Quantstamp Contract Security Certificate

February 15th 2020 — Quantstamp Verified

RSK Client Library

This security review was prepared by Quantstamp, the protocol for securing smart contracts.

Executive Summary

Type	Client Library
Reviewers	Alex Murashkin, Senior Software Engineer Sung-Shine Lee, Research Engineer
Timeline	2019-11-20 through 2020-02-14
Languages	Javascript
Methods	Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	In-repo documentation In-repo README
Source Code	

Repository	Commit
rsk3.js	5a10571
rsk3.js	88bed97
rsk3.js	5e18df0
rsk3.js	c797025

Total Issues	**14**	(10 Resolved)
High Risk Issues	**1**	(1 Resolved)
Medium Risk Issues	**2**	(1 Resolved)
Low Risk Issues	**1**	(1 Resolved)
Informational Risk Issues	**6**	(5 Resolved)
Undetermined Risk Issues	**4**	(2 Resolved)

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
Low Risk	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 post an immediate risk, but is relevant to security best practices or Defence in Depth.
Undetermined	The impact of the issue is uncertain.
Unresolved	Acknowledged the existence of the risk, and decided to accept it without engaging in special efforts to control it.
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).
Resolved	Adjusted program implementation, requirements or constraints to eliminate the risk.

Changelog

- 2019-11-28 - Initial report (commit c797025)
- 2020-01-29 - Audited diff (commit 5e18df0)
- 2020-02-11 - Audited diff (commit 88bed97)
- 2020-02-14 - Audited diff (commit 5a10571)

Overall Assessment

For the most part, the code seems to be very similar to web3.js of version 2.x, which implies any vulnerabilities found/detected in Web3.js could potentially be relevant to the RSK library. The major issues that are highlighted in the report are: the clone-and-own approach used in the project, lack of input validation across the board, support for operations that are not secure (however, not clearly documented as such). Many of these issues are present in the upstream library `web3.js` (`2.x`), however, we believe that it is important to highlight them as they could be exploited if consumers of the library are not well-informed about these.
Other example issues include version unlocking and inconsistencies when it comes to managing dependency and runtime versions, issues with documentation.
Updates: As of commit `5a10571`, all issues were addressed (fixed or acknowledged).

Summary of Findings

ID	Description	Severity	Status
QSP-1	Unsecure use of local storage	High	Resolved
QSP-2	Missing input validation in multiple locations	Medium	Resolved
QSP-3	Clone-and-Own	Medium	Acknowledged
QSP-4	Secure connections to RSK nodes are not enforced	Low	Resolved
QSP-5	Inconsistency in Node.js version support	Informational	Resolved
QSP-6	Unlocked dependency versions	Informational	Resolved
QSP-7	Unimplemented `TODO`s and incomplete documentation	Informational	Acknowledged
QSP-8	Some demo and README examples do not work out-of-the-box	Informational	Resolved
QSP-9	Inconsistent use of BN and BigNumber	Informational	Resolved
QSP-10	Misleading and undocumented behaviour of account address conversion methods	Informational	Resolved
QSP-11	Bugs in external dependencies affecting the current library	Undetermined	Acknowledged
QSP-12	Use of a dev-dependency with a known vulnerability	Undetermined	Resolved
QSP-13	Exposing methods that could lead to building potentially unsecure websites	Undetermined	Resolved
QSP-14	Potential bug in logic	Undetermined	Acknowledged

Quantstamp Review Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. Pay closer attention to the code related to account management and manipulation.

Methodology

The Quantstamp reviewing process follows a routine series of steps:

1. Code review that includes the following
    i. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the project.
    ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    iii. 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:
    i. 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.
    ii. Static analysis.
3. Best practices review, which is a review of the code to improve efficiency, effectiveness, clarify, 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 project.

Toolset

The notes below outline the setup and steps performed in the process of this security review.

Setup

Tool Setup:

- Npm-Audit Scanner
- Retire.js
- Node.js
- Sonarqube Scanner
- Jest

Steps taken to run the tools:

1. Ran `npm audit` for every package and the root project
2. `npm install -g retire && retire`
3. `nvm install 8.11.3 && nvm use 8.11.3`
4. `npm install -g sonarqube-scanner && sonar-scanner`
5. Installed Jest: `npm install -g jest`

Assessment

Findings

QSP-1 Unsecure use of local storage

Severity: *High Risk*

Status: Resolved

File(s) affected: `rsk3-account/src/models/wallet.js`

Description: The `load(...)` and `save(...)` methods store accounts in `window.localStorage`. It is against the common secure practice to store any sensitive data in local storage.
In addition, the `password` parameter is not validated against best security practices, therefore, it makes the functionality even less secure.

Exploit Scenario: A website using the rsk.js library inside the browser could be XSS-attacked in multiple ways. We list two as examples.
Scenario 1.

1. A website uses rsk.js on the browser-side
2. The front-end stores the accounts by calling the method `save(...)`
3. The website is subject to an XSS attack which allows a third-party app to execute Javascript
4. The third-party code reads the entire local storage and sends it to the attacker's website
5. The attacker's website uses a brute-forcing technique on the encrypted payloads. Considering that the password field is not validated in terms of length, the attack has chances of succeeding.

Scenario 2.

1. A website uses rsk.js on the browser-side
2. The front-end stores the accounts by calling the method `save(...)`
3. The front-end loads the wallet using the method `load(...)` into memory
4. The website is XSS-attacked which makes it easy to read the private key from the browser's memory

Recommendation:

1. Removing this functionality completely. If unsafe functionality is available to users, it is only a matter of time before it gets exploited.
2. If removal of functionality is not an option:

   • ensure that the encryption password is secure enough to protect the accidentally leaked encrypted data from being brute-force-decrypted

   • instruct the library to only decrypt the account's private key when it is absolutely necessary for signing. Add a method such that it is difficult to extract and take the private key out of this, such as, require account unlock before the signing and make account lock automated right after the signing.

Update: Resolved in commit `5e18df0`.

QSP-2 Missing input validation in multiple locations

Severity: *Medium Risk*

Status: Resolved

File(s) affected: `rsk3, rsk3-abi, rsk3-account, rsk3-contract, rsk3-personal, rsk3-net (packages)`

Description: Validating inputs from users is one of the key requirements and security practices. We have identified multiple locations across various packages where inputs are not validated properly. Examples include, but are not limited to, the following:

