W3Sign Oil Demo

Step-by-Step Signing and Independent Verification Walkthrough

Plain demonstration document for the crude-oil signing demo

Prepared: 26 June 2026

1. Demo Purpose

This demonstration shows how W3Sign can create a tamper-evident evidence trail for a high-value commercial transaction.

The example transaction is the sale of 500,000 barrels of crude oil on COD terms.

At every stage, the signed document can be retrieved from IPFS and hashed using SHA-256. The resulting hash is compared against the expected hash recorded for that signing state.

2. Transaction Overview

- Transaction: Sale of crude oil
- Quantity: 500,000 barrels
- Terms: COD
- Parties: Seller, Buyer, and Transport Company

Signature stages

- Seller and buyer sign the original contract.
- Transport company signs on collection.
- Transport company signs on loading.
- Transport company signs on delivery.
- Buyer signs on final receiving.

3. What W3Sign Proves

W3Sign is not only capturing a signature. It is recording a verifiable evidence state for each signature event.

- Who signed.
- Which signing namespace created the event.
- What document was signed.
- Which version of the document was signed.
- Where the signed document can be retrieved.
- What the SHA-256 hash should be.
- Whether the retrieved document still matches the signed state.

4. Verified Document States

The test output later in this document shows that all five document states were successfully downloaded from IPFS and matched against their expected hashes.

Stage	Action	Document	Expected result	Status
1A	Seller signs contract	oil1.docx	seller/buyer contract state verified	**PASS**
1B	Buyer signs contract	oil1.docx	same document hash as seller contract state	**PASS**
2	Transport signs on collection	oil1-collected.docx	collection state verified	**PASS**
3	Transport signs on loading	oil1-loaded.docx	loading state verified	**PASS**
4	Transport signs on delivery	oil1-delivered.docx	delivery state verified	**PASS**
5	Buyer signs on receiving	oil1-received.docx	final received state verified	**PASS**

5. Step-by-Step Walkthrough

Stage 1A - Seller signs the original contract

The seller signs the original contract for the sale of 500,000 barrels of crude oil on COD terms.

Party: `Seller`

Signed document: `oil1.docx`

Signing event URL: https://www.wiiicoin.io/api/wg?n=NgXeTMikfEwae4qNMLNPdZMNreET6rLVV2&k=oil1.json

IPFS document URL: https://ipfs.w3sign.io/ipfs/Qmdu4nyaiCs4EdX5EUZjbvHo7NCAijuHiAie4Tabg2XbyC

Expected SHA-256: `670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8`

Verification: download the IPFS document, run SHA-256, and compare the output against the expected hash above. The successful test output confirms this stage passes.

Stage 1B - Buyer signs the same contract

The buyer signs the same contract. Both buyer and seller signing events point to the same signed document state.

Party: `Buyer`

Signed document: `oil1.docx`

Signing event URL: https://www.wiiicoin.io/api/wg?n=NRrdrcoosujNi9FU1jznezpKcbySGkb2Y3&k=oil1.json

IPFS document URL: https://ipfs.w3sign.io/ipfs/Qmdu4nyaiCs4EdX5EUZjbvHo7NCAijuHiAie4Tabg2XbyC

Expected SHA-256: `670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8`

Verification: download the IPFS document, run SHA-256, and compare the output against the expected hash above. The successful test output confirms this stage passes.

Stage 2 - Transport company signs on collection

The transport company signs to confirm that the oil has been collected.

Party: `Transport Company`

Signed document: `oil1-collected.docx`

Signing event URL: https://www.wiiicoin.io/api/wg?n=NesAt2DTZVukjLcTfNp4NkWoMgL2Wxfew8&k=oil1-collected.json

IPFS document URL: https://ipfs.w3sign.io/ipfs/Qma6feGC9pAAijpwj3eQ2WUjMrh9B9Q88aArDuGz2Cpwuh

Expected SHA-256: `3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36`

Verification: download the IPFS document, run SHA-256, and compare the output against the expected hash above. The successful test output confirms this stage passes.

Stage 3 - Transport company signs on loading

The transport company signs to confirm that the oil has been loaded.

