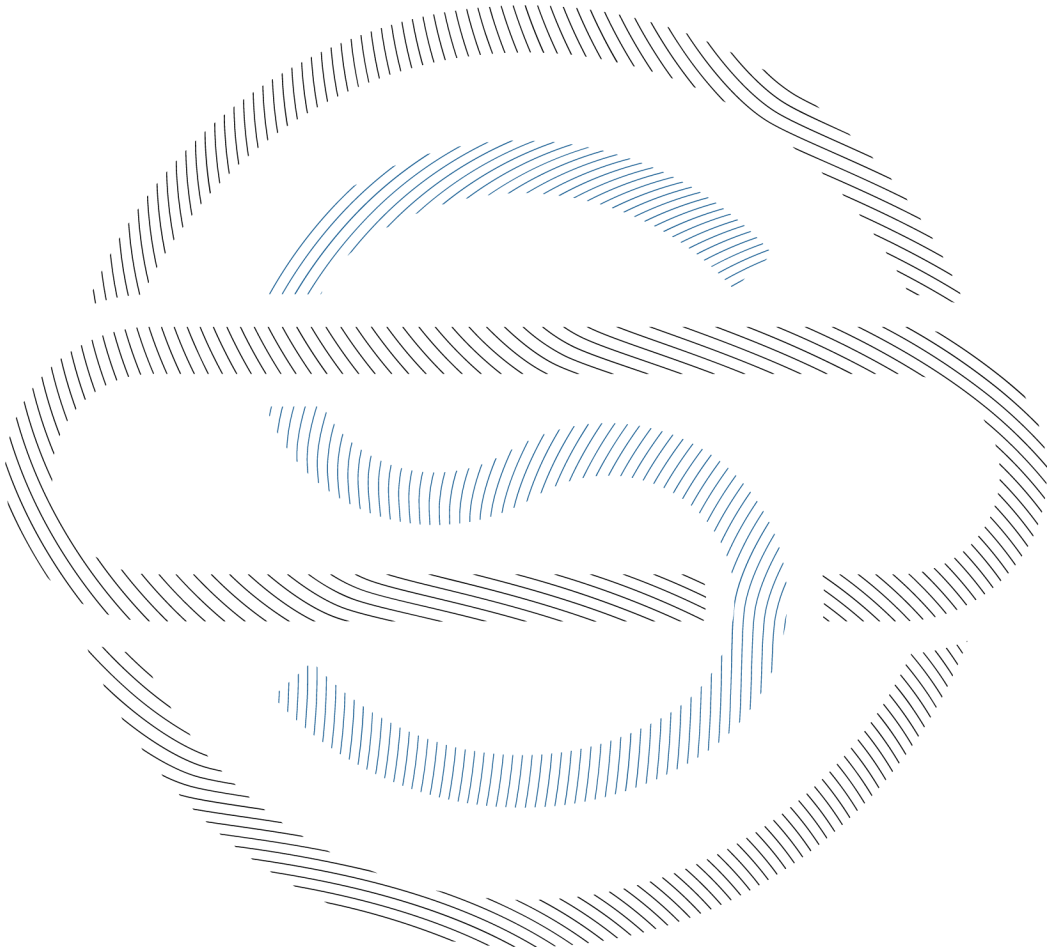
Timestamp Certificate

This timestamp was created with *Bitcoin*



originstamp

Timestamp

May-07-2020 00:01:22 UTC

Hash:

5d329ca42c57bab16b7932b8a1f7d5817bfbf1eecac9a02e5323277e34b4f7c0

Transaction:

75e77393abf090af06f79901d581dac469e1243741684d4f6badc7f6774033ab

Private Key:

9a658861cb83a3be71bf3f1ab313fa9bb17b042bbd76cab1a16cd25725e53346

Note:

testfile_originstamp_2020_05_06.txt

originstamp

Timestamp Certificate

Proof

The proof is necessary for the reproducibility of your document.


```
<?xml version="1.0" encoding="UTF-8"?>
<node type="key" value="9a658861cb83a3be71bf3f1ab313fa9bb17b042bbd76cab1a16cd25725e53346">
 <left type="mesh" value="d4419f4faf8ef7c080c83d6f5fa51a6e5dca4cee31858b4342bb16047c7d6138">
  <left type="mesh" value="648b01057f2ae9b2cff48561ac3cedc60dda107363f9d531fa85ebbbb048e316">
   <left type="mesh" value="77ae4514409dba3d63f913ea8906a448a724f7ce49a5d1ba7714638422750c7d"/>
   <right type="mesh" value="7d2dae2efa558f3d9de1a069112ad2bdb9e9b4d7ac8aa53d4d9069b0b5c69d36">
    <left type="mesh" value="6d146c6fb2058a6a3d1faec1938b50d6c92cbd650a6a5c2a7d40ce910901c5c3"/>
    <right type="mesh" value="94f58b709cf565c4d48e501ce44b314534af6730cfc3d24ddd2dc6a02f8f9e07">
     <left type="mesh" value="6bed8378283efbb4baac211004ad8775375c8b23d9d18849d58bbd8cfa63e72b">
      <left type="mesh" value="eec0e2d409dcd6f8bacf5cb306cc349bf98b89c0e8b519cb650d4198803748a8"/>
      <right type="mesh" value="c9b7c17a50af7bdbcdac4ae5a2db32794a13fe7e02512317f5f62f3179e8462a">
       <left type="mesh" value="09003e2a4f8b9a6b1e6b0ecdab107532fbe427323be3dc9e832c425092f812ee">
        <left type="mesh" value="68787da1e6a3253a16143bf7ff9dc78896c34588b6582d8d5260f9c5190d8196"/>
        <right type="mesh" value="7c62654cb149394760d546b2381b07a00e480577f7a6da085589ccec1cab3f42">
         <left type="mesh" value="f08151ba260194810c7a3038b12127b7487c95736c195d9130626dcd448469e9"/>
         <right type="mesh" value="72fca32f66d675cd1e79e4c8169c88bb2ebd89ba519485f019355bc7784a805b">
          <left type="mesh" value="001f7d1925df347dfc2b1ced71958d5ebbdb273a90b8fc686a0c92fd7071a6d9"/>
          <right type="mesh" value="f1b1a2f56d166684693e35761d28d1e3ccff434fb4f701f0996b3f450ffd1c6e">
           <left type="mesh" value="d8b2952d86d2b159b8175b8b9803e3c751ac2acc21c2f469d06dfbaddeb9484d"/>
           <right type="mesh" value="164a78d48c112126bc5ebaff2cafab43b0c89ceac8bbfbe08be386b488f54b95">
            <left type="mesh" value="ca3d266ee8e6f1faf2963cae860a051170b3698ddd9a8ef6adc8166ec259b98f">
             <left type="mesh" value="b2c779609fae54cd48dd495302fc033a67462311c8c67e834a40841a34aa8e1f">
              <left type="mesh" value="6d61a6fe08b303ed258e70841f4dacddbdec83a56a974ffe4d47f3223bdcdf93"/>
              <right type="mesh" value="bccbaecead3e832967e9f7a678e3ebf868325a239012cbf4d61f7cd60d1fb99f">
               <left type="mesh" value="b926ae577bce7784676d9c7085bb4be4ed10b7911d880f035b7fc3aaffc43a17"/>
               <right type="mesh" value="a8f6abde40b97cb0e960815f13d88c6594cd3b9c105cecbf838362f841543e7c">
                <left type="hash" value="5d329ca42c57bab16b7932b8a1f7d5817bfbf1eecac9a02e5323277e34b4f7c0"/>
                <right type="mesh" value="5d33dded8099097c8fc6a6f4378b6bf9617aaed139b860d8bd26b677a8283280"/>
               </right>
              </right>
             </left>
             <right type="mesh" value="57a19662e6c8e129f13b649bc86fe84f4b398f87f25754d983fe549d5e7ca283"/>
            </left>
            <right type="mesh" value="bdacf3596045913f58adf61887101bdc1ade838b60e4cf9677ba19ba7dedaabb"/>
           </right>
          </right>
         </right>
        </left>
        <right type="mesh" value="fb4cf811bb15eb3c484bad72556b95f96e46ebc712b508c36269ecf3fcb54b52"/>
       </right>
      </left>
      <right type="mesh" value="678eb5309b7e950558f983978668212e51707774fe0bcfc3d904e15f50be1ad7"/>
     </right>
    </right>
   </left>
   <right type="mesh" value="7f5a65ac068b83450e5783db0aa1c32e442b5f14a4b7fd795e4e9ac9977234a5"/>
  </left>
  <right type="mesh" value="5399eb93c807f68be1ff4cefaf42b74889fab4652befdcb264e7afc5c6e4ff7f"/>
</node>
```