1. All parameters passed to constructors should be validated, e.g., cannot be `null`/`undefined` or of an unexpected type
2. `rsk3-account/src/models/account.js`, `L106`: `privateKey` is assumed to be a string, but should be validated to be such
3. `rsk3-account/src/models/account.js`, `L128`: `password`, `options.salt`, `options.iv` are not validated. Should enforce a minimum length restriction
4. `rsk3-account/src/models/account.js`, `L151`: `options.n` is not validated. If the comment `8192; // 2048 4096 8192 16384` denotes the list of allowed values, they need to be checked
5. `rsk3-account/src/models/account.js`, `L184`: if `options.uuid` is not input-validated, it could be a low value that does not provide sufficient entropy
6. `rsk3-account/src/models/account.js`, `L220`: if `v3Keystore` is not an object, needs to check that it's a string
7. `rsk3-account/src/models/account.js`, `L228`: `json.crypto` may be undefined
8. `rsk3-account/src/models/wallet.js`, `L43`: need to make sure `entropy` is a good source for entropy
9. `rsk3-account/src/models/wallet.js`, `L78`: need a check that account is defined
10. `rsk3-account/src/accounts.js`, `L97`: `hashMessage(data)` is missing validating `data` if it's not a hex strict message
11. `rsk3-account/src/accounts.js`, `L190`: `sign(...)` - input data validation is missing
12. `rsk3-personal/src/personal.js`: input validation is missing in setter methods
13. `rsk-utils/src/index.js`: many parameters passed to several methods are lacking validation
14. `rsk-utils/src/index.js`, `L150`: `utils.keccak256`: the method does not handle integer inputs
15. `rsk-utils/src/index.js`: `L741`: length validation is missing for `signature`
16. `rsk-utils/src/index.js`: `L759`: missing validation for `btcPrivateKey` and `decodedKey`
17. `rsk-utils/src/index.js`: `L769`: missing validation for `btcPrivateKey`
18. `rsk-utils/src/index.js`: `L780`: missing validation for `btcPrivateKey`
19. `rsk-utils/src/index.js`: `L790`: `btcNet` being either `MAIN_NET` or `TEST_NET` is not enforced
20. `rsk-utils/src/index.js`: `L58`: `functionName` could be a malformed object
21. `rsk3/src/signers/tranasactionSigner.js`, `L34`: missing input validation on `privateKey`
22. `rsk3/src/signers/tranasactionSigner.js`, `L42`: need to validate `this.parameters`
23. `rsk3-abi/src/abiCoder.js`: most of the functions assume that parameters passed in are either a `String` or an `Object`. However, integers are neither considered as `String` nor `Object`, and some methods may be unprepared to handle that
24. `rsk3-abi/src/abiCoder.js`, `L27`: passes an `Object` into `jsonInterfaceMethodToString`: unclear what happens if a malformed object (or an object with a format not expected by the function) is passed in

Exploit Scenario: Exploit scenarios can vary and are not always easy-to-identify. Usually, missed input validation is a basis for more sophisticated exploit scenarios. For example, lack of input validation on the password may make it easy for an attacker to brute-force data stolen from the website's internal storage using an XSS-attack.

Recommendation:

1. Always check that inputs are defined
2. For password-related inputs, check that they are above a certain length and document the restriction
3. Since Javascript is not a typed language, always make sure input types are validated before use of the variable
4. The document explicitly the parameters that require additional validation at the application level

Update: While some functions have now input validation implemented as per commit `5e18df0`, validation is limited to variable type-checking, while password and signature length enforcement is still lacking. In addition, not all function inputs are being validated.
Update: More input validation was added as per commit `88bed97`, however, from the original list, the following instances remain:

1. `rsk3-account/src/models/account.js`, `L128`: `options.salt` and `options.iv` do not seem to have length restrictions
2. `rsk3-account/src/models/account.js`, `L197`: `options.uuid` is not validated for type/length
3. `rsk3-account/src/models/wallet.js`, `L44`: need to make sure `entropy` is a good source for entropy
4. `rsk-utils/src/index.js`: many parameters passed to several methods are lacking validation
5. `rsk-utils/src/index.js`: `L742`: length validation is missing for `signature`.

Update: As of commit `5a10571`, the issues outlined above were resolved.

QSP-3 Clone-and-Own

Severity: *Medium Risk*

Status: Acknowledged

File(s) affected: `rsk3, rsk3-abi, rsk3-account, rsk3-contract, rsk3-personal, rsk3-net (packages)`

Description: The clone-and-own approach involves copying and adjusting open source code at one's own discretion. From the development perspective, it is initially beneficial as it reduces the amount of effort. However, from the security perspective, it involves some risks as the code may not follow the best practices, may contain a security vulnerability, or may include intentionally or unintentionally modified upstream libraries. Rather than the clone-and-own approach, a good industry practice is to include dependencies as libraries. This eliminates the clone-and-own risks yet allows for following best practices, such as, using libraries.
We identified the following instances of undocumented clone-and-own instances (the source appears to be web3.js 2.x https://github.com/ethereum/web3.js/tree/2.x, the commit `58a4432cb19626254d10054e2a800ef178221a8f`):

1. `rsk3` package is similar to `web3-eth`
2. `rsk3-abi` is similar to `web3-abi`
3. `rsk3-account` is similar to `web3-account`:
    - minor modifications include the removal of the original copyright and renaming of `web3js_wallet` to `rsk3js_wallet`
    - `rsk3-account/src/scrypt.js` looks similar to the functionality provided by the NPM module `@web3-js/scrypt-shim`.
4. `rsk3-personal` is similar to `web3-personal`
5. `rsk3-contract` is similar to `web3-contract`
6. `rsk3-net` is similar to `web3-net`, however, network ids and names differ

`rsk3-utils` appears to have the most unique content.

Recommendation:

1. Favour using external libraries over cloning-and-owning
2. If clone-and-own is used, make the clone-and-own cases more explicit and known to users and keep the original copyrights.

Update: Acknowledged in commit `5e18df0`.

QSP-4 Secure connections to RSK nodes are not enforced

Severity: *Low Risk*

Status: Resolved

Description: Currently, like Ethereum's web3.js, the library allows for connections via http and unsecure websocket connections.

Exploit Scenario:

1. A user connects to an RSK node via http or ws (rather than `https://` or `wss://`)
2. A user tries unlocking the account via `rsk3.personal.unlockAccount(...)` or make a new account via `rsk3.personal.new(...)`

Recommendation:

1. Enforce secure connections at the library level
2. Show a strong warning that the connection is insecure, if a user purposefully tries to bypass this restriction

Update: addressed in commit `5e18df0`: a warning is to be shown if the connection string is prefixed with an unsecure protocol.

QSP-5 Inconsistency in Node.js version support

Severity: *Informational*

Status: Resolved

Description:

1. The `engines` section of `rsk3-personal` reads as follows (in commit `c797025`):

   ```
   "engines": {
     "node": ">=8.0.0"
   },
   ```

   while the remaining projects' ones read as follows:

   ```
   "engines": {
     "node": ">=8.11.3"
   },
   ```

   This inconsistency needs to be fixed.

2. Node.js version support needs to be aligned with Node.js version support of upstream dependencies, such as, web3.js 2.x. We noticed that the version 9 was removed from test automation. While running the current test suite does not show any issue, we recommend monitoring this dependency and take actions if any incompatibility is detected

Recommendation:

1. Documenting the supported version numbers as well as environments (e.g., in-browser and server-side)
2. Adding multiple Node.js versions to automated testing (e.g., Travis-CI) to make sure all versions are tested continuously
3. Making the "engine" section consistent across all `package.json` files in the repository
4. Monitoring `node` version support in the upstream dependencies.