Party: `Transport Company`

Signed document: `oil1-loaded.docx`

Signing event URL: https://www.wiiicoin.io/api/wg?n=NesAt2DTZVukjLcTfNp4NkWoMgL2Wxfew8&k=oil1-loaded.json

IPFS document URL: https://ipfs.w3sign.io/ipfs/QmXZMYFjgnKMHbGSX9srPTB3q97gCEN8Kk7NhFLFp9xvSL

Expected SHA-256: `9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6`

Verification: download the IPFS document, run SHA-256, and compare the output against the expected hash above. The successful test output confirms this stage passes.

Stage 4 - Transport company signs on delivery

The transport company signs to confirm delivery of the oil.

Party: `Transport Company`

Signed document: `oil1-delivered.docx`

Signing event URL: https://www.wiiicoin.io/api/wg?n=NesAt2DTZVukjLcTfNp4NkWoMgL2Wxfew8&k=oil1-delivered.json

IPFS document URL: https://ipfs.w3sign.io/ipfs/QmbrXGxVKcCQNbumtqajFEZiTXAxzSXp1pGULuNE1myzHf

Expected SHA-256: `9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18`

Verification: download the IPFS document, run SHA-256, and compare the output against the expected hash above. The successful test output confirms this stage passes.

Stage 5 - Buyer signs on receiving

The buyer signs to confirm final receipt of the crude oil.

Party: `Buyer`

Signed document: `oil1-received.docx`

Signing event URL: https://www.wiiicoin.io/api/wg?n=NRrdrcoosujNi9FU1jznezpKcbySGkb2Y3&k=oil1-received.json

IPFS document URL: https://ipfs.w3sign.io/ipfs/QmWCzYwjZyEGZZdS66fAREsw2RYxVJBUAjTZNRm9H2qDfo

Expected SHA-256: `1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e`

Verification: download the IPFS document, run SHA-256, and compare the output against the expected hash above. The successful test output confirms this stage passes.

6. How the Independent Verification Works

- Open the signing event to identify the document state and IPFS document link.
- Download the document from IPFS.
- Run SHA-256 against the downloaded document.
- Compare the actual hash against the expected signing-state hash.
- If the hashes match, the document retrieved from IPFS is the same document state recorded by W3Sign.
- If the hashes do not match, the document has changed or the wrong file has been retrieved.

7. Cross-Platform Verification Scripts

Use the script that matches your computer. Each script does the same thing: it downloads the five IPFS documents, calculates SHA-256, compares each result, and stops immediately if any hash fails.

Platform	Script file	Hash command used
Linux	run_test.sh	sha256sum
macOS	run_test_macos.sh	shasum -a 256
Windows PowerShell	run_test_windows.ps1	Get-FileHash -Algorithm SHA256

7.1 Linux script

Use this on Linux systems. It uses curl and sha256sum.

Save the following as run_test.sh.

Run commands

```
chmod +x run_test.sh
./run_test.sh
```

Script contents

```
#!/usr/bin/env bash
set -e

declare -A urls
declare -A expected

urls["oil1.docx"]="https://ipfs.w3sign.io/ipfs/Qmdu4nyaiCs4EdX5EUZjbvHo7NCAijuHiAie4Tabg2XbyC"
expected["oil1.docx"]="670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8"

urls["oil1-collected.docx"]="https://ipfs.w3sign.io/ipfs/Qma6feGC9pAAijpwj3eQ2WUjMrh9B9Q88aArDuGz2Cpwuh"
expected["oil1-collected.docx"]="3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36"

urls["oil1-loaded.docx"]="https://ipfs.w3sign.io/ipfs/QmXZMYFjgnKMHbGSX9srPTB3q97gCEN8Kk7NhFLFp9xvSL"
expected["oil1-loaded.docx"]="9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6"

urls["oil1-delivered.docx"]="https://ipfs.w3sign.io/ipfs/QmbrXGxVKcCQNbumtqajFEZiTXAxzSXp1pGULuNE1myzHf"
expected["oil1-delivered.docx"]="9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18"

urls["oil1-received.docx"]="https://ipfs.w3sign.io/ipfs/QmWCzYwjZyEGZZdS66fAREsw2RYxVJBUAjTZNRm9H2qDfo"
expected["oil1-received.docx"]="1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e"

for file in \
  "oil1.docx" \
  "oil1-collected.docx" \
  "oil1-loaded.docx" \
  "oil1-delivered.docx" \
  "oil1-received.docx"
do
  echo "Checking $file"
  curl -L "${urls[$file]}" -o "$file"

  actual=$(sha256sum "$file" | awk '{print $1}')

  echo "Expected: ${expected[$file]}"
  echo "Actual:   $actual"

  if [ "$actual" = "${expected[$file]}" ]; then
    echo "PASS: $file matches the signing-event hash"
  else
    echo "FAIL: $file does not match the signing-event hash"
    exit 1
  fi

  echo
done

echo "All W3Sign oil demo documents verified successfully."
```

