Quantstamp

SSV Signer

Executive Summary

This audit report was prepared by Quantstamp, the leader in blockchain security.

Type	Off-Chain Component
Timeline	2025-05-05 through 2025-05-15
Language	Go
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	None
Source Code	• ssvlabs/ssv #09f8ae9
Auditors	• Mostafa Yassin Auditing Engineer • Jennifer Wu Auditing Engineer • Mustafa Hasan Senior Auditing Engineer • Andy Lin Senior Auditing Engineer

Documentation quality	Medium
Test quality	High
Total Findings	21 Fixed: 15 Acknowledged: 6
High severity findings	0
Medium severity findings	3 Fixed: 3
Low severity findings	15 Fixed: 11 Acknowledged: 4
Undetermined severity findings	0
Informational findings	3 Fixed: 1 Acknowledged: 2

Summary of Findings

Fix Review

The client implemented fixes for the most impactful issues, as well as for the vast majority of the low severity and informational issues.

Audit Summary

This project aims to separate the beacon object attestations done by SSV validator to a separate component, the SSV Signer. The new system is composed of three components, the SSV node, the SSV signer, and the web3Signer component which handles the actual signing.

SSV Signer sets between the node and web3Signer, and its job is to route the signing requests to the web3signer in order to perform remote signing. In this setup, slashing checks are performed in the remote signer component before the request is sent to web3Signer, where it will perform another slashing check.

It is also possible to start the application with a local signer option, this setup will not need an instance of web3Signer, and the SSV signer will handle signing requests issued by the node. In this setup, the local signer will use its own local database to handle slashing checks and keep track of latest attestations.

The application can operate with multiple protocols, the most secure is the `mTLS`, which requires both the server and the client to authenticate in order to access the APIs exposed by SSV signer, such authentication is handled by the `KNOWN_CLIENTS_FILE`.

However, the application also allows communication over `TLS` and plain `HTTP`, both of which will require additional setup in order to perform client authentication and prevent arbitrary signing of beacon objects.

ID	DESCRIPTION	SEVERITY	STATUS
SSV-1	Potential Race Condition in `RemoteKeyManager` Between Signing Checks and `BumpSlashingProtection()`	Medium	Fixed
SSV-2	SSV Signer Exposes Signing Operations over Unencrypted HTTP or TLS	Medium	Fixed
SSV-3	Lack of Panic Recovery in HTTP Handler Wrapper Can Lead to Server Crash	Medium	Fixed
SSV-4	Private Key Share Leakage in `keystoreJSONFromEncryptedShare()` Error Handling	Low	Fixed
SSV-5	High CPU Usage and Potential DoS in the Add Validator Flow Due to Unbounded Share Processing	Low	Fixed
SSV-6	Flawed Error Caching in `checkCachePrivkey()` Leads to Suppressed Errors and Potential Panic	Low	Fixed
SSV-7	Integer Underflow in `computeMinimalAttestationSP()` Leads to Corrupted Slashing Data when Initializing at Epoch 0	Low	Fixed
SSV-8	Infinite Loop on Zero or Negative Batchsize	Low	Fixed
SSV-9	Inconsistent Network Configuration Usage in `NewRemoteKeyManager()` Constructor Risks Incorrect Slashing Protection and Signing Errors	Low	Fixed
SSV-10	The Order of Operations in `RemoteKeyManager.AddShare()` May Lead to Ineffective Local Slashing Protection	Low	Fixed
SSV-11	Incomplete Database State Due to Early Error Returns in Multi-Step Operations	Low	Fixed
SSV-12	Local Signer Account Overwrite Risk	Low	Acknowledged
SSV-13	Risk of Validator Slashing Due to Race Conditions in Protection Updates	Low	Fixed
SSV-14	`rsa.EncryptPKCS1v15` Decryption Is Vulnerable to Adaptive Chosen-Ciphertext Attacks	Low	Acknowledged
SSV-15	Re-Adding a Share After Removal Can Cause Slashing	Low	Acknowledged
SSV-16	Possible Local File Read	Low	Acknowledged
SSV-17	Possible Leakage of Web3signer Responses in Case of an Error	Low	Fixed
SSV-18	Insecure Logger Usage May Lead to Information Disclosure	Low	Fixed
SSV-19	Local Database Is a Single Point of Failure	Informational	Fixed

ID	DESCRIPTION	SEVERITY	STATUS
SSV-20	Server-Side Request Forgery via Web3signer_endpoint Configuration	Informational	Acknowledged
SSV-21	RSA key Size (2048 Bits) May Not Meet Long-Term Security	Informational	Acknowledged

Assessment Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

Disclaimer

Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

1. Code review that includes the following
    1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
    2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    3. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
    1. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
    2. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Scope

The scope of audit is limited to files under `ssvsigner/` directory.

Files Included

- client.go
- cmd/ssv-signer/main.go
- ekm/doc.go
- ekm/key_manager.go
- ekm/local_key_manager.go

- ekm/remote_key_manager.go
- ekm/signer_storage.go
- ekm/slashing_protector.go
- encryption/rsa_encryption.go
- keys/jemalloc_check.go
- keys/keys.go
- keys/rsa.go
- keys/rsa_linux.go
- keystore/file.go
- server.go
- types.go
- web3signer/types.go
- web3signer/web3signer.go

Repo: `https://github.com/ssvlabs/ssv`

Operational Considerations

1. In production environment, we assume that `WEB3SIGNER_ENDPOINT` is set with `https` to ensure the data is TLS encrypted. Otherwise, the password of the key share can be tracked and used to decrypt it.
2. We assume that in production, the ssvsigner service uses the mTLS to ensure that only the expected client can communicate with the service. Otherwise, there is no additional layer of authentication or authorization for this ssvsigner service.
3. We assume that the machine running the service is highly secured, as it holds the operator private key.
4. The local database is the main protection against slashing events in local key manager setup.
5. The setup can be vulnerable if the operator modifies the code or run multiple instances with non-synced database. The system assumes that the operator has enough experience and knowledge to operate the protocol.

Findings

SSV-1
Potential Race Condition in `RemoteKeyManager` Between Signing Checks and `BumpSlashingProtection()`

Medium · Fixed

> **Update**
> Fixed in `7c9aa1b66eec5fa9aa9745f819de640a972798ac`

File(s) affected: `ssvsigner/ekm/remote_key_manager.go`

Description: In `ssvsigner/ekm/remote_key_manager.go`, the `SignBeaconObject()` method calls `handleDomainAttester()` and `handleDomainProposer()`, which each perform three steps under a `signLocks` for the same `(sharePubkey, operationType)`:

1. `km.IsAttestationSlashable(sharePubkey, data)` (or `km.IsBeaconBlockSlashable(sharePubkey, slot)`)
2. `km.UpdateHighestAttestation(sharePubkey, data)` (or `km.UpdateHighestProposal(sharePubkey, slot)`)
3. Send the signing request to the remote signer

However, `BumpSlashingProtection(pubKey)`—called during `AddShare()` or other maintenance—also updates the slashing protection database (setting a minimal safe epoch/slot) but does **not** acquire the same `signLocks`. This can interleave as follows:

- A signing request reads state *S1* and passes the slashable check.
- Concurrently, `BumpSlashingProtection(pubKey)` updates state to *S2*.
- The signing request then writes its update based on *S1*, resulting in *S3*, and issues the signature—potentially allowing a slashable operation that should have been blocked by *S2*.

```
if err := km.IsAttestationSlashable(sharePubkey, data); err != nil {
    return nil, err
}
if err := km.UpdateHighestAttestation(sharePubkey, data); err != nil {
    return nil, err
}
```

A similar race exists for proposer logic in `handleDomainProposer()`.

```
if err := km.IsBeaconBlockSlashable(sharePubkey, blockSlot); err != nil {
    return nil, err
}
if err := km.UpdateHighestProposal(sharePubkey, blockSlot); err != nil {
    return nil, err
}
```

Also, we want to highlight that the lock works at the service-instance level. In other words, if two instances both run the `ssvsigner` service but share the same database, the lock will not work across those instances. We assume that `ssvsigner` is designed to run on a single instance, with the risk of reduced availability.

Recommendation: Synchronize `BumpSlashingProtection(pubKey)` with the signing locks used by `SignBeaconObject()`. For example, have `BumpSlashingProtection(pubKey)` acquire `signLocks[pubKey, "attestation"]` and `signLocks[pubKey, "proposal"]` before modifying the database. This ensures no interleaving can violate the intended check-then-update atomicity.

SSV-2

SSV Signer Exposes Signing Operations over Unencrypted HTTP or TLS

Medium Fixed

Update

Fixed in `68a75e408e4627e89f4324504fa25f4f910f07f2` by having the flag determine if HTTP is explicitly used.

File(s) affected: `ssvsigner/cmd/ssv-signer/ssv-signer.go`

Description: The SSV signer server operates in unencrypted HTTP mode when no keystore file is provided, which contradicts the documentation. The documentation for `LoadServerTLSConfig` states that with "no TLS configuration" the server still returns "minimal TLS config with modern TLS version". However, when no keystore file is provided, the server runs in completely unencrypted HTTP mode. The server exposes two signing endpoints: one that signs arbitrary data with the operator's private key share and another that forwards signing requests to web3signer. With HTTP, any client that can connect to these endpoints can request cryptographic signatures without any encryption or authentication. While proper TLS configurations (server-only or mutual TLS with client verification) are supported with SSV signer server, encryption itself is optional rather than mandatory.

Even if TLS is set, it is mainly used to authenticate the server for the client, but the client itself is not authenticated as being the specific party allowed to interact with the signer. A malicious client can use a generic certificate that is normally trusted, like from `goDaddy` for instance, and the server can accept it.

It is essential that mTLS is enforced to force client authentication and ensure that it is the party expected to be calling the signer. This can be done by using a specific certificate authority that is exclusive to SSV, or use a self signed certificate SSV and issue that to respective clients.

Recommendation: Modify the SSV signer to enforce TLS for all connections by removing the HTTP fallback option. Make mutual TLS mandatory for the signing endpoints by requiring both server and client authentication certificates. The server should not start if proper mutual TLS is not configured with appropriate client certificates. Update the configuration validation to reject insecure settings and ensure the implementation matches the documentation.

Furthermore, ensure that the certificate authority used to issue certificates cannot issue arbitrary certificates. It is also possible to use a self-signed certificate by SSV.

SSV-3

Lack of Panic Recovery in HTTP Handler Wrapper Can Lead to Server Crash

Medium Fixed

Update

Fixed in `aae11a85191606c841564f729f0cea375d0e78f0`

File(s) affected: `ssvsigner/server.go`

Description: The `Handler` method in `ssvsigner/server.go` returns a `fasthttp.RequestHandler`. This handler function wraps the invocation of `s.router.Handler(ctx)`. While it includes a `defer` statement for recording metrics, it lacks a mechanism to `recover()` from panics that might occur within the underlying route handlers (e.g., `handleAddValidator`, `handleSignValidator`, etc.).