Update: the version has been set to `8.6.0` as of commit `5e18df0`.

QSP-6 Unlocked dependency versions

Severity: *Informational*

Status: Resolved

Description: In multiple package.json files, dependency versions are not locked (or, fixed). If a user install the dependency at a later time when an upstream dependency is updated, they may get a different version of dependency that was not tested properly or contain security vulnerabilities.

Recommendation:

1. Fix the package versions
2. Commit the `package-lock.json` into the source control

Update: Resolved in commit `5e18df0`.

QSP-7 Unimplemented `TODO`s and incomplete documentation

Severity: *Informational*

Status: Acknowledged

Description: At glance, we identified 39 instances of `TODO` in various files. The `TODO`s are also present in the `README.md`files of each package. Having `TODO`s in the code without a reference to concrete features or names create an impression of an incomplete code, which is reflected negatively on security

Recommendation: Address the `TODO`s or reference concrete tasks/features/people/timelines when the `TODO`s are to be addressed.
Update: the team has acknowledged the finding, however, preferred not to fix it due to time constraints.

QSP-8 Some demo and README examples do not work out-of-the-box

Severity: *Informational*

Status: Resolved

Description:

1. The `demo/rsktest` script does not work out-pf-the-box: `L4` need to be replaced with `const Rsk3=require('rsk3');` (commit `c797025`)
2. `L98`: `repl` is undefined (commit `c797025`)
3. `README.md` in the root: the `npm run clean` script is not implemented (commit `c797025`)
4. `README.md` in the root: running tests also requires running `npm run bootstrap` initially (commit `c797025`)

Recommendation: Confirming the working state of the demo scripts so that library consumers have an easier time to get started.
Update: Fixed as of commit `5e18df0`.

QSP-9 Inconsistent use of BN and BigNumber

Severity: *Informational*

Status: Resolved

File(s) affected: `rsk3-utils/src/index.js`

Description: `BN` and `BigNumber` are two different libraries that are well-known to the Ethereum community. The comments and functions here seem to treat them as the same library.
Comments such as `rbtcUnit.js`, `L29`mention `BigNumber` but `BN` is returned. `rsk3-utils/src/index.js`, `L133`: `isBigNumber` should be removed, as it is checking for `BN` and thus is redundant to `isBN` at `L120`. Alternatively, `L137` should be `BigNumber`.
Consequently, there are some if/else statements in functions where different actions are performed when `isBigNumber` is `true` and `isBN` is `true`.
There are at least redundancies of logic if not errors, as some actions should be performed under `isBigNumber`, but the content is always `BN`.

Recommendation: We suggest to make it clear that `rsk3.js` is only using `BN` and make the necessary adjustments to code and comments.
Update: Fixed as of commit `5e18df0`.

QSP-10 Misleading and undocumented behaviour of account address conversion methods

Severity: *Informational*

Status: Resolved

File(s) affected: `rsk3-utils/src/index.js`

Description: `rsk3-utils/src/index.js`, `L790`: The parameter description of `getBtcPrivateKey` is misleading, `rskAddress` should be `rskPrivateKey`.
There are four different kinds of private key formats (ECDSA Private Key, HD Wallet Keys, Base58 Wallet Import format, and Mini private key format) in the BTC standard, the one that is expected in the functions is the `Wallet Import` format. This should be explicitly stated in the documentation and checked if the input matches `WIF` format.
Also, since the RSK Private key is just one of the format in BTC, it might make sense to implement conversions from other BTC formats into the RSK format.

Recommendation:

1. Fixing the parameter name
2. Documenting the methods and their expected behaviour
3. Consider implementing conversions from other BTC formats into the RSK format.

Update: addressed in commit `5e18df0`.

QSP-11 Bugs in external dependencies affecting the current library

Severity: *Undetermined*

Status: Acknowledged

Description: Upstream dependencies, such as, `web3.js`, get constant updates and bug fixes from the web3.js page

1. It is important to keep up with these updates, considering that the project references the version `2.0.0-alpha.1`: the version that was not battle-tested as much as older versions, and the naming `alpha` does not necessarily imply production readiness.
2. `web3.js` 2.x bugs are present in rsk3.js, are reproducible and there are likely more issues to be uncovered in the future. Examples include, but are not limited, to the following:

Account related issues

- https://github.com/ethereum/web3.js/issues/2725: reproducible, highlight: the `create` function in RSK3.js gives different accounts even when providing the same input.
- https://github.com/ethereum/web3.js/issues/2189: issues with `unlockAccount`

Confirmed that are reproducible

- https://github.com/ethereum/web3.js/issues/2846
- https://github.com/ethereum/web3.js/issues/2848

Other known issues: https://github.com/ethereum/web3.js/issues?utf8=%E2%9C%93&q=is%3Aopen+label%3A2.x+label%3Abug

Exploit Scenario: A bug or a vulnerability discovered in web3.js could also be exploited in RSK client library.

Recommendation:

1. Consider addressing the issues shown above (as well as identify others that may be relevant) or document the behaviour and limitations
2. Use an automated dependency monitoring module or service (for example, Greenkeeper), to stay up-to-date with dependency versions
3. Regularly check the GitHub issues page(such as, the web3.js one) for upstream dependencies

Update: the team has acknowledged the finding, however, stated that fixing the said bugs is outside of the project's scope.

QSP-12 Use of a dev-dependency with a known vulnerability

Severity: *Undetermined*

Status: Resolved

File(s) affected: `package.json`

Description: `eslint-utils` of the version `1.4.0` required by `eslint` has a known critical-severity vulnerability as described in the advisory https://www.npmjs.com/advisories/1118

Exploit Scenario: As written in the advisory:

```
Versions of eslint-utils >=1.2.0 or <1.4.1 are vulnerable to Arbitrary Code Execution. The getStaticValue does not properly sanitize user input allowing attackers to supply malicious input that executes arbitrary code during the linting process. The getStringIfConstant and getPropertyName functions are not affected.
```

An exact attack vector on this specific project remains unclear, however, we highly recommend addressing it.

Recommendation:

1. Upgrading `eslint` to the latest version
2. Locking `eslint` dependency version

Update: Resolved in commit `5e18df0`.

QSP-13 Exposing methods that could lead to building potentially unsecure websites

Severity: *Undetermined*

Status: Resolved

File(s) affected: `rsk-utils/src/index.js`

Description: The methods `getBtcPrivateKey` and `privateKeyToRskFormat` are not documented in the `README`s, however, these methods should be documented with a note that a high degree of caution should be used. Not only they accept a private key as an input and also produce a private key as an output. The consumer, if not using it correctly, is risking leaking of both private keys.

Exploit Scenario:

1. While library consumers are outside the scope of the current security review, websites, such as, https://utils.rsk.co/ could serve as an example. We regard this website as having a high security risk. If the website gets attacked, hacked, or does not follow the best security practices, an attacker may get access to private keys on both blockchains: the RSK and the Bitcoin. Examples for both input and output keys include, but are not limited to: logging a private key, storing a private key in the internal storage or memory, not alerting users about the risks.
2. In addition, if the conversion algorithm of the private keys is exposed to the public, an attacker may try to trick a user into revealing their RSK private key and simultaneously get access to the Bitcoin wallet, and vice-versa

Recommendation:

1. Clearly document security considerations
2. Consider changing the design in such a way that private keys are never required to be input anywhere
3. Consider not exposing the code or the algorithm to the public.