Successful output pattern

```
./run_test.sh
Checking oil1.docx
Expected: 670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
Actual:   670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
PASS: oil1.docx matches the signing-event hash

Checking oil1-collected.docx
Expected: 3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
Actual:   3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
PASS: oil1-collected.docx matches the signing-event hash

Checking oil1-loaded.docx
Expected: 9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
Actual:   9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
PASS: oil1-loaded.docx matches the signing-event hash

Checking oil1-delivered.docx
Expected: 9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
Actual:   9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
PASS: oil1-delivered.docx matches the signing-event hash

Checking oil1-received.docx
Expected: 1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
Actual:   1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
PASS: oil1-received.docx matches the signing-event hash

All W3Sign oil demo documents verified successfully.
```

7.2 macOS script

Use this on macOS. It avoids Bash associative arrays so it works with the older default macOS Bash, and it uses shasum -a 256.

Save the following as run_test_macos.sh.

Run commands

```
chmod +x run_test_macos.sh
./run_test_macos.sh
```

Script contents

```
#!/bin/bash
set -e

while IFS='|' read -r file url expected; do
  echo "Checking $file"
  curl -L "$url" -o "$file"

  actual=$(shasum -a 256 "$file" | awk '{print $1}')

  echo "Expected: $expected"
  echo "Actual:   $actual"

  if [ "$actual" = "$expected" ]; then
    echo "PASS: $file matches the signing-event hash"
  else
    echo "FAIL: $file does not match the signing-event hash"
    exit 1
  fi

  echo
done <<'DOCS'
oil1.docx|https://ipfs.w3sign.io/ipfs/Qmdu4nyaiCs4EdX5EUZjbvHo7NCAijuHiAie4Tabg2XbyC|670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
oil1-collected.docx|https://ipfs.w3sign.io/ipfs/Qma6feGC9pAAijpwj3eQ2WUjMrh9B9Q88aArDuGz2Cpwuh|
3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
oil1-loaded.docx|https://ipfs.w3sign.io/ipfs/QmXZMYFjgnKMHbGSX9srPTB3q97gCEN8Kk7NhFLFp9xvSL|
9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
oil1-delivered.docx|https://ipfs.w3sign.io/ipfs/QmbrXGxVKcCQNbumtqajFEZiTXAxzSXp1pGULuNE1myzHf|
9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
oil1-received.docx|https://ipfs.w3sign.io/ipfs/QmWCzYwjZyEGZZdS66fAREsw2RYxVJBUAjTZNRm9H2qDfo|
1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
DOCS

echo "All W3Sign oil demo documents verified successfully."
```

Successful output pattern

```
./run_test_macos.sh
Checking oil1.docx
Expected: 670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
Actual:   670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
PASS: oil1.docx matches the signing-event hash

Checking oil1-collected.docx
Expected: 3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
Actual:   3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
PASS: oil1-collected.docx matches the signing-event hash

Checking oil1-loaded.docx
Expected: 9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
Actual:   9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
PASS: oil1-loaded.docx matches the signing-event hash

Checking oil1-delivered.docx
Expected: 9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
Actual:   9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
PASS: oil1-delivered.docx matches the signing-event hash

Checking oil1-received.docx
Expected: 1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
Actual:   1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
PASS: oil1-received.docx matches the signing-event hash

All W3Sign oil demo documents verified successfully.
```

7.3 Windows PowerShell script

Use this on Windows PowerShell. The execution-policy command only applies to the current PowerShell session.

Save the following as run_test_windows.ps1.