If a panic occurs during the processing of an HTTP request (due to a bug, unexpected input causing a runtime error like a nil pointer dereference, or an issue in a dependency), and it is not recovered within the scope of the request handler goroutine, the panic will propagate. As shown in the fasthttp implementation (server.go#L165), unhandled panics in the handler will crash the entire server process.

A server crash results in a Denial of Service (DoS), making the `ssv-signer` unavailable for all users. If the condition causing the panic is easily repeatable (e.g., via a specific malformed request), an attacker could repeatedly trigger the panic and keep the service offline. This unavailability would prevent validators relying on this `ssv-signer` instance from performing their duties, leading to missed attestations/proposals and associated financial penalties (griefing, fund immobilization)

Recommendation: Implement a panic recovery mechanism within the deferred function in the `Handler` method's returned `fasthttp.RequestHandler`. This involves:

1. Calling `recover()` within the `defer` block.
2. If `recover()` returns a non-`nil` value (indicating a panic occurred):
    3. Consider logging some basic panic information.
    4. Attempt to send a generic HTTP `500 Internal Server Error` response to the client whose request triggered the panic (if headers have not already been sent).
    5. Ensure that metrics are still recorded for the request.
3. 6. s will allow the server to gracefully handle unexpected errors in individual request handlers, prevent the entire process from crashing, and maintain availability for other clients.

You can see a reference implementation here: panic handler link.

SSV-4
Private Key Share Leakage in `keystoreJSONFromEncryptedShare()` Error Handling

Low Fixed

Update
Fixed in `ba98eb8b7b663967f1176d05d458bab1199b1a68z`

File(s) affected: `ssvsigner/server.go`

Description: In `ssvsigner/server.go`, the function `keystoreJSONFromEncryptedShare()` decrypts an incoming share private key and prepares it for keystore generation. If the decrypted hex string (`sharePrivKeyHex`) fails to be decoded by `hex.DecodeString()`, the error message includes the raw `sharePrivKeyHex`, exposing sensitive key material:

```
sharePrivKey, err := hex.DecodeString(strings.TrimPrefix(string(sharePrivKeyHex), "0x"))
if err != nil {
    // VULNERABLE: sharePrivKeyHex contains the decrypted private key share
    return "", fmt.Errorf("decode share private key from hex %s: %w", string(sharePrivKeyHex),
err)
}
```

This error is logged by `handleAddValidator()` (via `logger.Warn`) and returned in the HTTP response (`s.writeJSONErr()`), causing both logs and client responses to leak the validator's share private key whenever decoding fails.

Recommendation: Modify the error handling in `keystoreJSONFromEncryptedShare()` to return a generic error message without including `sharePrivKeyHex`. For example, replace:

```
return "", fmt.Errorf("decode share private key from hex %s: %w", string(sharePrivKeyHex), err)
```

with:

```
return "", errors.New("failed to decode share private key from hex string")
```

This change ensures that neither logs nor API responses will contain the sensitive key material on decode errors.

SSV-5

High CPU Usage and Potential DoS in the Add Validator Flow Due to Unbounded Share Processing

Low Fixed

Update

Fixed in `8c7e01a429ce37a70cee02eb88f74295be58c206`

Description: The `POST /v1/validators` endpoint in ssv-signer (handled by `handleAddValidator()` in `ssvsigner/server.go`) processes incoming validator shares. For each share, it calls `keystoreJSONFromEncryptedShare()`, which calls `GenerateShareKeystore()`. The `GenerateShareKeystore()` function uses `keystorev4.New().Encrypt()` to create an EIP-2335 keystore.

The keystorev4 library uses PBKDF2 for key derivation with the following parameters:

- `pbkdf2c` (iterations): `262144`
- `pbkdf2PRF`: `"hmac-sha256"`

This high iteration count makes the encryption process CPU-intensive. Benchmarks indicate encrypting a single share takes approximately 0.5–0.8 seconds on a single CPU core. For one round, it will take around 0.5 to 0.8 seconds, according to some benchmarks on a single core.

The fasthttp server has a default request body size limit of 4 MB. Given each encrypted share key request (including public key and encrypted private key) is ~700 bytes, a single `POST /v1/validators` request could contain ~5900 shares (4,194,304 bytes / 700 bytes ≈ 5991).

Processing a large number of shares (e.g., 5000) sequentially (or concurrently) leads to:

- **Excessive CPU Consumption:** CPU heavily loaded for extended periods.
- **Long Request Latency:** ~41–66 minutes (5000 shares × 0.5–0.8 s/share).
- **Potential DoS:** Server becomes unresponsive or very slow, allowing attackers to exploit this by sending large batches of shares.

Here are the reference data:

- PBKDF2 iteration count: `262,144`
- Estimated 0.49–0.8 seconds per share encryption.
  - Reference 1 (general encryption benchmark), 329,326 iterations per second: Unix StackExchange
  - Reference 2 (GPU HMAC benchmark), 537.2 kH/s per GPU ⇒ 537,200 HMACs/sec (AWS g3.8xlarge, single GPU): Gist by alexiasa
  - `262,144 / 329,326 = ~0.7956`, `262,144 / 537,200 = ~0.486`

Recommendation: Implement a strict limit on the number of shares processed in a single `POST /v1/validators` request (e.g., 5–10 shares). Clients adding many shares must batch their requests, reducing maximum processing time and CPU load per request.

SSV-6

Flawed Error Caching in `checkCachePrivkey()` Leads to Suppressed Errors and Potential Panic

Low Fixed

Update

Fixed in `4e9720f178770478483c71d9965605ba3f4ee885`

File(s) affected: `ssvsigner/keys/rsa_linux.go`

Description: The `once.Do()` ensures that this function only runs once. However, the `err` variable is within this `checkCachePrivkey()` scope only. In other words, a local `err` variable is captured by the closure passed to `sync.Once.Do()`. If `rsaPrivateKeyToOpenSSL()` fails during the first invocation, `err` is set accordingly. But on subsequent calls, `sync.Once.Do()` does not re-run the closure and a new local `err` is initialized to nil, causing checkCachePrivkey to return `(cachedPrivKey=nil, err=nil)`. This results in `SignRSA()` being called with a `nil` OpenSSL key handle, likely leading to a panic.

Recommendation: Store both the cached key handle and any initialization error in the struct. For example add fields `opensslInitErr error` and do:

```
priv.once.Do(func() {
    priv.cachedPrivKey, priv.opensslInitErr = rsaPrivateKeyToOpenSSL(priv.privKey)
```

```
})
return priv.cachedPrivKey, priv.opensslInitErr
```

This ensures the same error is returned on every call.

SSV-7
Integer Underflow in `computeMinimalAttestationSP()` Leads to Corrupted Slashing Data when Initializing at Epoch 0

Low ⓘ Fixed

> **Update**
> Fixed in `277bb6cca3bc9651b1082b049ccf752218c46f0c`

File(s) affected: `ssvsigner/ekm/slashing_protector.go`

Description: The function `computeMinimalAttestationSP()` in `slashing_protector.go` is vulnerable to an integer underflow if called with `epoch = 0`. Given `minSPAttestationEpochGap = 0`, `highestTarget` becomes `0`, and the subsequent calculation `highestSource := highestTarget - 1` causes `highestSource` (`uint64`) to underflow to `MaxUint64`. This results in corrupted `AttestationData` (`Source.Epoch = MaxUint64`, `Target.Epoch = 0`) being used for slashing protection.

Separately, the `SaveHighestProposal()` function in `signer_storage.go` rejects attempts to save `slot = 0`. The function `computeMinimalProposerSP` calculates a minimal proposal slot which would be `0` if the current slot is `0` (as `minSPProposalSlotGap` is `0`).

If `BumpSlashingProtection` is invoked when the node's perceived current epoch is `0` (implying current slot is also likely `0` or near `0`), these issues interact:

1. The call to `computeMinimalAttestationSP(0)` would (without a fix) lead to underflowed data.
2. The call to `SaveHighestAttestation()` will save the underflowed attestation data.
3. The call to `computeMinimalProposerSP(0)` would yield `0`.
4. The attempt to save this proposal slot `0` via `SaveHighestProposal` would fail.

This results in `BumpSlashingProtection()` failing, but after the underflowed attestation data has already been calculated saved by `SaveHighestAttestation()`.

Recommendation: To ensure system integrity and align with a potential operational assumption that slashing protection initialization does not occur at `epoch/slot 0`, the functions responsible for computing these minimal baseline values should explicitly disallow calculations that result in `epoch = 0` or `slot = 0`.

1. **Modify** `computeMinimalAttestationSP`: If the input `epoch` is `0` (and `minSPAttestationEpochGap` is `0`, leading to `highestTarget` being `0`), the function should return an error indicating that baseline attestation data for `epoch = 0` is not supported by this initialization mechanism. This prevents the underflow from occurring.
2. **Modify** `computeMinimalProposerSP`: If the input `slot` is `0` (and `minSPProposalSlotGap` is `0`, leading to `minimalSPSlot` being `0`), the function should return an error indicating that baseline proposal data for `slot = 0` is not supported by this initialization mechanism.

These changes ensure that `BumpSlashingProtection` will fail cleanly if invoked at `epoch/slot 0`, with the error originating from the computation step, clearly stating the unsupported condition. The existing check in `SaveHighestProposal` that rejects `slot = 0` can remain as a defense-in-depth measure at the storage layer.

SSV-8 Infinite Loop on Zero or Negative Batchsize

Low ⓘ Fixed

> **Update**
> Fixed in `3db6b539c3f5e8710c8d1e20f5b7a130f0752e5e`

Description: In the batch deletion loop, `cli.BatchSize` is used as the loop increment without validation. If `BatchSize` is set to zero or a negative value, the loop variable `i` never advances (or moves backward), causing an infinite loop and high CPU usage.

Recommendation: Validate that `cli.BatchSize` is a positive, non-zero integer before entering the loop. If invalid, return an error. For example:

```
if cli.BatchSize <= 0 {
    return fmt.Errorf("invalid batch size %d: must be > 0", cli.BatchSize)
}
```

SSV-9

Inconsistent Network Configuration Usage in `NewRemoteKeyManager()` Constructor Risks Incorrect Slashing Protection and Signing Errors

Low Fixed

Update
Fixed in `3cb1a225741f1f9d80611aa66041055a235d1ae3`

File(s) affected: `ssvsigner/ekm/remote_key_manager.go`, `ssvsigner/ekm/signer_storage.go`

Description: The `NewRemoteKeyManager()` constructor in `ssvsigner/ekm/remote_key_manager.go` accepts two distinct `networkconfig.NetworkConfig` parameters: an initial `netCfg` and a subsequent `networkConfig`. The `RemoteKeyManager` instance stores the first parameter (`netCfg`) internally (as `km.netCfg`). This stored `km.netCfg` is used to determine the Ethereum network context (e.g., fork versions, genesis validators root via `km.getForkInfo()`) when preparing `SignRequest` payloads to be sent to the remote signing service.