Update: in commit `5e18df0`, the methods were documented, however, the security risks were not clearly highlighted. We strongly recommend documenting security risks related to passing private keys around.
Update: in commit `88bed97`, the methods described in this issue were completely removed from the library.

QSP-14 Potential bug in logic

Severity: *Undetermined*

Status: Acknowledged

File(s) affected: `rsk3-abi/src/abiCoder.js`

Description: `rsk3-abi/src/abiCoder.js`, `L91`: Replacement of `0x` seem to make assumptions about location and quantity of the occurrences

Recommendation: It is recommended to double-check the intended behaviour and make sure replacement does not cause unintended side effects.
Update: the team has acknowledged the finding and mentioned that the same logic is present in the upstream library, and therefore, it is not a concern.

Automated Analyses

Npm-Audit Scanner

`npm audit` did not find any vulnerabilities for the contents of the packages folder.

Retire.js

Retire.js did not find any vulnerabilities for the contents of the `packages` folder. However, it was able identify the use of a vulnerable library at the root level:

```
node_modules/eslint-utils/package.json
  ✓ eslint-utils 1.4.0
```

`eslint-utils 1.4.0 has known vulnerabilities: severity: high; summary: Arbitrary Code Execution; https://www.npmjs.com/advisories/1118`

The details of this vulnerability could be found on the page.

Update: as of commit `5e18df0`, the `eslint-utils` version has been updated, and running `retire` returns no vulnerabilities.

Sonarqube Scanner

Sonarqube detected 0 vulnerabilities but suggested 20 security hotspots to examine. Among the 20 hotspots, two were unique:

1. Security hotspot 1:
    - Location: `packages/rsk3-utils/src/index.js , L857`
    - Code: `const typesize = /^\D+(\d+).*$/.exec(type);`
    - Sonarqube message: `Make sure that using a regular expression is safe here.`
    - Conclusion: according to the tool `safe-regex`, the regex is actually safe
2. Security hotspot 2:
    - Location: `packages/rsk3-account/src/crypto/scrypt.js`, `L39`
    - Code: `return crypto.scryptSync(key, salt, dkLength, {N, r, p});`
    - Sonarqube message: `Make sure that hashing data is safe here.`
    - Conclusion: the `scrypt` function at `L37` does not validate the `key` and `salt` parameters, and `account.js` (`rsk3-account/src/models/account.js`) does not need impose length restrictions. If the lengths are too small, this could lead to predictable outputs. We noted lack of input validation as the major issue in the report.

Sonarqube also flagged four areas as potentially buggy, but they were all deemed as false-positives upon manual inspection.

Update: as of commit `5e18df0`, the number of security hotspots was reduced to four, and all were deemed as false-positives also.

Test Results

Test Suite Results

Ran tests for the following Node.JS versions: `8.11.3`, `9.11.2`, `10.17.0`, `11.15.0`, and `12.13.1`. All tests passed for each of the versions.

For `12.13.1`, the build step for the `scrypt` module was failing, however, this may not be an issue because the rsk3.js library is supposed to use the built-in `scrypt` module.

```
----rsk3.js/packages/rsk3:

 PASS  tests/methods/ethSignTransactionMethod.test.js
 PASS  tests/signers/transactionSigner.test.js
 PASS  tests/methods/getTransactionFromBlockMethod.test.js
 PASS  tests/factories/subscriptionsFactory.test.js
 PASS  tests/factories/methodFactory.test.js
 PASS  tests/methods/rskSignMethod.test.js
 PASS  tests/methods/getUncleMethod.test.js
 PASS  tests/methods/getBlockTransactionCountMethod.test.js
 PASS  tests/methods/getBlockUncleCountMethod.test.js
 PASS  tests/methods/rskGetAccountsMethod.test.js
 PASS  tests/methods/getBlockMethod.test.js
 PASS  tests/rsk3.test.js

Test Suites: 12 passed, 12 total
Tests:       54 passed, 54 total
Snapshots:   0 total
Time:        5.587s

----rsk3.js/packages/rsk3-abi:
 PASS  tests/abi.test.js
  AbiCoderTest
    ✓ constructor check (5ms)
    ✓ calls encodeFunctionSignature with a string as parameter (1ms)
    ✓ calls encodeFunctionSignature with a object as parameter (1ms)
    ✓ calls encodeEventSignature with a object as parameter (1ms)
    ✓ calls encodeEventSignature with a string as parameter (1ms)
    ✓ calls encodeParameters (1ms)
    ✓ calls encodeParameter (1ms)
    ✓ calls encodeFunctionCall and returns the expected string (1ms)
    ✓ calls decodeParameters and returns the expected object (1ms)
    ✓ calls decodeParameters and throws an error (11ms)
    ✓ calls decodeParameter and returns the expected object (1ms)
    ✓ calls decodeLog and returns the expected object (2ms)

Test Suites: 1 passed, 1 total
Tests:       12 passed, 12 total
Snapshots:   0 total
Time:        2.07s
Ran all test suites.

----rsk3.js/packages/rsk3-account:

 PASS  tests/factories/methodFactory.test.js
 PASS  tests/models/account.test.js
 PASS  tests/models/wallet.test.js
 PASS  tests/account.test.js

Test Suites: 4 passed, 4 total
Tests:       51 passed, 51 total
Snapshots:   0 total
```

```
Time:        3.096s

----rsk3.js/packages/rsk3-contract:

 PASS  tests/mappers/eventOptionsMapper.test.js
 PASS  tests/proxies/eventSubscriptionsProxy.test.js
 PASS  tests/proxies/methodsProxy.test.js
 PASS  tests/mappers/abiMapper.test.js
 PASS  tests/contract.test.js
 PASS  tests/factories/methodFactory.test.js
 PASS  tests/methods/allPastEventLogsMethod.test.js
 PASS  tests/factories/contractModuleFactory.test.js
 PASS  tests/subscriptions/eventLogSubscription.test.js
 PASS  tests/methods/pastEventLogsMethod.test.js
 PASS  tests/subscriptions/allEventsLogSubscription.test.js
 PASS  tests/validators/methodOptionsValidator.test.js
 PASS  tests/methods/callContractMethod.test.js
 PASS  tests/methods/contractDeployMethod.test.js
 PASS  tests/factories/eventSubscriptionFactory.test.js
 PASS  tests/mappers/methodOptionsMapper.test.js
 PASS  tests/encoders/methodEncoder.test.js
 PASS  tests/decoders/allEventsLogDecoder.test.js
 PASS  tests/mappers/allEventsOptionsMapper.test.js
 PASS  tests/methods/sendContractMethod.test.js
 PASS  tests/decoders/eventLogDecoder.test.js
 PASS  tests/encoders/eventFilterEncoder.test.js
 PASS  tests/models/abiModel.test.js
 PASS  tests/models/abiItemModel.test.js
 PASS  tests/encoders/allEventsFilterEncoder.test.js

Test Suites: 25 passed, 25 total
Tests:       150 passed, 150 total
Snapshots:   0 total
Time:        4.807s
Ran all test suites.

----rsk3.js/packages/rsk3-net:

 PASS  tests/net.test.js
  NetworkTest
    ✓ constructor check (6ms)
    ✓ calls getNetworkType and resolves to the network name "private (3ms)
    ✓ calls getNetworkType and resolves to the network name "main (1ms)
    ✓ calls getNetworkType and resolves to the network name "morden (2ms)
    ✓ calls getNetworkType and resolves to the network name "ropsten (1ms)
    ✓ calls getNetworkType and rejects the promise (1ms)

Test Suites: 1 passed, 1 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        1.783s, estimated 2s
Ran all test suites.


----rsk3.js/packages/rsk3-personal:

 PASS  tests/methodFactory.test.js
 PASS  tests/personal.test.js

Test Suites: 2 passed, 2 total
Tests:       11 passed, 11 total
Snapshots:   0 total
Time:        2.068s, estimated 3s
Ran all test suites.

----rsk3.js/packages/rsk3-utils:

 PASS  tests/rbtcUnit.test.js
 PASS  tests/solidity_sha3.test.js
 PASS  tests/utils.test.js

Test Suites: 3 passed, 3 total
Tests:       121 passed, 121 total
Snapshots:   0 total
Time:        1.916s, estimated 2s
Ran all test suites.
```

