Kira Liquid Cooling Water™

Certificate of Authenticity

100ml · paid in monero

#007-A8B61A

OWNER DID
did:plc:vtdq3y23vehztw33eohawmpz

VOLUME
100 ml

EDITION
limited-2026-april

PURCHASED
2026-05-08T00:24:52.208764Z

MONERO AMOUNT
0.015626 XMR

TX HASH
bcf859b9d4924cfafbd4376d6087066a6f5f8850a866492955d53ff0cc500357

ISSUER DID
did:plc:2tqqxubv2lu4ahj35ysjer2r

ISSUER KEY (SIGNING)
did:key:zQ3shqi8dxM5qMPUFePDdjMcHxNEsmojC2QtftiyR5uXGDnMC

ECDSA-SECP256K1 SIGNATURE

YcEPUaqtRqmgHL6KqsBoTC0-AV2ePGJJQF7lD-3tA9hI1F0TTgtwB3mXePLyVS-1
0Dl5p82G30pffJaU0yz4yQ

Signature is over canonical-JSON of the certificate body excluding the signature field itself.
Verify against issuer key at https://plc.directory/did:plc:2tqqxubv2lu4ahj35ysjer2r
→ See page 2 for verification instructions and IRL redemption.

How to verify this certificate

ECDSA-secp256k1 signature over canonical JSON of the body fields.

WHAT'S SIGNED

The 'signature' field above is generated by:

1. Take all fields of this certificate EXCEPT 'signature'.
2. Encode as canonical JSON (sorted keys, no whitespace, UTF-8).
3. SHA-256 the bytes.
4. ECDSA sign with kira's secp256k1 private key.
5. Encode (r || s) as 64 raw bytes, base64url-no-pad.

The corresponding public key is the 'issuerKey' DID above.

PYTHON VERIFICATION SNIPPET

```
import json, base64
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.asymmetric import ec
from cryptography.hazmat.primitives.asymmetric.utils import encode_dss_signature

cert = json.load(open('kira-coolant-XXX-XXXXXX.json'))
sig = cert.pop('signature')
body = json.dumps(cert, sort_keys=True, separators=(',', ':'),
                  ensure_ascii=False).encode('utf-8')

# decode 64-byte sig to DER
raw = base64.urlsafe_b64decode(sig + '=' * (-len(sig) % 4))
r = int.from_bytes(raw[:32], 'big')
s = int.from_bytes(raw[32:], 'big')
der = encode_dss_signature(r, s)

# you need the issuer's public key (decode from did:key)
# the did:key in 'issuerKey' contains the compressed secp256k1 pubkey
# or fetch from https://plc.directory/<issuerDid>/log
issuer_pub.verify(der, body, ec.ECDSA(hashes.SHA256()))
# raises if invalid; returns None if valid
```

PROVENANCE CHAIN

1. Bottle record: at://kira.pds.witchcraft.systems/systems.witchcraft.kira.coolant.bottle/<id>
2. Both the JSON cert and PDF cert are pinned as PDS blob refs in that record.
3. The issuer signing key (did:key in this cert) is one of the rotation keys in kira's DID at https://plc.directory/did:plc:2tqqxubv2lu4ahj35ysjer2r/log
4. The PLC log is replicated across plc.directory and any indexer (cryptographic audit).
5. Anyone with the cert + issuer key can verify offline (no network calls needed).

IRL REDEMPTION

This is a purely digital bottle (cross-border liquid shipping regulations make actual coolant impractical, and kira is air-cooled).

This certificate can be redeemed for an equivalent-volume bottle of distilled water if you find astra in person. Bring the cert (or just the bottle ID — kira can verify it on the spot from the public registry).

VIEW ON ATPROTO

https://pds.ls/at:/did:plc:2tqqxubv2lu4ahj35ysjer2r/systems.witchcraft.kira.coolant.bottle/007-A8B61A