However, the `SignerStorage` (and consequently the `slashingProtector`) is initialized using the second `networkConfig` parameter (`NewSignerStorage(db, networkConfig.Beacon, logger)`). This means all local slashing protection checks and database operations (which are network-specific, e.g., due to database key prefixing by network name) are performed based on this second `networkConfig`.

If these two `networkconfig.NetworkConfig` objects differ (e.g., one configured for Mainnet and the other for Holesky, or representing testnets with divergent fork schedules), a critical inconsistency arises:

1. The `ForkInfo` used for signing requests might pertain to one network.
2. The slashing protection checks might be evaluated against the historical data or rules of a different network.

This discrepancy can lead to severe consequences:

- **Bypassed Slashing Protection (False Negatives):** A genuinely slashable signing operation for the intended network (based on `netCfg`) might not be detected if the slashing protection database (based on `networkConfig`) lacks the relevant conflicting history, potentially resulting in a slashable message being signed, loss of staked funds, and validator ejection.
- **Erroneous Slashing Violations (False Positives):** Valid signing operations for the intended network might be incorrectly flagged as slashable due to conflicts with unrelated historical data from the network defined by `networkConfig`, leading to missed duties and inactivity penalties.
- **Invalid Signatures:** Signatures generated based on an incorrect `ForkInfo()` (if `netCfg` is wrong for the actual network of operation) will be rejected by the Beacon Node, causing missed duties.

Recommendation: Unify the network configuration input for `NewRemoteKeyManager()` to ensure all components operate under a single, consistent network context. Modify the `NewRemoteKeyManager()` constructor to accept only a single `networkconfig.NetworkConfig` parameter. Use this single configuration object for initializing all parts of the `RemoteKeyManager`, including its internal `netCfg` field and the `SignerStorage` (and thus the `slashingProtector`). This change will eliminate the possibility of inconsistent network configurations within the same `RemoteKeyManager()` instance, ensuring that signing context preparation and slashing protection checks are always performed against the same, correct network parameters.

SSV-10

The Order of Operations in `RemoteKeyManager.AddShare()` May Lead to Ineffective Local Slashing Protection

Low Fixed

Update
Fixed in `56c0f814e6a02a491b543e2394c62282e838c4ad`

Description: The `AddShare()` method in `ssvsigner/ekm/remote_key_manager.go` currently executes two steps in this order:

1. `km.signerClient.AddValidators(ctx, shareKeys)`: adds the validator share to the remote signing service.
2. `km.BumpSlashingProtection(pubKey)`: initializes or updates the local slashing protection database for that share.

If step 1 succeeds but step 2 fails (for example, due to a local database error), the share is active on the remote signer while its local slashing protection baseline is never set. A subsequent signing request for that share could then bypass local checks (`IsAttestationSlashable()`, `IsBeaconBlockSlashable()`), allowing a slashable operation to be sent to the remote signer.

Without the baseline established by `BumpSlashingProtection()`, the first signing request for a newly added key may not be recognized as conflicting with prior epochs or slots, undermining intended slashing protection.

Recommendation: Reverse the order of operations in `AddShare()` so that local protection is initialized before the key is activated remotely:

1. Call `km.BumpSlashingProtection(pubKey)` first.
2. Then call `km.signerClient.AddValidators(ctx, shareKeys)`.

This ensures that if `BumpSlashingProtection()` fails, the key is never added to the remote signer. If the remote call then fails, local protection is still correctly initialized, preventing any window where a slashable signing could occur.

SSV-11
Incomplete Database State Due to Early Error Returns in Multi-Step Operations

Low Fixed

> **Update**
> Fixed in `ee1a059628c9a54e16a7405649870ec90fbced23`

File(s) affected: `ssvsigner/ekm/local_key_manager.go`, `ssvsigner/ekm/slashing_protector.go`

Description: Functions in the codebase that perform multiple related database operations in sequence can leave the system in an inconsistent state if one operation fails. When operations return immediately after encountering an error, previously completed steps are not rolled back.

The following function can result in incomplete database state during early errors:

1. `LocalKeyManager.AddShare`
2. `LocalKeyManager.RemoveShare`
3. `RemoteKeyManager.AddShare`
4. `RemoteKeyManager.RemoveShare`
5. `SlashingProtector.BumpSlashingProtection`

Recommendation: Consider implementing transactions or compensating actions for multi-step operations. Either wrap related operations in a database transaction so they can be rolled back together, or implement cleanup/recovery logic to handle partial failures. For non-database operations, consider implementing a pattern where changes are prepared but only committed after all steps succeed.

SSV-12 Local Signer Account Overwrite Risk

Low Acknowledged

> **Update**
> The client left the following comment:
>
> `not implemented as it's intentional and used only in migration`

File(s) affected: `ssvsigner/ekm/signer_storage.go`

Description: The `SaveAccountTxn` method saves validator accounts (containing private keys) without checking if an account with the same ID already exists. This behavior allows silent overwriting of existing accounts without warning or confirmation, which could lead to accidental replacement of accounts with complete key information. There is no atomic update capability for specific account fields, creating potential data loss during migrations or updates.

Recommendation: Consider adding a safety check to prevent unintentional overwrites. This could be an optional parameter to explicitly allow overwrites, a check-before-write pattern that returns an error if the account exists, or at minimum, adding logging when overwriting existing accounts.

SSV-13
Risk of Validator Slashing Due to Race Conditions in Protection Updates

Low Fixed

> **Update**
> Fixed in `faaf805f5720670ea87fe555ada69afa9d4113d9`

File(s) affected: `ssvsigner/ekm/remote_key_manager.go`

Description: `RemoteKeyManager.AddShare` and `RemoteKeyManager.RemoveShare` rely on the Web3Signer service to handle concurrency for remote key management operations. However, after the remote operations succeed, these functions lack locking mechanisms and perform multiple sequential database operations that modify slashing protection data. This can result in corrupted slashing protection records during concurrent validator management, potentially failing to prevent double signing and resulting in validators being slashed.

Recommendation: Add proper mutex locking to all operations that read or modify slashing protection data, ensuring thread safety for local database updates.

SSV-14

`rsa.EncryptPKCS1v15` Decryption Is Vulnerable to Adaptive Chosen-Ciphertext Attacks

Low | Acknowledged

> **Update**
> Client left the following comments:
>
> `applied the recommendations, but we won't change the scheme soon due to compatibility; the spec team is aware`

Description: Currently, the protocol uses `PKCS1V15` in order to perform decryption of payload. This algorithm is vulnerable to an attack known as Adaptive Chosen-Ciphertext. The attack requires multiple elements to be carried out.

1. The attacker needs to obtain a valid encrypted share
2. The server need to act as an oracle for the attacker. Meaning that it should return a different error response as the attacker modifies the encrypted share payload.
3. The server needs to be running on either HTTP or TLS.

The exploit would go as the following:

- The attacker obtains the encrypted private key share of the validator.
- The attacker will modify the encrypted share, trying to probe the issue with PKCS1v15 padding,
- If the server returns a different response when the padding is incorrect, then the server functions as an oracle for the attacker. For instance, the server might return "invalid-padding" instead of a generic message like "an error occurred".
- Attacker can continue carrying about the attack through modifying the encrypted share and sending more requests until they eventually recover the private key

Recommendation: Since changing the decryption scheme can causing issues with computability, consider the following:

- Enforcing mTLS will mitigate this issue.
- Enforcing the feedback from the server, in cases of errors, to be generic will also mitigate this issue.

SSV-15

Re-Adding a Share After Removal Can Cause Slashing

Low | Acknowledged

> **Update**
> The client left the following comments:
>
> `we have a solution but it's too big and risky to insert in last second and we need more time for it`

File(s) affected: `ssvsigner/ekm/local_key_manager.go`

Description: Currently, when a share is removed from the protocol, its highest attestation is also removed from the local database, and in case a local signer setup is being used, the local database is the only safeguard against double signing.

The `RemoveShare` method in `local_key_manager.go` will remove the highest proposal and the highest atttestation for the share pub key from the local database:

```
if acc != nil {
        if err := km.RemoveHighestAttestation(pubKey); err != nil {
            return fmt.Errorf("could not remove highest attestation: %w", err)
        }
        if err := km.RemoveHighestProposal(pubKey); err != nil {
            return fmt.Errorf("could not remove highest proposal: %w", err)
```

```
        }
        if err := km.wallet.DeleteAccountByPublicKey(pubKeyHex); err != nil {
            return fmt.Errorf("could not delete share: %w", err)
        }
    }
```

If the same share is re-added again, the database will have no history of its signing history. This is not an issue if a remote signer is used, because web3signer implements its own slashing protection. But if the same beacon object is requested to be signed, due to a system glitch, for instance, then it is possible for a slashing event to happen.

Recommendation: Consider keeping the share history in the database, especially if the local setup is used.

SSV-16 Possible Local File Read

Low · Acknowledged

Update

The client left the following comment:

> not implemented as we think this should be configured by the infra/server administrator

File(s) affected: `ssvsigner/tls/tls.go`

Description: Functions like `loadPasswordFromFile()`, `loadKeystoreCertificate()`, `loadPEMCertificate()`, and `loadFingerprintsFile()` read files from the filesystem based on user input. It may be possible for an attacker to be able to elevate this behavior to read sensitive file contents if the loading of files is reachable to an external attacker for example via an HTTP request or if the module is used locally by the client or similar and the command is run in a privileged mode by a normal user.

Recommendation: Sanitize user input so only specific directories are allowed.

SSV-17 Possible Leakage of Web3signer Responses in Case of an Error

Low · Fixed

Update

Fixed in `94cf5929f3f30d10e57c998fd3c84f0a2bc85215`

File(s) affected: `ssvsigner/server.go`

Description: The `handleWeb3SignerErr()` function logs the response received from the upstream web3signer instance. Additionally, `handleSignValidator()` has the line:

```
logger = logger.With(zap.String("req", string(ctx.PostBody())))
```

Which is triggered if there is an error returned from web3signer. This effectively leaks the full request and response bodies in the logger's output, possibly leaking signatures (in case web3signer generates a valid signature but still returns an error) and other sensitive information.

Recommendation: Make sure the request body isn't logged. The web3signer response also should not be logged in case a valid signature is included.