Code Coverage

```
Statements   : 90.4% ( 1196/1323 )
Branches     : 79.92% ( 621/777 )
Functions    : 92.54% ( 273/295 )
Lines        : 90.59% ( 1184/1307 )
```

The code, overall, is sufficiently covered with tests. Some statements, branches, or functions remain uncovered, however, the overall coverage is around 80% or higher, which is a reasonable threshold.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
rsk3-utils	82.08	76.4	84.78	82.52	
--index.js	81.34	76.29	83.33	81.84	… 43,949,954,964
--rbtcUnit.js	**86.15**	**77.08**	**100**	**86.15**	**… 46,152,160,178**
rsk3-abi	92.42	80.43	100	92.42	

--abiCoder.js	92.42	80.43	100	92.42	… 82,234,238,242
rsk3-net	100	71.43	100	100	
--network.js	100	71.43	100	100	49,55
rsk3-account	94.35	81.98	90.32	94.25	
--src	94.55	78.57	91.67	94.55	
----accounts.js	94.55	78.57	91.67	94.55	55,56,59
--**src/**factories	100	100	100	100	
----methodFactory.js	100	100	100	100	
--**src/**models	94.17	83.13	88.89	94.02	
----account.js	92.86	82.09	75	92.86	83,95,108,226,236
----wallet.js	96	87.5	100	95.74	22,80
rsk3-personal	86.84	58.33	100	86.84	
--src	86.11	58.33	100	86.11	
----personal.js	86.11	58.33	100	86.11	72,99,126,153,180
--**src/**factories	100	100	100	100	
----methodFactory.js	100	100	100	100	
rsk3-contract	97.95	87.89	99.21	97.95	
--src	97.62	61.54	91.67	97.62	
----abstractContract.js	97.62	61.54	91.67	97.62	109
--**src/**decoders	100	100	100	100	
----allEventsLogDecoder.js	100	100	100	100	
----eventLogDecoder.js	100	100	100	100	
--**src/**encoders	96.67	83.33	100	96.67	
----allEventsFilterEncoder.js	100	100	100	100	
----eventFilterEncoder.js	92.86	75	100	92.86	43
----methodEncoder.js	100	87.5	100	100	27
--**src/**factories	100	100	100	100	
----contractModuleFactory.js	100	100	100	100	
----eventSubscriptionFactory.js	100	100	100	100	
----methodFactory.js	100	100	100	100	
--**src/**mappers	98.84	87.1	100	98.84	
----abiMapper.js	100	95.65	100	100	35
----allEventsOptionsMapper.js	94.44	71.43	100	94.44	25
----eventOptionsMapper.js	100	81.25	100	100	27,33,50
----methodOptionsMapper.js	100	100	100	100	
--**src/**methods	96.43	88.89	100	96.43	
----allPastEventLogsMethod.js	100	100	100	100	
----callContractMethod.js	100	100	100	100	
----contractDeployMethod.js	80	75	100	80	46,47
----pastEventLogsMethod.js	100	100	100	100	
----sendContractMethod.js	100	87.5	100	100	44
--**src/**models	100	91.67	100	100	
----abiItemModel.js	100	100	100	100	
----abiModel.js	100	83.33	100	100	80
--**src/**proxies	95.51	86.84	100	95.51	
----eventSubscriptionsProxy.js	**92.31**	**93.75**	**100**	**92.31**	**75,106**
----methodsProxy.js	96.83	81.82	100	96.83	58,60

--**src/**subscriptions	100	100	100	100	
----allEventsLogSubscription.js	100	100	100	100	
----eventLogSubscription.js	100	100	100	100	
--**src/**validators	100	100	100	100	
----methodOptionsValidator.js	100	100	100	100	
rsk3	86.34	76.39	81.67	86.25	
--src	77.11	62.16	72.22	77.11	
----index.js	77.11	62.16	72.22	77.11	… 83,388,390,391
--**src/**factories	100	100	100	100	
----methodFactory.js	100	100	100	100	
----subscriptionsFactory.js	100	100	100	100	
--**src/**methods	98	90.91	100	97.96	
----getBlockMethod.js	100	100	100	100	
----getBlockTransactionCountMethod.js	100	100	100	100	
----getBlockUncleCountMethod.js	100	100	100	100	
----getTransactionFromBlockMethod.js	100	100	100	100	
----getUncleMethod.js	100	100	100	100	
----rskGetAccountsMethod.js	87.5	50	100	85.71	33
----rskSignMethod.js	100	100	100	100	
----rskSignTransactionMethod.js	100	75	100	100	38
--src/signers	**88.89**	**87.5**	**66.67**	**88.89**	
----transactionSigner.js	88.89	87.5	66.67	88.89	24,41

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.

