference = new Reference { Uri = "" };
  reference.AddTransform(new XmlDsigEnvelopedSignatureTransform());
signedXml.AddReference(reference);
signedXml.ComputeSignature();
doc.DocumentElement.AppendChild(doc.ImportNode(signedXml.GetXml(), true));

string signed = "<?xml version=\"1.0\" encoding=\"UTF-8\"?>" +
doc.DocumentElement.OuterXml;
```

A.3 Posting in C#

```
  var client = new HttpClient();
  var content = new StringContent(signed, Encoding.UTF8, "text/xml");
  var result = await client.PostAsync(
      "https://edi-test.itf-oecd.org/api/license/ImportTrucks", content);
  string response = await result.Content.ReadAsStringAsync();
```

A.4 Signing in PHP (xmlseclibs)

```
$objDSig = new XMLSecurityDSig('');
$objDSig->setCanonicalMethod(XMLSecurityDSig::EXC_C14N);
$objDSig->addReference($doc, XMLSecurityDSig::SHA256,
    array('http://www.w3.org/2000/09/xmldsig#enveloped-signature'));
$objKey = new XMLSecurityKey(XMLSecurityKey::RSA_SHA512,
array('type'=>'private'));
$objKey->loadKey('private_key', TRUE);
$objDSig->sign($objKey);
$objDSig->appendSignature($doc->documentElement);
$doc->save('_signed.xml');
```