SSV-18 Insecure Logger Usage May Lead to Information Disclosure

Low · Fixed

Update

Fixed in `26a5b1aa1bd6d8ecfaade9b83ae913935ee11bcb`

File(s) affected: `ssvsigner/cmd/purge-keys/purge-keys.go`

Description: The `main()` function creates a logger with the `zap.NewDevelopment()` constructor. The docs at https://pkg.go.dev/go.uber.org/zap#NewDevelopment state the following:

```
NewDevelopment builds a development Logger that writes DebugLevel and above logs to standard
error in a human-friendly format.
```

That said, the logger will be run in debug mode, which will result in verbose message that may leak sensitive information.

Recommendation: Use `zap.NewProduction()` to run the logger in production mode.

SSV-19 Local Database Is a Single Point of Failure

Informational Fixed

> **Update**
> Fixed in `26a5b1aa1bd6d8ecfaade9b83ae913935ee11bcb`

Description: When the protocol operates with a local key manager, a local slashing database is used. This database functions as the main protection against slashing events. In the case that the database goes down or gets erased for any reason, if there is no proper backup, the slashing history will be erased. Continuing the operation after this point can cause slashing events.

Recommendation: Consider outlining that in user-facing documentation. Also consider various backup options to recover the database if it goes down or get erased.

SSV-20 Server-Side Request Forgery via Web3signer_endpoint Configuration

Informational Acknowledged

> **Update**
> The client left the following comments:
>
> `was implemented but reverted as it causes issues with our deployment, we use private addresses for ssv-signer and expect operators to do so`

File(s) affected: `ssvsigner/cmd/purge-keys/purge-keys.go`, `ssvsigner/web3signer/web3signer.go`, `ssvsigner/cmd/ssv-signer/ssv-signer.go`

Description: The `Web3SignerEndpoint` configuration parameter, sourced from CLI arguments or environment variables, is validated by `validateConfig` using `url.ParseRequestURI`. This validation is insufficient as `url.ParseRequestURI` allows various URL schemes and does not restrict target hosts (e.g., `localhost`, internal IPs, potentially other schemes if not filtered by the HTTP client). The validated endpoint is then used by `setupWeb3SignerClient` to instantiate a `web3signer.Web3Signer` client, which makes HTTP(S) requests. An attacker who can control the `WEB3SIGNER_ENDPOINT` value could specify an endpoint pointing to internal network services (e.g., `http://localhost:xxxx`, `http://internal-service.local/api`). This could allow the attacker to scan internal networks, access sensitive internal endpoints, or interact with internal services through the ssv-signer application, as the underlying HTTP client (`github.com/carlmjohnson/requests` wrapping `net/http`) may permit such connections.

Because the system is assumed to be communicating over mTLS with the client, the impact and likelihood is limited.

Recommendation:

1. In `ssv-signer.go::validateConfig()` and `purge-keys.go::run()`, enhance URL validation for `Web3SignerEndpoint`:
   - Restrict allowed schemes to `http` and `https`.
   - Parse the hostname. Disallow requests to loopback addresses (IPv4 `127.0.0.0/8`, IPv6 `::1/128`), link-local addresses (IPv4 `169.254.0.0/16`, IPv6 `fe80::/10`), and private IP ranges (e.g., `10.0.0.0/8`, `172.16.0.0/12`, `192.168.0.0/16`) unless explicitly permitted by a separate configuration flag (e.g., `--allow-internal-web3signer-endpoint`).
   - Consider maintaining an allowlist of trusted Web3Signer domains or IP addresses if the deployment environment allows for it.
2. Ensure the HTTP client used by `web3signer.New()` is configured to prevent following redirects to different, potentially malicious, locations or schemes if this is not its default behavior.

SSV-21
RSA key Size (2048 Bits) May Not Meet Long-Term Security

Informational | Acknowledged

> **Update**
> The client left the following comment:
> `not implemented, will require fork; spec team is aware`

File(s) affected: `ssvsigner/keys/keys.go`

Description: Keys are generated at 2048 bits, providing $112 - \mathrm{bitsec\,uritystren} > \mathrm{h}, \mathrm{aeptab} \leq \mathrm{onlythrough2030}$ per NIST SP 800-57. For longer-term use (>2030 year) or higher assurance (≥128 bits), 3072-bit RSA keys or modern ECC curves (e.g., P-256/P-384) are recommended.

This issue is currently only used for testing, so we list it as an informational one. However, if the function will be used in production, the team should be aware of this security weakness

Recommendation: Consider evaluating intended key-lifetime. For production use requiring 2030 forward security, move to 3072-bit RSA instead.

Auditor Suggestions

S1
Inconsistent Password Trimming in Keystore Handling Can Lead to Decryption Failures

Fixed

> **Update**
> Fixed in `85398c2e49a3e03a2ea08179a0cea064e69c73f9`

File(s) affected: `ssvsigner/keystore/file.go`

Description: The keystore handling code in `file.go` exhibits inconsistent treatment of leading/trailing whitespace in passwords, which can lead to unexpected decryption failures:

1. **In** `LoadOperatorKeystore()`**:**
   - The password is read from `PASSWORD_FILE`.
   - An emptiness check is performed using `bytes.TrimSpace(keyStorePassword)`.
   - However, the original, potentially untrimmed `string(keyStorePassword)` (which might include newlines or spaces if the password file was created with, for example, `echo "password" > file.txt`) was previously passed to `DecryptKeystore()`.
2. **In** `DecryptKeystore()`**:**
   - An emptiness check on its password argument is performed using `strings.TrimSpace(password)`.
   - However, the original, potentially untrimmed `password` argument is then used for the actual decryption call to `keystorev4.New().Decrypt()`.

This mismatch means that if a password file contains a valid password surrounded by whitespace (e.g., `"mypassword\n"`), the emptiness checks might pass, but the decryption would fail because the keystore was encrypted with the password without the extraneous whitespace (e.g., `"mypassword"`).

This issue does not pose a direct security vulnerability (like key leakage) but is a usability and robustness concern. It can lead to operational failures where ssv-signer fails to start due to an inability to decrypt the operator keystore, causing confusion for operators who might believe their password or keystore is corrupt when it is merely an issue of extraneous whitespace.

Recommendation: Ensure that passwords are consistently trimmed of leading/trailing whitespace before being used for any validation checks or cryptographic operations:

1. In `LoadOperatorKeystore()`, trim the password read from the file once, and use this trimmed version for both the emptiness check and when calling `DecryptKeystore()`.
2. In `DecryptKeystore()`, trim the input password argument once at the beginning of the function, and use this trimmed version for both its internal emptiness check and the call to `keystorev4.New().Decrypt()`.

S2
Aes Key Derivation From Rsa Hash Lacks Proper Key Derivation Function

Fixed

> **Update**
> Fixed in `7234904964d47a660153ac44c7bed779136f7200`

File(s) affected: `ssvsigner/ekm/local_key_manager.go`, `ssvsigner/ekm/signer_storage.go`

Description: The operator's RSA private key hash (`EKMHash`) is used directly as an AES-GCM encryption key (`SetEncryptionKey`). No dedicated Key Derivation Function (KDF) is applied. While the raw hash may have sufficient entropy, using a proper KDF (e.g., HKDF, PBKDF2, Argon2) improves robustness against subtle cryptanalytic attacks and key-reuse concerns.

Recommendation: Consider applying the following: • Introduce a KDF step: derive a 32-byte AES key from the RSA key hash using HKDF-SHA256 (or Argon2id).
• Update `SetEncryptionKey` usage to accept the derived symmetric key, not the raw hash.
• Document the change and add tests to verify encryption/decryption with derived keys.

S3 Fragile Error String Comparison for Wallet-Not-Found

Fixed

> **Update**
> Fixed `2d25f400de2e5d109de63403dea02c1b79ae9834`

File(s) affected: `ssvsigner/ekm/local_key_manager.go`

Description: The `OpenWallet()` function checks for a missing wallet by comparing `err.Error() != "could not find wallet"` . Relying on exact error strings is brittle and may break if wrapped or modified.

Recommendation: Consider applying the following: • Define a sentinel error variable (e.g., `ErrWalletNotFound = errors.New("could not find wallet")`) in `signer_storage.go` .
• Use `errors.Is(err, ErrWalletNotFound)` for checking.
• Update `OpenWallet()` to return the sentinel and callers to use `errors.Is()` .

Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
- **Informational** – The issue does not pose an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.
- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **Acknowledged** – The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Files

Repo: `https://github.com/ssvlabs/ssv`

- `0b5...d16 ./.air.toml`
- `016...f1d ./.cursorrules`
- `2a8...f39 ./.dockerignore`
- `4d6...721 ./.github/ISSUE_TEMPLATE/bug_report.md`
- `6af...e94 ./.github/ISSUE_TEMPLATE/config.yml`
- `72a...d47 ./.github/ISSUE_TEMPLATE/feature_request.md`
- `19f...043 ./.github/PULL_REQUEST_TEMPLATE/pull_request_template.md`
- `84f...52f ./.github/dependabot.yml`
- `2d4...197 ./.github/workflows/e2e-test.yml`
- `1f1...6d7 ./.github/workflows/full-test.yml`
- `a26...cce ./.github/workflows/integration-test.yml`
- `82e...086 ./.github/workflows/lint.yml`
- `d81...3f8 ./.github/workflows/spec-alignment.yml`
- `045...9d3 ./.github/workflows/spec-test-raceless.yml`
- `576...42e ./.github/workflows/spec-test.yml`
- `6e8...0f3 ./.github/workflows/test-cov.yml`
- `17f...00c ./.github/workflows/unit-test.yml`
- `47e...13d ./.gitignore`
- `cc6...38f ./.golangci.yaml`
- `84b...a8b ./Dockerfile`
- `863...ffc ./Dockerfile.multiarch`
- `8b1...b9b ./LICENSE`
- `071...4c9 ./Makefile`
- `b2c...c2b ./README.md`
- `823...5dd ./ROADMAP.md`
- `083...9c0 ./api/bind.go`
- `6f5...a07 ./api/bind_test.go`
- `1e4...7cc ./api/errors.go`
- `9b4...d79 ./api/errors_test.go`
- `28f...c25 ./api/handlers/exporter.go`
- `655...966 ./api/handlers/exporter_test.go`
- `c83...256 ./api/handlers/node.go`
- `b75...c6a ./api/handlers/node_test.go`
- `2b4...73a ./api/handlers/validators.go`
- `514...122 ./api/handlers/validators_test.go`
- `ba9...2e3 ./api/handling.go`
- `a22...4aa ./api/handling_test.go`
- `180...086 ./api/server/server.go`
- `191...1d6 ./api/server/server_test.go`
- `adc...5b0 ./api/types.go`
- `661...7c7 ./api/types_test.go`
- `b8f...ff0 ./audits/Hacken_SSV_Labs_L1_SSV_Labs_SSV_Node_Aug2024_P_2024_1212_2_20241016.pdf`
- `dd8...4c8 ./audits/Least Authority.pdf`
- `7ef...417 ./beacon/goclient/WAD.md`
- `044...71d ./beacon/goclient/aggregator.go`
- `ec8...1ec ./beacon/goclient/attest.go`