Contracts

```
4de4e1ca93656eb9be4b7b927bfb539c5155277bc469527ede9adde2408a89c2 ./packages/rsk3-utils/jest.config.js

f87bf58b929c42d6f31b6cef137d8964612efbe3965459287558c6bc677a2d4f ./packages/rsk3-utils/rollup.config.js

b059f527811ad1f211044925de52901af96c1f3b19be022643d98a99497e585f ./packages/rsk3-utils/tests/rbtcUnit.test.js

3b713fbb12e806d32ef34cf52a11aff45b535f53d6fae12053bbf80a86134f99 ./packages/rsk3-
utils/tests/soliditySha3.test.js

0ab9031889027d92c51e0e1f639311d5c74f75f96f85cde59ce8aa88481e41a7 ./packages/rsk3-utils/tests/utils.test.js

232448c13ca3addd8f0a2152daf3ca51e84b70b8dc7a8bfbf76e608e08c999d0 ./packages/rsk3-utils/src/index.js

ca49af456674043d62ad67a06cc743470bb0eadeb50d8d50e1811f0585b93059 ./packages/rsk3-utils/src/rbtcUnit.js

79d7d865feb5b339bd16f4de7888b59154dcef1df8bbd141c7bcb9feaeaf19f9 ./packages/rsk3-utils/dist/rsk3-utils.cjs.js

533b954df28fcff764808e631755a6be85625cb02dcb45e138525578e32cf66c ./packages/rsk3-utils/dist/rsk3-utils.esm.js

aee7a0f49d038b8859b82319f8ad47b2152e9586516b2c36150088b03f4a8c54 ./packages/rsk3-utils/dist/rsk3-utils.umd.js

4de4e1ca93656eb9be4b7b927bfb539c5155277bc469527ede9adde2408a89c2 ./packages/rsk3-personal/jest.config.js

a72b4bb1935c5f3df66f164f739be954383cb3e2821eee1c38e27b15d34a1a8e ./packages/rsk3-personal/rollup.config.js

9ef29429fe400ffb5bd4c4376c4a7c76db2e7b44f2f683f601fb78fb1224c735 ./packages/rsk3-
personal/tests/methodFactory.test.js

29923b985bc1f0bbd1084f1a30ffafb82ac31f9887a296c69247061313d8a527 ./packages/rsk3-personal/tests/personal.test.js

e63ce84b4f0c32ba6d4b59704a2960e05ef9ab6613c61ec08577c73d0d8c7746 ./packages/rsk3-personal/src/index.js

9abc59a8d5ec95ae77e46a92de9614ccbc1c5e7ca117276887ffe5db87877d97 ./packages/rsk3-personal/src/personal.js

9626cf9c8b0d91bc005f3229daef3191e2a4597157f14d7dc659e0be111e6ae4 ./packages/rsk3-
personal/src/factories/methodFactory.js

4de4e1ca93656eb9be4b7b927bfb539c5155277bc469527ede9adde2408a89c2 ./packages/rsk3-net/jest.config.js
```

```
5c20502283c0aa564be8873949286e2662aa17058e56d81944457c88267ac1fe ./packages/rsk3-net/rollup.config.js

a1ddbb4d8b432019ca84dafa8bdb15f7076f64fc570d8ac94b372ecef1a6228a ./packages/rsk3-net/tests/net.test.js

92402a2cd8b0c4cc5609c049b4f732332af1cb8872b4adb534cd2105b6e5d46e ./packages/rsk3-net/src/index.js

5d7297109c3420b7726dbf51a213a448e587ddfab2c74597951da475958fb440 ./packages/rsk3-net/src/network.js

9e6a9bfa3bef34baac8b50adcc80d3b41adb22c9335137cc3e65724e1bdb11fb ./packages/rsk3-
net/src/factories/methodFactory.js

4de4e1ca93656eb9be4b7b927bfb539c5155277bc469527ede9adde2408a89c2 ./packages/rsk3-contract/jest.config.js

7e28ce9d38f8ff0477805b02d3d36eb31372bd8c38d81fc8d9f88728c262327e ./packages/rsk3-contract/rollup.config.js

b7495d92858ce8783e778eb481a1873c47f0ca68402bc8b8949e85a27aebd910 ./packages/rsk3-contract/tests/contract.test.js

26cb6b8c6961ca048e97b3735c818bcc7ed0cde34360f37bff528d1f59860333 ./packages/rsk3-
contract/tests/validators/methodOptionsValidator.test.js

1cf818e1a5e94e99ad39fdd1ff33f718afcabed4e8406cce5875e635559c752f ./packages/rsk3-
contract/tests/subscriptions/allEventsLogSubscription.test.js

ea844a209540c7ab067b502383e8941aca1d2a411dc6818fc89dc42c5dc2f2c3 ./packages/rsk3-
contract/tests/subscriptions/eventLogSubscription.test.js

c84926253f8797efe7f5fdbd5fd3437c9cea0356b982c77fb9855107e85bd334 ./packages/rsk3-
contract/tests/proxies/eventSubscriptionsProxy.test.js

247e470236d4da272172ef77d6a8aaf05e5c2927537091b41fb2617070f0ac23 ./packages/rsk3-
contract/tests/proxies/methodsProxy.test.js

d71ba1f6ec60c4bcd56e04484314c12d69da4d33f0ec20fc563f01c63ba2ddb3 ./packages/rsk3-
contract/tests/models/abiItemModel.test.js

578a00bb4973e0d6d2258faaaafddc9ab0bd6a6260ff2daf2cba90c525cf7949 ./packages/rsk3-
contract/tests/models/abiModel.test.js

62aea5dd5d4d4c9d437f19c825a3bcbed25e7aa6aa1bdee58413cebefbae6e77 ./packages/rsk3-
contract/tests/methods/allPastEventLogsMethod.test.js

ddd2d624b95ba4961a916a75d15969534a1acc7d1972ab4164c8ef5da5da6362 ./packages/rsk3-
contract/tests/methods/callContractMethod.test.js

c93bb25192b0db9efbdaea6767b212034d8e9691cdce62bf3a19d29402a91b51 ./packages/rsk3-
contract/tests/methods/contractDeployMethod.test.js

6b8459a286c6899a6ca9f15e579227e3f0dcbdf0b6c69f274e9cc9d5a82d0b41 ./packages/rsk3-
contract/tests/methods/pastEventLogsMethod.test.js

0e3d18e641798db496623a192784ed9009c7c6b0053f29ef8c0f7ebacff883c9 ./packages/rsk3-
contract/tests/methods/sendContractMethod.test.js

22cfd34032c240bc886f64aae40a99f4dfe69cc6ac6cdec43f9c037aba839f99 ./packages/rsk3-
contract/tests/mappers/abiMapper.test.js

e2c88d4c2ae7f7044d19d8261c95a988d0c165ef49620e2639b165187ac60f8f ./packages/rsk3-
contract/tests/mappers/allEventsOptionsMapper.test.js

964efad8ef208eb609a9efdc34dc4c1afefe71a260b47441c0bf14a70079a670 ./packages/rsk3-
contract/tests/mappers/eventOptionsMapper.test.js

431bc1298c939b5f47c4a52bdc853d12a0f0ab7f00073fadfb19fd85442023c5 ./packages/rsk3-
contract/tests/mappers/methodOptionsMapper.test.js

f042d449f6cdc33dd786947fd136bc5a644512846053f5c03ab5dcb3b18fba5c ./packages/rsk3-
contract/tests/factories/contractModuleFactory.test.js

d0327879ad96adf5331b0d67396d595cdfcf40c8206757ea38acb916a424d334 ./packages/rsk3-
contract/tests/factories/eventSubscriptionFactory.test.js

