**# Signatures**

**The Safe supports different types of signatures. All signatures are combined into a single `bytes` and transmitted to the contract when a transaction should be executed.**

**### Encoding**

**Each signature has a constant length of 65 bytes. If more data is necessary it can be appended to the end of concatenated constant data of all signatures. The position is encoded into the constant length data.**

**Constant part per signature: `{(max) 64-bytes signature data}{1-byte signature type}`**

**All the signatures are sorted by the signer address and concatenated.**

**#### ECDSA Signature**

**`31 > signature type > 26`**

**To be able to have the ECDSA signature without the need of additional data we use the signature type byte to encode `v`.**

**\*\*Constant part:\*\***

**`{32-bytes r}{32-bytes s}{1-byte v}`**

**`r`, `s` and `v` are the required parts of the ECDSA signature to recover the signer.**

**#### `eth\_sign` signature**

**`signature type > 30`**

**To be able to use `eth\_sign` we need to take the parameters `r`, `s` and `v` from calling `eth\_sign` and set `v = v + 4`**

**\*\*Constant part:\*\***

**`{32-bytes r}{32-bytes s}{1-byte v}`**

**`r`, `s` and `v`are the required parts of the ECDSA signature to recover the signer. `v` will be substracted by `4` to calculate the signature.**

**#### Contract Signature \(EIP-1271\)**

**`signature type == 0`**

**\*\*Constant part:\*\***

**`{32-bytes signature verifier}{32-bytes data position}{1-byte signature type}`**

**\*\*Signature verifier\*\* - Padded address of the contract that implements the EIP 1271 interface to verify the signature**

**\*\*Data position\*\* - Position of the start of the signature data \(offset relative to the beginning of the signature data\)**

**\*\*Signature type\*\* - 0**

**\*\*Dynamic part \(solidity bytes\):\*\***

**`{32-bytes signature length}{bytes signature data}`**

**\*\*Signature data\*\* - Signature bytes that are verified by the signature verifier**

**The method `signMessage` can be used to mark a message as signed on-chain.**

**#### Pre-Validated Signatures**

**`signature type == 1`**

**\*\*Constant Part:\*\***

**`{32-bytes hash validator}{32-bytes ignored}{1-byte signature type}`**

**\*\*Hash validator\*\* - Padded address of the account that pre-validated the hash that should be validated. The Safe keeps track of all hashes that have been pre validated. This is done with a \*\*mapping address to mapping of bytes32 to boolean\*\* where it is possible to set a hash as validated by a certain address \(hash validator\). To add an entry to this mapping use `approveHash`. Also if the validator is the sender of transaction that executed the Safe transaction it is \*\*not\*\* required to use `approveHash` to add an entry to the mapping. \(This can be seen in the [Team Edition tests](**[**https://github.com/gnosis/safe-contracts/blob/v1.0.0/test/gnosisSafeTeamEdition.js**](https://github.com/gnosis/safe-contracts/blob/v1.0.0/test/gnosisSafeTeamEdition.js)**)\)**

**\*\*Signature type\*\* - 1**

**### Examples**

**Assuming that three signatures are required to confirm a transaction where one signer uses an EOA to generate a ECDSA signature, another a contract signature and the last a pre-validated signature:**

**We assume that the following addresses generate the following signatures:**

**1. `0x3` \(EOA address\) -&gt; `bde0b9f486b1960454e326375d0b1680243e031fd4fb3f070d9a3ef9871ccfd5` \(r\) + `7d1a653cffb6321f889169f08e548684e005f2b0c3a6c06fba4c4a68f5e00624` \(s\) + `1c` \(v\)**

**2. `0x1` \(EIP-1271 validator contract address\) -&gt; `0000000000000000000000000000000000000000000000000000000000000001` \(address\) + `00000000000000000000000000000000000000000000000000000000000000c3` \(dynamic position\) + `00` \(signature type\)**

**- The contract takes the following `bytes` \(dynamic part\) for verification `00000000000000000000000000000000000000000000000000000000deadbeef`**

**3. `0x2` \(Validator address\) -&gt; `0000000000000000000000000000000000000000000000000000000000000002` \(address\) +`0000000000000000000000000000000000000000000000000000000000000000` \(padding - not used\) + `01` \(signature type\)**

**The constant parts need to be sorted so that the recovered signers are sorted \*\*ascending\*\* \(natural order\) by address \(not checksummed\).**

**The signatures bytes used for `execTransaction` would therefore be the following:**

**```text**

**"0x" +**

**"000000000000000000000000000000000000000000000000000000000000000100000000000000000000000000000000000000000000000000000000000000c300" + // encoded EIP-1271 signature**

**"0000000000000000000000000000000000000000000000000000000000000002000000000000000000000000000000000000000000000000000000000000000001" + // encoded pre-validated signature**

**"bde0b9f486b1960454e326375d0b1680243e031fd4fb3f070d9a3ef9871ccfd57d1a653cffb6321f889169f08e548684e005f2b0c3a6c06fba4c4a68f5e006241c" + // encoded ECDSA signature**

**"000000000000000000000000000000000000000000000000000000000000000800000000000000000000000000000000000000000000000000000000deadbeef"     // length of bytes + data of bytes**