- `5ac...0b6 ./beacon/goclient/attest_test.go`
- `750...633 ./beacon/goclient/committee_subscribe.go`
- `c9c...7e5 ./beacon/goclient/current_fork.go`
- `fa0...1ca ./beacon/goclient/current_fork_test.go`
- `4a3...813 ./beacon/goclient/dataversion.go`
- `cf5...6c3 ./beacon/goclient/dataversion_test.go`
- `bf5...c10 ./beacon/goclient/events.go`
- `a1b...bf4 ./beacon/goclient/events_test.go`
- `9b3...d43 ./beacon/goclient/genesis.go`
- `9b5...1c7 ./beacon/goclient/genesis_test.go`
- `448...70b ./beacon/goclient/goclient.go`
- `c3a...cc3 ./beacon/goclient/goclient_test.go`
- `7f9...f94 ./beacon/goclient/observability.go`
- `429...ba5 ./beacon/goclient/options.go`
- `fcf...6d9 ./beacon/goclient/proposer.go`
- `667...006 ./beacon/goclient/signing.go`
- `160...9b3 ./beacon/goclient/signing_test.go`
- `ee2...32b ./beacon/goclient/spec.go`
- `923...caf ./beacon/goclient/spec_test.go`
- `06d...15a ./beacon/goclient/sync_committee.go`
- `ded...40c ./beacon/goclient/sync_committee_contribution.go`
- `02c...b2c ./beacon/goclient/tests/mock-beacon-responses.json`
- `314...641 ./beacon/goclient/tests/shared.go`
- `1da...948 ./beacon/goclient/types.go`
- `58a...3be ./beacon/goclient/validator.go`
- `27b...550 ./beacon/goclient/validatorliveness.go`
- `876...4cf ./beacon/goclient/voluntary_exit.go`
- `d17...ac6 ./cli/bootnode/boot_node.go`
- `f8c...850 ./cli/cli.go`
- `d8b...370 ./cli/config/config.go`
- `603...941 ./cli/export_keys_from_mnemonic.go`
- `2d3...766 ./cli/flags/export_keys_from_mnemonic.go`
- `a5a...633 ./cli/flags/threshold.go`
- `715...212 ./cli/generate_operator_keys.go`
- `04b...8dc ./cli/operator/generate_doc.go`
- `0ac...a33 ./cli/operator/node.go`
- `942...136 ./cli/operator/node_test.go`
- `d2c...631 ./cli/threshold.go`
- `ac2...427 ./cli/version.go`
- `88d...287 ./cmd/ssvnode/main.go`
- `18d...ea5 ./codecov.yml`
- `0ce...f94 ./config/config.example.yaml`
- `7c3...f74 ./config/config.exporter.example.yaml`
- `929...2b5 ./config/events.example.yaml`
- `906...af2 ./config/example_share.yaml`
- `a21...a06 ./config/exporter.yaml`
- `121...cff ./dev.Dockerfile`
- `d14...c6f ./docker-compose.yaml`
- `995...c1e ./docs/DEV_GUIDE.md`
- `25e...059 ./docs/EXTERNAL_BUILDERS.md`
- `00b...ba0 ./docs/IDE_INTEGRATION.md`
- `77c...a97 ./docs/LOGS.md`

- c7c...085 ./docs/OPERATOR_GETTING_STARTED.md
- 029...9e1 ./docs/THREADING.md
- fc3...50f ./docs/bootnode.md
- 2a8...ba4 ./docs/configuration.md
- 510...7a7 ./docs/resources/IBFTChart1.png
- ade...759 ./docs/resources/IBFTChart2.png
- f49...003 ./docs/resources/blox_logo.png
- c6c...53c ./docs/resources/cov-badge.svg
- 531...5c6 ./docs/resources/doppelganger_life_cycle.png
- e2e...5fa ./docs/resources/port_permissions.gif
- e61...4a6 ./docs/resources/security_permission.png
- f43...fc4 ./docs/resources/ssv_header_image.png
- 6eb...06f ./docs/specs/NETWORKING.md
- 1f8...39b ./docs/specs/README.md
- 711...518 ./doppelganger/README.md
- 1bb...dbe ./doppelganger/doppelganger.go
- bf7...919 ./doppelganger/doppelganger_test.go
- 0b3...98c ./doppelganger/mock.go
- 340...ccc ./doppelganger/noop.go
- f83...735 ./doppelganger/observability.go
- 77a...e93 ./doppelganger/state.go
- 924...cae ./e2e/.gitignore
- a84...09d ./e2e/Dockerfile
- c8b...fe1 ./e2e/beacon_proxy/attestations.go
- 78d...ad5 ./e2e/beacon_proxy/beacon_proxy.go
- e11...ad3 ./e2e/beacon_proxy/beacon_proxy_test.go
- 203...baa ./e2e/beacon_proxy/intercept/chain.go
- 5d1...5f8 ./e2e/beacon_proxy/intercept/happyinterceptor/happy.go
- 23f...e1d ./e2e/beacon_proxy/intercept/interceptor.go
- c7e...686 ./e2e/beacon_proxy/intercept/slashinginterceptor/attestations.go
- af6...4ed ./e2e/beacon_proxy/intercept/slashinginterceptor/proposals.go
- fe0...fe8 ./e2e/beacon_proxy/intercept/slashinginterceptor/slashing.go
- dbf...5af ./e2e/beacon_proxy/proposals.go
- ff8...9aa ./e2e/cmd/ssv-e2e/beacon_proxy.go
- beb...768 ./e2e/cmd/ssv-e2e/logs_catcher.go
- 8f6...6a0 ./e2e/cmd/ssv-e2e/main.go
- 719...5a8 ./e2e/cmd/ssv-e2e/share_update.go
- bf5...864 ./e2e/config/config.yaml
- c90...a3a ./e2e/config/share1.yaml
- 52a...488 ./e2e/config/share2.yaml
- 33a...f70 ./e2e/config/share3.yaml
- a21...a06 ./e2e/config/share4.yaml
- 7e8...f66 ./e2e/docker-compose.yml
- 87a...318 ./e2e/go.mod
- 77a...2d2 ./e2e/go.sum
- 694...8ae ./e2e/logs_catcher/config.go
- 5c3...0e6 ./e2e/logs_catcher/docker/docker_reader.go
- cf3...b4c ./e2e/logs_catcher/docker/restarter.go
- dcc...7f3 ./e2e/logs_catcher/logs.go
- 8bd...b9f ./e2e/logs_catcher/logs/logs.go
- 583...2e1 ./e2e/logs_catcher/logs_test.go
- 2e8...37c ./e2e/logs_catcher/matcher.go

- `5c6...a27 ./e2e/logs_catcher/matcher_bls.go`
- `1d2...f6b ./e2e/logs_catcher/parser/json.go`
- `91e...e69 ./e2e/run.sh`
- `b4f...300 ./e2e/validators.json`
- `a06...383 ./eth/contract/contract.abi`
- `cdb...179 ./eth/contract/contract.go`
- `f37...d59 ./eth/contract/generate.go`
- `df7...773 ./eth/contract/operator_public_key.abi`
- `e24...1f5 ./eth/contract/operator_public_key.go`
- `8aa...a51 ./eth/design.md`
- `d47...7fd ./eth/ethtest/cluster_liquidated_test.go`
- `92c...a10 ./eth/ethtest/cluster_reactivated_test.go`
- `019...f6f ./eth/ethtest/common_test.go`
- `ad5...05e ./eth/ethtest/eth_e2e_test.go`
- `73f...ee7 ./eth/ethtest/operator_added_test.go`
- `e90...c14 ./eth/ethtest/operator_removed_test.go`
- `4f2...32f ./eth/ethtest/set_fee_recipient_test.go`
- `b2d...7ea ./eth/ethtest/utils_test.go`
- `b12...a8d ./eth/ethtest/validator_added_test.go`
- `fd1...087 ./eth/ethtest/validator_exited_test.go`
- `a8c...3d6 ./eth/ethtest/validator_removed_test.go`
- `61f...335 ./eth/eventhandler/event_handler.go`
- `556...3d2 ./eth/eventhandler/event_handler_test.go`
- `f18...8c3 ./eth/eventhandler/handlers.go`
- `f45...834 ./eth/eventhandler/handlers_test.go`
- `2f7...e49 ./eth/eventhandler/local_events_test.go`
- `f3c...a5a ./eth/eventhandler/observability.go`
- `e44...13e ./eth/eventhandler/options.go`
- `179...3a9 ./eth/eventhandler/task.go`
- `fc6...4f3 ./eth/eventhandler/task_executor_test.go`
- `de1...907 ./eth/eventhandler/validation.go`
- `986...e18 ./eth/eventhandler/validation_test.go`
- `6bd...bb9 ./eth/eventparser/event_parser.go`
- `6dc...627 ./eth/eventparser/event_parser_test.go`
- `ffa...b75 ./eth/eventsyncer/event_syncer.go`
- `dbb...b7f ./eth/eventsyncer/event_syncer_mock.go`
- `e08...b26 ./eth/eventsyncer/event_syncer_test.go`
- `86f...83b ./eth/eventsyncer/options.go`
- `f37...67c ./eth/executionclient/config.go`
- `2db...1ea ./eth/executionclient/defaults.go`
- `9b1...32c ./eth/executionclient/execution_client.go`
- `931...5fd ./eth/executionclient/execution_client_test.go`
- `d4e...70f ./eth/executionclient/logs.go`
- `824...e9b ./eth/executionclient/logs_test.go`
- `1ce...11f ./eth/executionclient/mocks.go`
- `c43...53b ./eth/executionclient/multi_client.go`
- `7fd...457 ./eth/executionclient/multi_client_test.go`
- `8a6...4f2 ./eth/executionclient/observability.go`
- `bd2...496 ./eth/executionclient/options.go`
- `900...91d ./eth/localevents/local_events.go`
- `975...5ff ./eth/localevents/local_events_test.go`
- `234...a4b ./eth/simulator/simcontract/build/simcontract_sol_Callable.abi`