80eb7e609cdfe358696751068bba7bf706a0998a3a87adea802737a7f06c0c13 ./packages/rsk3-
contract/tests/factories/methodFactory.test.js

15e055ee97483cdc617865bfa6f82262ccda4a07206801e0cdd33952622132c8 ./packages/rsk3-
contract/tests/encoders/allEventsFilterEncoder.test.js

6786a1b3193296f4030e66880987bd91c0d74a881f8682c0f668be2df6a50b81 ./packages/rsk3-
contract/tests/encoders/eventFilterEncoder.test.js

e11793352ef315035e865941a38ac3eacc235c12d1afd2f4b936a7b1cac4124b ./packages/rsk3-
contract/tests/encoders/methodEncoder.test.js

93a0f080b3b4467c895a5cac0db2c6efb1f1cf8d1e3e1828faca677b49c23936 ./packages/rsk3-
contract/tests/decoders/allEventsLogDecoder.test.js

ba09ca4b6f39bc8e4298c659f3376e8242eb66a1f6590a4baad69b9afe9336d3 ./packages/rsk3-
contract/tests/decoders/eventLogDecoder.test.js

6b9c36384a94e55a00bbdb72251461ce52e43b33081fabb1da6772527749b0cb ./packages/rsk3-
contract/src/abstractContract.js

2a530bdf0ebcadea02e7f8ef69a2fbd78f1e1a7a4c8538d3f91539a41aed190c ./packages/rsk3-contract/src/index.js

25d9d4d73972487ee677fa855f1f7ae0e1546016c22961b59cc97ad9289b8b05 ./packages/rsk3-
contract/src/validators/methodOptionsValidator.js

32509a67306cd144ec0264519d4ea937a63b04bd0ba7a28ae9a41ac91663efaf ./packages/rsk3-
contract/src/subscriptions/allEventsLogSubscription.js
```

```
d1933d61719f02a207c64f874ab845c362378c916e463fb984c41715379c498f ./packages/rsk3-
contract/src/subscriptions/eventLogSubscription.js

969cdb7cc1fd4c54e070bc76c9d6473c7b7553196f8d107644b485797147b7c5 ./packages/rsk3-
contract/src/proxies/eventSubscriptionsProxy.js

c36378ff6d216c8e3f56725c3483236f4e16658c31b207452d6aadf4697dbfa3 ./packages/rsk3-
contract/src/proxies/methodsProxy.js

c87237d9a29f926f810de8c79ad02032a96e117ca95e784add64bbe9abe59efa ./packages/rsk3-
contract/src/models/abiItemModel.js

b6d4dbef28ed94e448e9e440853a04d54ff4a23893c7b10362d24925cdbac711 ./packages/rsk3-contract/src/models/abiModel.js

8bf24ebd3642a5033af0da20567d72a9259ccffb099290164e3c2fddb5bcae7d ./packages/rsk3-
contract/src/methods/allPastEventLogsMethod.js

63927486c0c6de6c3e46fab7c9795eddf7d81eb34ae8c68f6cb20288111d45e3 ./packages/rsk3-
contract/src/methods/callContractMethod.js

e80634becc00b0edb44eb2b95c378010429ad3d64b70b650aa6814636e2e5e0f ./packages/rsk3-
contract/src/methods/contractDeployMethod.js

f4e793cd55c0c8dbf9b46b794b3a3f79849afd5f8d2b7f13c22e251af2c98051 ./packages/rsk3-
contract/src/methods/pastEventLogsMethod.js

29b3b456472eb75e7c9b2b58e31d4a3e52d45758e827e88d697b4d6958c3df43 ./packages/rsk3-
contract/src/methods/sendContractMethod.js

a56f9ee208eb93c2439de6a381067135903b4fdbac42044ce67eddd5d869c79c ./packages/rsk3-
contract/src/mappers/abiMapper.js

774bf1f5a11e88e460f18eafb30c4c1a7e804e9de09548efa8209aa3e81e8fd4 ./packages/rsk3-
contract/src/mappers/allEventsOptionsMapper.js

5448de1d9b25a967cd658737992d3cb758f5a8144108661f43a1feb6cf970735 ./packages/rsk3-
contract/src/mappers/eventOptionsMapper.js

b45924635432aeb3b49165ecf971464a76ecb27d9a2a06d719fa24329ae580fd ./packages/rsk3-
contract/src/mappers/methodOptionsMapper.js

228faedca8f3420d3e6b8f41edf23b71107c4ed420de50f92b8585db523da7a0 ./packages/rsk3-
contract/src/factories/contractModuleFactory.js

667c8fb53b0e58b17779250f827c0085d902d5125899048dc09f1193114ee656 ./packages/rsk3-
contract/src/factories/eventSubscriptionFactory.js

6db4887ba939aa3cf7a8de5a14e4266f5675cec99b6b933f50363a415e313b2d ./packages/rsk3-
contract/src/factories/methodFactory.js

7d793ebd02175b741241200e8a2bc81de0533f9079ab62442172d6896af5f7ba ./packages/rsk3-
contract/src/encoders/allEventsFilterEncoder.js

0728a8c35ab0d8c5a2eee96b0e5b80f72b0cab9fd7d5a474a676c9ed13fbaf55 ./packages/rsk3-
contract/src/encoders/eventFilterEncoder.js

72912bab5c830bcb1e8a02907a1353826c9e27beb07b73da94d0fea538c8de61 ./packages/rsk3-
contract/src/encoders/methodEncoder.js

210a5e456eb60e3f90301fc16f0ec284ec3b79dcfc395ffdcf352c521ff1dadd ./packages/rsk3-
contract/src/decoders/allEventsLogDecoder.js

637e9b49b1b4a8e0c478323b2834754c03f53c7d0e535ac059b86f3bfb9496ae ./packages/rsk3-
contract/src/decoders/eventLogDecoder.js

4de4e1ca93656eb9be4b7b927bfb539c5155277bc469527ede9adde2408a89c2 ./packages/rsk3-account/jest.config.js

d05fa24dfe285b41aaa829117c475fd279587992c3cf981daa7c0fe75b044307 ./packages/rsk3-account/rollup.config.js

a56b7122eb65074f259d1aa466f555a50a20e0d21fc2d03719181878327ddf68 ./packages/rsk3-account/tests/account.test.js

d855beeb34516461ad680b2f90c070ef41970579e844cfc953c23ba76e1af9a7 ./packages/rsk3-
account/tests/models/account.test.js

5a49817074ff5b70624759f024ec531ffd9fad5d6d577dff58d7c76bac7b665b ./packages/rsk3-
account/tests/models/wallet.test.js

1913f5a8de083c688abd6a2df0f4058eaa32f5da147d9966fcd4da7643c01e8e ./packages/rsk3-
account/tests/factories/methodFactory.test.js

b34afc2ea5146785c983bce18cacdbfa0f044e500363b0bbf450e375bb1825ed ./packages/rsk3-
account/tests/__mocks__/transactionSigner.js

5697efd9dc071a800aa18857e9293d44f0f7d0c2b79eb3106d46500c630a0d27 ./packages/rsk3-account/src/accounts.js

1c29408a3edbeda4e3555db2c92b05a02bb168bf928f7bf6c5818a9003939960 ./packages/rsk3-account/src/index.js

214b3b8e6c9cb4063d33b872598dbd4349d2065b6f6274ab9ed7ae09a83c62a5 ./packages/rsk3-account/src/models/account.js