originstamp

Timestamp Certificate

Verification

OriginStamp is a timestamp service that uses various blockchains like the Bitcoin Blockchain to create tamper-proof timestamps for your data. Instead of backing up your data, OriginStamp embeds a cryptographic fingerprint in the blockchain. It is de facto impossible to deduce the content of your data from your fingerprint. Therefore, the data remains in your company and is not publicly accessible. All you need to do is send this fingerprint to OriginStamp via the interface. The integration of the RESTful API is very simple and convenient and allows all data to be easily tagged with a tamper-proof timestamp. This document shows the procedure and gives instructions for verifying a timestamp created with OriginStamp.

1. Determine the SHA-256 of your original file

There are numerous programs and libraries to calculate the SHA-256 of a file, such as MD5FILE.COM. Simply drag and drop or select your file, to retrieve the SHA-256 of your file.

2. Validate your proof

First, it must be verified that the hash of the original is part of the evidence. In order to check this, the proof can be opened with a conventional editor and its content can be searched for the hash. If the hash cannot be found, either the file was manipulated or the wrong evidence was selected.

3. Determine the private key

The Merkle tree is a tree structure, that allows to organize the seed more efficient than a plain-text seed file. The tree is built from the bottom to the top and follows a defined schema. The value of a node is determined by the aggregated hash of its children.

Left child = a8eb9f308b08397df77443697de4959c156fd4c68c489995163285dbd3eedaef

Right child = ab95adaee8eb02219d556082a7f4fb70d19b8000097848112eb85b1d2fca8f67

Node = SHA-256(a8eb9f308b08397df77443697de4959c156fd4c68c489995163285dbd3eedaefab95adaee8eb02219d556082a7f4fb70d19b8000097848112eb85b1d2fca8f67) = 47e47c96302eeba62ed443dd0c89b3411bbddd2c1ff6bdfb1f833fa11e060b85

This step is performed for all levels of the tree until the hash of the root has been calculated. If the hash of the root is identical as proof, the calculation was successful and the root hash is verified. The top hash corresponds to the private key we embedded in the blockchain through a transaction. For a more detailed explanation of the Merkle tree, we want to refer to Wikipedia.

4. Determine the Bitcoin address

Having determined the private key in the previous step, we can use this private key for a new Bitcoin address. The detailed steps to calculate the uncompressed Bitcoin address can be found here. Lets assume the private key is *4eac8a92f8846ea801669b5834aa733e5345cc5e90875152ea6b9f38c724701e*. The calculated Bitcoin Address is *1CV9tyNSdzrKFC2gtpx3Y5GU9rPWb81R4T*.

5. Check the transactions

Check the transactions. By using any blockexplorer for Bitcoin, you can search for the previously calculated Bitcoin address: *1CV9tyNSdzrKFC2gtpx3Y5GU9rPWb81R4T*. The first transaction for this address testifies to the proof of existence. The timestamp of the file corresponds to the block time of the first transaction.