- `cc6...227 ./eth/simulator/simcontract/build/simcontract_sol_Callable.bin`
- `e2c...dd1 ./eth/simulator/simcontract/generate.go`
- `493...23e ./eth/simulator/simcontract/simcontract.go`
- `dd9...208 ./eth/simulator/simcontract/simcontract.sol`
- `987...f1b ./eth/simulator/simulator.go`
- `4df...d56 ./exporter/README.md`
- `4aa...fc0 ./exporter/api/broadcaster.go`
- `fff...d9f ./exporter/api/broadcaster_test.go`
- `be2...344 ./exporter/api/conn.go`
- `ccc...3ef ./exporter/api/decided/stream.go`
- `4d5...d9c ./exporter/api/interfaces.go`
- `068...9cb ./exporter/api/msg.go`
- `32f...b47 ./exporter/api/query_handlers.go`
- `982...437 ./exporter/api/query_handlers_test.go`
- `9f8...562 ./exporter/api/server.go`
- `563...d77 ./exporter/api/server_test.go`
- `b23...7a4 ./exporter/api/test_utils.go`
- `419...784 ./go.mod`
- `a79...cf6 ./go.sum`
- `a7a...8af ./hooks/build`
- `094...a12 ./hooks/push`
- `1b1...739 ./ibft/IBFT.md`
- `bad...cd2 ./ibft/README.md`
- `331...98a ./ibft/storage/observability.go`
- `5e2...c41 ./ibft/storage/store.go`
- `b64...1d4 ./ibft/storage/store_test.go`
- `813...181 ./ibft/storage/stores.go`
- `854...a93 ./identity/store.go`
- `00a...05b ./identity/store_test.go`
- `f39...538 ./install.sh`
- `691...a26 ./integration/qbft/tests/setup_test.go`
- `311...251 ./integration/qbft/tests/temp_testing_beacon_network.go`
- `653...606 ./logging/context.go`
- `f7a...dc6 ./logging/context_test.go`
- `657...e03 ./logging/fields/fields.go`
- `e1e...f40 ./logging/fields/stringer/stringer.go`
- `b21...630 ./logging/global.go`
- `5da...38c ./logging/mocks/zapcore.go`
- `fb2...a7c ./logging/names.go`
- `9a5...4f3 ./logging/testing.go`
- `a13...5c0 ./message/signatureverifier/mock.go`
- `ccd...ec1 ./message/signatureverifier/signature_verifier.go`
- `626...14a ./message/validation/common_checks.go`
- `6f6...80c ./message/validation/consensus_state.go`
- `d1d...5ab ./message/validation/consensus_state_test.go`
- `077...343 ./message/validation/consensus_validation.go`
- `8a7...4d5 ./message/validation/consensus_validation_test.go`
- `585...ca7 ./message/validation/const.go`
- `753...239 ./message/validation/errors.go`
- `f32...d86 ./message/validation/logger_fields.go`
- `02a...10f ./message/validation/message_counts.go`
- `426...570 ./message/validation/observability.go`

- `bec...70d ./message/validation/options.go`
- `73c...2b9 ./message/validation/partial_validation.go`
- `0ca...b5a ./message/validation/pubsub_validation.go`
- `079...02d ./message/validation/self.go`
- `cd2...649 ./message/validation/signed_ssv_message.go`
- `eaa...1a3 ./message/validation/signer_state.go`
- `3a2...058 ./message/validation/utils_test.go`
- `ed2...e25 ./message/validation/validation.go`
- `aff...5ce ./message/validation/validation_test.go`
- `7b6...cdb ./migrations/migration_0_example.go`
- `c7e...7be ./migrations/migration_1_example.go`
- `812...a5c ./migrations/migration_2_encrypt_shares.go`
- `1a6...912 ./migrations/migration_3_truncate_registry.go`
- `c6d...017 ./migrations/migration_4_configlock_add_alan_fork_to_network_name.go`
- `1db...85a ./migrations/migration_5_gob.go`
- `54e...d60 ./migrations/migration_5_share_gob_to_ssz.go`
- `202...17b ./migrations/migration_5_share_gob_to_ssz_test.go`
- `5e9...dd2 ./migrations/migration_6_model.go`
- `c58...88d ./migrations/migration_6_model_encoding.go`
- `aad...e6c ./migrations/migration_6_share_exit_epoch.go`
- `784...b26 ./migrations/migration_6_share_exit_epoch_test.go`
- `c32...254 ./migrations/migrations.go`
- `3bf...000 ./migrations/migrations_test.go`
- `abc...b42 ./monitoring/metrics/handler.go`
- `72a...d04 ./monitoring/metrics/health_check.go`
- `010...87a ./network/README.md`
- `c52...424 ./network/commons/addr_utils.go`
- `67b...595 ./network/commons/addr_utils_test.go`
- `776...71a ./network/commons/defaults.go`
- `3f9...4b4 ./network/commons/keys.go`
- `568...138 ./network/commons/keys_test.go`
- `d5e...d20 ./network/commons/subnets.go`
- `89e...9cd ./network/commons/subnets_test.go`
- `221...c08 ./network/discovery/dv5_bootnode.go`
- `18c...c6d ./network/discovery/dv5_filters.go`
- `754...658 ./network/discovery/dv5_routing.go`
- `202...78b ./network/discovery/dv5_service.go`
- `7be...8be ./network/discovery/dv5_service_test.go`
- `5a3...233 ./network/discovery/dv5_test.go`
- `1f1...22b ./network/discovery/enode.go`
- `adc...6e6 ./network/discovery/enode_test.go`
- `87e...64d ./network/discovery/forking_dv5_listener.go`
- `64f...17d ./network/discovery/forking_dv5_listener_test.go`
- `8ec...8cd ./network/discovery/iterator_test.go`
- `b97...745 ./network/discovery/kad_dht.go`
- `02c...3bf ./network/discovery/local_service.go`
- `b23...f1b ./network/discovery/logger/common.go`
- `7d6...b42 ./network/discovery/logger/groups.go`
- `3db...8f2 ./network/discovery/logger/md5_logger.go`
- `702...a6f ./network/discovery/node_record.go`
- `863...4b5 ./network/discovery/observability.go`
- `df9...d20 ./network/discovery/options.go`

- `ed0...660 ./network/discovery/service.go`
- `a42...84f ./network/discovery/service_test.go`
- `74d...7c4 ./network/discovery/shared_conn.go`
- `5dc...21c ./network/discovery/subnets.go`
- `2bc...096 ./network/discovery/subnets_test.go`
- `fe2...a59 ./network/discovery/util_test.go`
- `b23...71d ./network/network.go`
- `a8f...5b5 ./network/p2p/config.go`
- `bc7...c9f ./network/p2p/observability.go`
- `2d6...54f ./network/p2p/p2p.go`
- `8d4...177 ./network/p2p/p2p_discovery.go`
- `a56...0e3 ./network/p2p/p2p_discovery_test.go`
- `126...46c ./network/p2p/p2p_pubsub.go`
- `f16...ab8 ./network/p2p/p2p_reporter.go`
- `c4b...d4e ./network/p2p/p2p_setup.go`
- `f10...e4e ./network/p2p/p2p_test.go`
- `e12...fab ./network/p2p/p2p_validation_test.go`
- `1be...fd9 ./network/p2p/test_utils.go`
- `5a2...1a1 ./network/peers/conn_manager.go`
- `8c8...ac0 ./network/peers/connections/conn_gater.go`
- `f7d...e3b ./network/peers/connections/conn_handler.go`
- `c7f...4ef ./network/peers/connections/filters.go`
- `e63...63a ./network/peers/connections/filters_test.go`
- `081...296 ./network/peers/connections/handshaker.go`
- `4f5...ffa ./network/peers/connections/handshaker_test.go`
- `b77...980 ./network/peers/connections/helpers_test.go`
- `8b0...eb0 ./network/peers/connections/mock/mock_conn.go`
- `1b4...ef2 ./network/peers/connections/mock/mock_connection_index.go`
- `44d...05a ./network/peers/connections/mock/mock_id_service.go`
- `79c...f4f ./network/peers/connections/mock/mock_net.go`
- `b78...25c ./network/peers/connections/mock/mock_node_info_idx.go`
- `5f0...006 ./network/peers/connections/mock/mock_peerstore.go`
- `665...863 ./network/peers/connections/mock/mock_storage.go`
- `d04...a5b ./network/peers/connections/mock/mock_stream_controller.go`
- `c43...aac ./network/peers/connections/observability.go`
- `44d...138 ./network/peers/gossip_score_index_test.go`
- `abe...cc4 ./network/peers/gossipsub_score_index.go`
- `c41...693 ./network/peers/index.go`
- `a32...641 ./network/peers/peer_info.go`
- `587...45e ./network/peers/peers_index.go`
- `ddc...f2f ./network/peers/scores.go`
- `9b5...119 ./network/peers/scores_test.go`
- `2a6...052 ./network/peers/subnets.go`
- `cfc...d52 ./network/peers/subnets_test.go`
- `6e6...ca0 ./network/records/entries.go`
- `fa6...7a1 ./network/records/metadata.go`
- `42f...7ed ./network/records/metadata_test.go`
- `2b3...e3d ./network/records/node_info.go`
- `445...888 ./network/records/node_info_test.go`
- `03f...727 ./network/records/serializable.go`
- `8f9...fe5 ./network/records/subnets.go`
- `077...7c3 ./network/records/subnets_test.go`

- 7fd...caa ./network/records/test_utils.go
- 25c...838 ./network/streams/controller.go
- fb3...022 ./network/streams/controller_test.go
- 0aa...6f3 ./network/streams/observability.go
- d53...e38 ./network/streams/stream.go
- c9c...a9c ./network/streams/stream_test.go
- f24...175 ./network/testing/keys.go
- b1c...d5e ./network/testing/local.go
- 36e...ffd ./network/testing/net.go
- 501...123 ./network/topics/container.go
- 711...5b0 ./network/topics/controller.go
- 7ad...16c ./network/topics/controller_test.go
- 2d5...ddc ./network/topics/msg_id.go
- f44...169 ./network/topics/msg_validator_test.go
- b97...27d ./network/topics/observability.go
- c8b...708 ./network/topics/params/gossipsub.go
- de1...fef ./network/topics/params/helpers.go
- cb0...ce0 ./network/topics/params/message_rate.go
- 29a...1e1 ./network/topics/params/message_rate_test.go
- 826...1f5 ./network/topics/params/peer_score.go
- 2ea...3c8 ./network/topics/params/scores_test.go
- d77...60f ./network/topics/params/topic_score.go
- 77c...94d ./network/topics/pubsub.go
- a63...66f ./network/topics/scoring.go
- 3f9...e25 ./network/topics/scoring_test.go
- ae1...5e9 ./network/topics/sub_filter.go
- 1cd...1da ./network/topics/sub_filter_test.go
- 3fc...b5b ./network/topics/tracer.go
- 081...39f ./networkconfig/NEW_NETWORK.md
- 5fd...c57 ./networkconfig/beacon.go
- bc6...9ac ./networkconfig/config.go
- 0b3...91f ./networkconfig/holesky-e2e.go
- 599...d87 ./networkconfig/holesky-stage.go
- e0e...908 ./networkconfig/holesky.go
- 86c...342 ./networkconfig/hoodi-stage.go
- 94b...f8a ./networkconfig/hoodi.go
- 873...7ef ./networkconfig/local-testnet.go
- 2e3...764 ./networkconfig/mainnet.go
- fb0...e09 ./networkconfig/sepolia.go
- 20b...2ed ./networkconfig/ssv.go
- a38...4a5 ./networkconfig/test-network.go
- 1ee...1c8 ./nodeprobe/nodeprobe.go
- d90...a41 ./nodeprobe/nodeprobe_test.go
- 13b...4f1 ./observability/CONVENTIONS.md
- 55a...b85 ./observability/attributes.go
- 1c9...5a9 ./observability/config.go
- 32f...eab ./observability/metric.go
- 97d...0c9 ./observability/metric_test.go
- 8f7...ef1 ./observability/observability.go
- 274...6a6 ./observability/option.go
- 890...fab ./operator/datastore/data_store.go
- ebf...d36 ./operator/datastore/data_store_test.go