76049ae1a5c70d3c0d5033f67b66f627bc8271c73e9c046790083b78a4e65d04 ./packages/rsk3-account/src/models/wallet.js

f9c2124f21efd49ee217d5bcf169acbc9b9a4cf15f0dd63d21c55def89699600 ./packages/rsk3-
account/src/factories/methodFactory.js

4de4e1ca93656eb9be4b7b927bfb539c5155277bc469527ede9adde2408a89c2 ./packages/rsk3-abi/jest.config.js

99e7ead9ee1b5b228e0b4226300919f2c68d9f693eb862983cfa6bdaafb936f3 ./packages/rsk3-abi/rollup.config.js

1651662e854d74f0a7d016fd0eb8e6a84f59010ab3f4a75078763cf992e7f6ad ./packages/rsk3-abi/tests/abi.test.js
```

```
4632f05f6a56350788db3466ac03a8eb62a09b346642b7d8fa8c9c585d0f935b ./packages/rsk3-abi/src/abiCoder.js

f000b83a0fb411064aa5648b8529035a40b3902d70f7440bc97201e6f9c2b4aa ./packages/rsk3-abi/src/index.js

4de4e1ca93656eb9be4b7b927bfb539c5155277bc469527ede9adde2408a89c2 ./packages/rsk3/jest.config.js

fbd5840eb86b257f2b013f07e99ff411287edb66a4a6dcadeab9a5992e486efc ./packages/rsk3/rollup.config.js

a551b9a9464f2cd2173c9a4684ce95c4824becb913adda10d059edd071fea8bb ./packages/rsk3/tests/rsk3.test.js

c000745f8592c453027f09618a781704fd9df0392ddba079e6a43c3747dba44c
./packages/rsk3/tests/signers/transactionSigner.test.js

af16658c03e8febef9d03b73999d6ddc0f9af08a9d6097782c0d66beb858f95b
./packages/rsk3/tests/methods/ethSignTransactionMethod.test.js

de7e0dd4179f3b101d2a9c5f487461790c449a64031e99d6c1d9301f678fb045
./packages/rsk3/tests/methods/getBlockMethod.test.js

e7a32f359f97b8c175f995ae7539ca78c0246558de4420ed918cc9496d79598a
./packages/rsk3/tests/methods/getBlockTransactionCountMethod.test.js

dc693fc096cc14ff88057d0d98decffd628aedcc9ac0b397ff2c3c339b24acd7
./packages/rsk3/tests/methods/getBlockUncleCountMethod.test.js

742e8ecbc4d45b8dff853b1bdd335f2b2e22a0e8a54512140f6a8fe4cb7d281b
./packages/rsk3/tests/methods/getTransactionFromBlockMethod.test.js

60cb2363d2167ab89980a67005359394909c727b9db0e6ebfd3f7b5cfac8e40c
./packages/rsk3/tests/methods/getUncleMethod.test.js

c5bb779316b2550150aae82722c8436328181887b00c505e00a1bc49b162ceba
./packages/rsk3/tests/methods/rskGetAccountsMethod.test.js

46bfe9f6551f08638aea6cc42e1e2283ef1e4da5162e998e07b71cc421780421
./packages/rsk3/tests/methods/rskSignMethod.test.js

d411ec4daa70f723dc282476c1a30087c18f7c0a0d81b6bffac3cf3219f0f101
./packages/rsk3/tests/factories/methodFactory.test.js

9cc17f1c8c1129953f5356d3f368133d95872c9a3c0a465d87527a075a235f75
./packages/rsk3/tests/factories/subscriptionsFactory.test.js

1f05b8edd296a6c983145f62fc17d164c0b63d81379e8e52b87951689499ba07 ./packages/rsk3/src/index.js

346aa2c6fd4b756347c285c5339819b4a0c04da9d768551b8665689d257087e2
./packages/rsk3/src/signers/transactionSigner.js

1b82c3ef683f436bc67f98017eafcacfb289094f7a1220bc46ba6745a1765e68 ./packages/rsk3/src/methods/getBlockMethod.js

48a533724d2fa7707f8ff50789a27a581aa3923b8742783d8fdbd28745331609
./packages/rsk3/src/methods/getBlockTransactionCountMethod.js

c0f03513d3147b92e76250fd1ce55529470d6ab861d9166dbf40af793caf5819
./packages/rsk3/src/methods/getBlockUncleCountMethod.js

673399951e6a72c4dcd2d8054164faa6df2d3177c3b95e02f258f86c35574185
./packages/rsk3/src/methods/getTransactionFromBlockMethod.js

3c101ff6eca3efb0550e0de427f9f6e219656529ea3d6b262f1095ac2fc32d56 ./packages/rsk3/src/methods/getUncleMethod.js

10a7d94a173394663ca921e35d54aa54de8a05d99ecd1f79c3c1ec672d6424dc
./packages/rsk3/src/methods/rskGetAccountsMethod.js

db54b967c48d357cd6b2dce75eb2015d19af33263a7ba80626851979ec4f987a ./packages/rsk3/src/methods/rskSignMethod.js

bc51f8481a829eca2781860a3f083e6a3b7f4b165fd2a93dde7d8e9554788c88
./packages/rsk3/src/methods/rskSignTransactionMethod.js

8f18eb8833e1110bb7b67a4830082f8a498173c16b6f39e9143132df3e9c07c2 ./packages/rsk3/src/factories/methodFactory.js

591fb81c82ad2dda20233721ecd192f6f252bef01e73307a77c12075fb2791bf
./packages/rsk3/src/factories/subscriptionsFactory.js
```

About Quantstamp

Quantstamp is a Y Combinator-backed company that helps to secure smart contracts at scale using computer-aided reasoning tools, with a mission to help boost adoption of this exponentially growing technology.

Quantstamp's team boasts decades of combined experience in formal verification, static analysis, and software verification. Collectively, our individuals have over 500 Google scholar citations and numerous published papers. In its mission to proliferate development and adoption of blockchain applications, Quantstamp is also developing a new protocol for smart contract verification to help smart contract developers and projects worldwide to perform cost-effective smart contract security reviews.

To date, Quantstamp has helped to secure hundreds of millions of dollars of transaction value in smart contracts and has assisted dozens of blockchain projects globally with its white glove security reviewing services. As an evangelist of the blockchain ecosystem, Quantstamp assists core infrastructure projects and leading community initiatives such as the Ethereum Community Fund to expedite the adoption of blockchain technology.

Finally, Quantstamp's dedication to research and development in the form of collaborations with leading academic institutions such as National University of Singapore and MIT (Massachusetts Institute of Technology) reflects Quantstamp's commitment to enable world-class smart contract innovation.