Run commands

```
Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass
.\run_test_windows.ps1
```

Script contents

```
$ErrorActionPreference = "Stop"

$tests = @(
  @{
    File = "oil1.docx"
    Url = "https://ipfs.w3sign.io/ipfs/Qmdu4nyaiCs4EdX5EUZjbvHo7NCAijuHiAie4Tabg2XbyC"
    Expected = "670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8"
  },
  @{
    File = "oil1-collected.docx"
    Url = "https://ipfs.w3sign.io/ipfs/Qma6feGC9pAAijpwj3eQ2WUjMrh9B9Q88aArDuGz2Cpwuh"
    Expected = "3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36"
  },
  @{
    File = "oil1-loaded.docx"
    Url = "https://ipfs.w3sign.io/ipfs/QmXZMYFjgnKMHbGSX9srPTB3q97gCEN8Kk7NhFLFp9xvSL"
    Expected = "9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6"
  },
  @{
    File = "oil1-delivered.docx"
    Url = "https://ipfs.w3sign.io/ipfs/QmbrXGxVKcCQNbumtqajFEZiTXAxzSXp1pGULuNE1myzHf"
    Expected = "9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18"
  },
  @{
    File = "oil1-received.docx"
    Url = "https://ipfs.w3sign.io/ipfs/QmWCzYwjZyEGZZdS66fAREsw2RYxVJBUAjTZNRm9H2qDfo"
    Expected = "1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e"
  }
)

foreach ($test in $tests) {
  Write-Host "Checking $($test.File)"
  Invoke-WebRequest -Uri $test.Url -OutFile $test.File

  $actual = (Get-FileHash -Algorithm SHA256 -Path $test.File).Hash.ToLower()

  Write-Host "Expected: $($test.Expected)"
  Write-Host "Actual:   $actual"

  if ($actual -eq $test.Expected) {
    Write-Host "PASS: $($test.File) matches the signing-event hash"
  }
  else {
    Write-Host "FAIL: $($test.File) does not match the signing-event hash"
    exit 1
  }

  Write-Host ""
}

Write-Host "All W3Sign oil demo documents verified successfully."
```

Successful output pattern

```
.\run_test_windows.ps1
Checking oil1.docx
Expected: 670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
Actual:   670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
PASS: oil1.docx matches the signing-event hash

Checking oil1-collected.docx
Expected: 3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
Actual:   3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
PASS: oil1-collected.docx matches the signing-event hash

Checking oil1-loaded.docx
Expected: 9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
Actual:   9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
PASS: oil1-loaded.docx matches the signing-event hash

Checking oil1-delivered.docx
Expected: 9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
Actual:   9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
PASS: oil1-delivered.docx matches the signing-event hash

Checking oil1-received.docx
Expected: 1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
Actual:   1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
PASS: oil1-received.docx matches the signing-event hash

All W3Sign oil demo documents verified successfully.
```

8. Verified Test Result Summary

The successful test run confirms that each IPFS document hash matches the expected W3Sign signing-event hash.

Document	Actual SHA-256 from test run	Result
oil1.docx	670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8	**PASS**
oil1-collected.docx	3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36	**PASS**
oil1-loaded.docx	9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6	**PASS**

oil1-delivered.docx	9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18	**PASS**
oil1-received.docx	1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e	**PASS**

Clean terminal output, with transfer-progress noise omitted for readability:

```
./run_test.sh
Checking oil1.docx
Expected: 670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
Actual:   670d57772e9c215a55c52e5493e5a8f01e385145bd02ad0fac0f1d328c55c1c8
PASS: oil1.docx matches the signing-event hash

Checking oil1-collected.docx
Expected: 3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
Actual:   3cc6f5f0c8d33fc1a10b40c294dab581093e8464eb26a558cf5fc878a3a7da36
PASS: oil1-collected.docx matches the signing-event hash

Checking oil1-loaded.docx
Expected: 9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
Actual:   9c11fd705ef8c41eaa0b6b4e4282de6bb0311c70db9a49ec30f1a70e7ce741b6
PASS: oil1-loaded.docx matches the signing-event hash

Checking oil1-delivered.docx
Expected: 9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
Actual:   9d2af67b2d989647e7417b59ce65ca0361d53993e981db24a563fa9d45557b18
PASS: oil1-delivered.docx matches the signing-event hash

Checking oil1-received.docx
Expected: 1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
Actual:   1c7c783d1e419e6d6d48069612c3af070a390dce28c48fba51fa06579ac8052e
PASS: oil1-received.docx matches the signing-event hash

All W3Sign oil demo documents verified successfully.
```

9. Demo Closing Statement

This oil demo shows that W3Sign can record a complete signing lifecycle across contract signing, collection, loading, delivery, and final receipt.

The important point is that the evidence can be independently tested: retrieve the document from IPFS, hash it using SHA-256, and compare it with the expected signing-state hash.

No blind trust. No platform dependency. Independent verification. Tamper-evident evidence.