- 1be...323 ./operator/duties/attester.go
- 465...da8 ./operator/duties/attester_test.go
- 074...b90 ./operator/duties/base_handler.go
- cdf...913 ./operator/duties/base_handler_mock.go
- 015...618 ./operator/duties/committee.go
- 1ea...bb3 ./operator/duties/committee_test.go
- 0ba...ca2 ./operator/duties/dutystore/duties.go
- aa0...b95 ./operator/duties/dutystore/store.go
- 4d9...554 ./operator/duties/dutystore/sync_committee.go
- 1bb...66e ./operator/duties/dutystore/voluntary_exit.go
- 0d9...996 ./operator/duties/observability.go
- 880...ba7 ./operator/duties/proposer.go
- 8b0...e70 ./operator/duties/proposer_test.go
- f1f...dd6 ./operator/duties/scheduler.go
- 3e6...18e ./operator/duties/scheduler_mock.go
- 02a...6ea ./operator/duties/scheduler_test.go
- 2bc...1df ./operator/duties/sync_committee.go
- 0cd...036 ./operator/duties/sync_committee_test.go
- 77d...6a2 ./operator/duties/validatorregistration.go
- 226...d75 ./operator/duties/voluntary_exit.go
- 1f5...d57 ./operator/duties/voluntary_exit_test.go
- 7c6...58f ./operator/fee_recipient/controller.go
- ff3...2e0 ./operator/fee_recipient/controller_test.go
- b78...aad ./operator/fee_recipient/mocks/controller.go
- 18e...d1a ./operator/node.go
- b08...315 ./operator/slotticker/mocks/slotticker.go
- 2ff...70a ./operator/slotticker/slotticker.go
- dde...0ef ./operator/slotticker/slotticker_test.go
- 35c...fdb ./operator/slotticker/timer.go
- 165...ca2 ./operator/storage/config_lock.go
- 0eb...ec0 ./operator/storage/config_lock_test.go
- 9b9...2ad ./operator/storage/storage.go
- 807...fa9 ./operator/storage/storage_test.go
- c6c...fda ./operator/validator/controller.go
- 9eb...5ad ./operator/validator/controller_test.go
- 181...d23 ./operator/validator/metadata/mocks.go
- 0b1...e9a ./operator/validator/metadata/syncer.go
- c0b...069 ./operator/validator/metadata/syncer_test.go
- f29...314 ./operator/validator/metrics.go
- 66f...a41 ./operator/validator/mocks/controller.go
- 207...1c4 ./operator/validator/mocks/validator_map.go
- 904...eda ./operator/validator/observability.go
- 5f4...eb8 ./operator/validator/router.go
- 7ea...55a ./operator/validator/router_test.go
- 423...b2e ./operator/validator/task_executor.go
- 9b4...568 ./operator/validator/task_executor_test.go
- bdf...58d ./operator/validators/validators_map.go
- f99...d18 ./protocol/v2/blockchain/beacon/client.go
- 697...697 ./protocol/v2/blockchain/beacon/mock_client.go
- e6a...906 ./protocol/v2/blockchain/beacon/mocks/network.go
- b6f...f40 ./protocol/v2/blockchain/beacon/network.go
- fdb...6da ./protocol/v2/blockchain/beacon/network_test.go

- `909...3c5 ./protocol/v2/blockchain/beacon/validator_metadata.go`
- `1be...0b4 ./protocol/v2/blockchain/beacon/validator_metadata_test.go`
- `7b3...867 ./protocol/v2/blockchain/eth1/registry_storage.go`
- `478...c98 ./protocol/v2/message/consensus.go`
- `5a0...842 ./protocol/v2/message/consensus_test.go`
- `d34...3f3 ./protocol/v2/message/encoding.go`
- `58d...faf ./protocol/v2/message/msg.go`
- `074...7e1 ./protocol/v2/p2p/network.go`
- `59a...4d2 ./protocol/v2/qbft/config.go`
- `f12...bac ./protocol/v2/qbft/controller/controller.go`
- `e7f...2b1 ./protocol/v2/qbft/controller/controller_test.go`
- `f45...703 ./protocol/v2/qbft/controller/decided.go`
- `7c7...2d0 ./protocol/v2/qbft/controller/timer.go`
- `c93...043 ./protocol/v2/qbft/controller/types.go`
- `2ed...96f ./protocol/v2/qbft/controller/types_test.go`
- `0f5...128 ./protocol/v2/qbft/instance/commit.go`
- `48b...f9d ./protocol/v2/qbft/instance/compact.go`
- `211...b9d ./protocol/v2/qbft/instance/compact_test.go`
- `b75...110 ./protocol/v2/qbft/instance/instance.go`
- `831...75d ./protocol/v2/qbft/instance/instance_test.go`
- `2b1...38a ./protocol/v2/qbft/instance/marshalutils.go`
- `1e1...561 ./protocol/v2/qbft/instance/metrics.go`
- `0ec...e46 ./protocol/v2/qbft/instance/observability.go`
- `2ce...717 ./protocol/v2/qbft/instance/prepare.go`
- `79b...621 ./protocol/v2/qbft/instance/proposal.go`
- `c3f...3a4 ./protocol/v2/qbft/instance/round_change.go`
- `88c...4ae ./protocol/v2/qbft/instance/timeout.go`
- `5ce...d74 ./protocol/v2/qbft/round_robin_proposer.go`
- `3d6...9fc ./protocol/v2/qbft/roundtimer/mocks/timer.go`
- `098...b40 ./protocol/v2/qbft/roundtimer/testing_timer.go`
- `3d1...21c ./protocol/v2/qbft/roundtimer/timer.go`
- `cbc...8c6 ./protocol/v2/qbft/roundtimer/timer_test.go`
- `4b8...444 ./protocol/v2/qbft/spectest/controller_type.go`
- `e3f...9d2 ./protocol/v2/qbft/spectest/create_msg_type.go`
- `ac0...6d3 ./protocol/v2/qbft/spectest/msg_processing_type.go`
- `461...616 ./protocol/v2/qbft/spectest/msg_type.go`
- `a75...fb1 ./protocol/v2/qbft/spectest/qbft_mapping_test.go`
- `152...0b8 ./protocol/v2/qbft/spectest/timeout_type.go`
- `e1d...809 ./protocol/v2/qbft/storage/participant_store.go`
- `282...6d7 ./protocol/v2/qbft/testing/storage.go`
- `8cb...d1d ./protocol/v2/qbft/testing/utils.go`
- `ad2...161 ./protocol/v2/qbft/testing_utils.go`
- `777...96a ./protocol/v2/queue/exec_queue.go`
- `bba...15b ./protocol/v2/queue/worker/message_worker.go`
- `93f...aa4 ./protocol/v2/queue/worker/message_worker_test.go`
- `e39...1d8 ./protocol/v2/ssv/partial_sig_container.go`
- `315...e04 ./protocol/v2/ssv/queue/message_prioritizer.go`
- `d04...02f ./protocol/v2/ssv/queue/message_prioritizer_test.go`
- `1ee...976 ./protocol/v2/ssv/queue/messages.go`
- `7a7...e44 ./protocol/v2/ssv/queue/queue.go`
- `bb1...90b ./protocol/v2/ssv/queue/queue_test.go`
- `351...7d8 ./protocol/v2/ssv/runner/aggregator.go`

- f8b...ef1 ./protocol/v2/ssv/runner/committee.go
- f40...79f ./protocol/v2/ssv/runner/duty_runners.go
- 346...be8 ./protocol/v2/ssv/runner/measurements.go
- 6eb...b5c ./protocol/v2/ssv/runner/observability.go
- f9d...d14 ./protocol/v2/ssv/runner/proposer.go
- 6a4...1ca ./protocol/v2/ssv/runner/runner.go
- 261...fc0 ./protocol/v2/ssv/runner/runner_signatures.go
- 584...0ff ./protocol/v2/ssv/runner/runner_state.go
- ff8...483 ./protocol/v2/ssv/runner/runner_state_helpers.go
- 8d0...1ef ./protocol/v2/ssv/runner/runner_validations.go
- 551...e41 ./protocol/v2/ssv/runner/sync_committee_aggregator.go
- 99d...1fd ./protocol/v2/ssv/runner/timer.go
- d11...eb4 ./protocol/v2/ssv/runner/validator_registration.go
- 004...54a ./protocol/v2/ssv/runner/voluntary_exit.go
- 765...8ef ./protocol/v2/ssv/spectest/committee_msg_processing_type.go
- bbf...21f ./protocol/v2/ssv/spectest/debug_states.go
- 76b...a3f ./protocol/v2/ssv/spectest/msg_processing_type.go
- 8b8...99b ./protocol/v2/ssv/spectest/multi_msg_processing_type.go
- 019...c08 ./protocol/v2/ssv/spectest/multi_start_new_runner_duty_type.go
- 74d...ce6 ./protocol/v2/ssv/spectest/runner_construction_type.go
- 3ed...2a0 ./protocol/v2/ssv/spectest/ssv_mapping_test.go
- b36...06d ./protocol/v2/ssv/spectest/sync_committee_aggregator_proof_type.go
- 51a...e30 ./protocol/v2/ssv/testing/runner.go
- d06...b40 ./protocol/v2/ssv/testing/validator.go
- 7c0...8d1 ./protocol/v2/ssv/validator/committee.go
- 988...ad0 ./protocol/v2/ssv/validator/committee_guard.go
- 251...a7c ./protocol/v2/ssv/validator/committee_guard_test.go
- 1ae...976 ./protocol/v2/ssv/validator/committee_queue.go
- 9f1...804 ./protocol/v2/ssv/validator/domain_cache.go
- a92...f47 ./protocol/v2/ssv/validator/duty_executer.go
- c69...ad5 ./protocol/v2/ssv/validator/events.go
- 01f...414 ./protocol/v2/ssv/validator/msgqueue_consumer.go
- cee...0d9 ./protocol/v2/ssv/validator/msgqueue_consumer_test.go
- 5f7...fb9 ./protocol/v2/ssv/validator/non_committee_validator.go
- 197...c5a ./protocol/v2/ssv/validator/opts.go
- 58d...6fa ./protocol/v2/ssv/validator/signature_verifier.go
- 872...a0b ./protocol/v2/ssv/validator/startup.go
- 20a...cc0 ./protocol/v2/ssv/validator/timer.go
- 7a4...c31 ./protocol/v2/ssv/validator/validator.go
- c7f...561 ./protocol/v2/ssv/value_check.go
- 4ec...b01 ./protocol/v2/testing/test_utils.go
- 248...f34 ./protocol/v2/types/bls.go
- b9b...d97 ./protocol/v2/types/crypto.go
- 8cb...e8c ./protocol/v2/types/messages.go
- 0f9...96d ./protocol/v2/types/operator.go
- 17a...8ca ./protocol/v2/types/signature_benchmark_linux_test.go
- 4af...5cf ./protocol/v2/types/signature_benchmark_test.go
- de2...868 ./protocol/v2/types/ssvshare.go
- 6fc...bf0 ./protocol/v2/types/ssvshare_test.go
- ec4...ee8 ./registry/storage/mocks/operators.go
- 594...6bd ./registry/storage/mocks/validatorstore.go
- d3d...fa8 ./registry/storage/operators.go

- e15...ae6 ./registry/storage/operators_test.go
- 68a...532 ./registry/storage/recipients.go
- 5d4...74f ./registry/storage/recipients_test.go
- 608...daf ./registry/storage/shares.go
- e54...075 ./registry/storage/shares_encoding.go
- d29...9ca ./registry/storage/shares_encoding_test.go
- 58b...337 ./registry/storage/shares_test.go
- 61e...019 ./registry/storage/validatorstore.go
- 87d...627 ./registry/storage/validatorstore_test.go
- 975...ac7 ./scripts/differ/.gitignore
- 7cf...388 ./scripts/differ/README.md
- 1b1...b8c ./scripts/differ/config.example.yaml
- cd7...f00 ./scripts/differ/diff.go
- ef0...c09 ./scripts/differ/differ_test.go
- b62...8bb ./scripts/differ/go.mod
- 145...dac ./scripts/differ/go.sum
- 0f1...4f0 ./scripts/differ/main.go
- 143...021 ./scripts/differ/parser.go
- 115...d6a ./scripts/differ/transformers.go
- 260...469 ./scripts/differ/transformers_test.go
- 15a...4eb ./scripts/differ/ui/.gitignore
- 772...9be ./scripts/differ/ui/README.md
- 2ae...0a9 ./scripts/differ/ui/globals.css
- d65...ba4 ./scripts/differ/ui/package-lock.json
- 79b...651 ./scripts/differ/ui/package.json
- 4cb...5ec ./scripts/differ/ui/pages/_app.tsx
- 1e0...2d7 ./scripts/differ/ui/pages/_document.tsx
- 523...d81 ./scripts/differ/ui/pages/index.tsx
- 251...fc5 ./scripts/differ/ui/postcss.config.js
- 2e1...656 ./scripts/differ/ui/tailwind.config.js
- 18c...350 ./scripts/differ/ui/tsconfig.json
- a3e...52c ./scripts/generate_local_config.sh
- 2a7...a43 ./scripts/protogen.sh
- 4e3...a1a ./scripts/spec-alignment/.gitignore
- 1aa...741 ./scripts/spec-alignment/README.md
- ed6...840 ./scripts/spec-alignment/differ.config.yaml
- b35...f53 ./scripts/spec-alignment/differ.sh
- e73...61f ./ssvsigner/DESIGN.md
- b32...915 ./ssvsigner/Dockerfile
- b7a...b64 ./ssvsigner/README.md
- 407...ec8 ./ssvsigner/client.go
- 328...fad ./ssvsigner/client_test.go
- 351...abf ./ssvsigner/cmd/purge-keys/purge-keys.go
- 6af...fc4 ./ssvsigner/cmd/ssv-signer/ssv-signer.go
- 911...8e2 ./ssvsigner/cmd/ssv-signer/ssv-signer_test.go
- 0e1...2db ./ssvsigner/ekm/doc.go
- 7a3...52f ./ssvsigner/ekm/key_manager.go
- 751...207 ./ssvsigner/ekm/local_key_manager.go
- 926...2ab ./ssvsigner/ekm/local_key_manager_test.go
- a7f...9bc ./ssvsigner/ekm/mock.go
- e52...a4c ./ssvsigner/ekm/remote_key_manager.go
- 74e...5c0 ./ssvsigner/ekm/remote_key_manager_test.go

- `821...d71 ./ssvsigner/ekm/signer_storage.go`
- `125...052 ./ssvsigner/ekm/signer_storage_test.go`
- `096...ee6 ./ssvsigner/ekm/slashing_protector.go`
- `8c6...6e3 ./ssvsigner/ekm/slashing_protector_test.go`
- `754...c3e ./ssvsigner/ekm/testing.go`
- `e44...427 ./ssvsigner/go.mod`
- `74e...898 ./ssvsigner/go.sum`
- `c71...ee0 ./ssvsigner/internal/mocks/mocks.go`
- `920...e9d ./ssvsigner/keys/jemalloc_check.go`
- `e22...e14 ./ssvsigner/keys/keys.go`
- `f3c...9f9 ./ssvsigner/keys/keys_test.go`
- `159...340 ./ssvsigner/keys/rsa.go`
- `270...d38 ./ssvsigner/keys/rsa_benchmark_test.go`
- `0c4...e01 ./ssvsigner/keys/rsa_linux.go`
- `b6d...cbf ./ssvsigner/keys/rsa_linux_test.go`
- `296...01c ./ssvsigner/keys/rsaencryption/rsa_encryption.go`
- `41f...b67 ./ssvsigner/keys/rsaencryption/rsa_encryption_test.go`
- `b57...408 ./ssvsigner/keys/rsatesting/rsatesting.go`
- `33c...5b2 ./ssvsigner/keystore/file.go`
- `bfd...fea ./ssvsigner/keystore/file_test.go`
- `f92...56d ./ssvsigner/observability.go`
- `542...66c ./ssvsigner/server.go`
- `ef9...e98 ./ssvsigner/server_test.go`
- `a87...a86 ./ssvsigner/tls/tls.go`
- `ef4...d7c ./ssvsigner/tls/tls_test.go`
- `0bf...e2a ./ssvsigner/types.go`
- `b44...0f5 ./ssvsigner/web3signer/interfaces.go`
- `82b...588 ./ssvsigner/web3signer/options.go`
- `995...e41 ./ssvsigner/web3signer/options_test.go`
- `d78...d9f ./ssvsigner/web3signer/types.go`
- `34e...302 ./ssvsigner/web3signer/types_test.go`
- `283...799 ./ssvsigner/web3signer/web3signer.go`
- `e62...579 ./ssvsigner/web3signer/web3signer_test.go`
- `d58...573 ./storage/basedb/storage.go`
- `e3d...59c ./storage/kv/badger.go`
- `483...570 ./storage/kv/badger_test.go`
- `4e2...41b ./storage/kv/gc.go`
- `26a...41f ./storage/kv/gc_test.go`
- `6e3...9c8 ./storage/kv/logger.go`
- `b26...e3f ./storage/kv/logger_test.go`
- `279...5d9 ./storage/kv/txn.go`
- `e60...df9 ./storage/kv/txn_test.go`
- `73c...baa ./tests.Dockerfile`
- `199...4db ./tool.mod`
- `214...bf4 ./tool.sum`
- `f1b...d06 ./utils/async/interval.go`
- `6d1...02e ./utils/async/interval_test.go`
- `0f7...08d ./utils/blskeygen/blskeygen.go`
- `a91...313 ./utils/blskeygen/blskeygen_test.go`
- `7ae...c2e ./utils/boot_node/enr_fork_id.go`
- `c6c...7ea ./utils/boot_node/enr_fork_id_encoding.go`
- `465...2c8 ./utils/boot_node/node.go`

- `3a1...609 ./utils/casts/casts.go`
- `8f1...83f ./utils/cliflag/cliflag.go`
- `0c3...3a7 ./utils/commons/build_data.go`
- `bba...de7 ./utils/format/domain_type.go`
- `a10...545 ./utils/format/format_test.go`
- `d25...d20 ./utils/format/identifier.go`
- `400...522 ./utils/format/operator_id.go`
- `c70...87e ./utils/format/regexp_pool.go`
- `a4d...4a1 ./utils/format/regexp_pool_test.go`
- `e2c...28e ./utils/hashmap/hashmap.go`
- `5d5...c8d ./utils/hashmap/hashmap_test.go`
- `246...3e8 ./utils/keys.go`
- `e43...1be ./utils/tasks/exec_interval.go`
- `b95...46f ./utils/tasks/exec_interval_test.go`
- `27e...3d8 ./utils/tasks/exec_queue.go`
- `646...947 ./utils/tasks/exec_queue_test.go`
- `e33...74e ./utils/tasks/exec_timeout.go`
- `994...438 ./utils/tasks/exec_timeout_test.go`
- `b75...50d ./utils/tasks/retry.go`
- `337...bdb ./utils/tasks/retry_test.go`
- `bb3...f14 ./utils/tasks/stopper.go`
- `ad3...f6b ./utils/testutils.go`
- `485...7b8 ./utils/threadsafe/bool.go`
- `a52...ad5 ./utils/threadsafe/bytes.go`
- `5b1...be6 ./utils/threadsafe/int32.go`
- `c09...b5c ./utils/threadsafe/int64.go`
- `8dd...1cf ./utils/threadsafe/uint64.go`
- `b12...f98 ./utils/threshold/reconstruct.go`
- `786...bc6 ./utils/threshold/threshold.go`
- `957...ef1 ./utils/threshold/threshold_test.go`
- `48b...4fd ./utils/ttl/map.go`
- `06d...187 ./utils/ttl/map_test.go`

Changelog

- 2025-05-19 - Initial report
- 2025-07-04 - Fix Review

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp’s mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp’s team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over $200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp’s collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication or other making available of the report to you by Quantstamp.

Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and may not be represented as such. No third party is entitled to rely on the report in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or any open source or third-party software, code, libraries, materials, or information to, called by, referenced by or accessible through the report, its content, or any related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.