Audit process rationale

In order to ensure that our upcoming migration complies with the best security practices, Golem Factory has conducted three separate audit processes:

- an audit of the new Golem token smart contract (the new) Golem Network Token (GLM) and the associated migration proxy contract (the contract that facilitates the actual migration process) conducted by Trail of Bits, based on commit `1fb991c87b2ddc0f0c76585e77e948de4cabeade` of the `golemfactory/gnt2` repository,
- an audit of the new Golem token smart contract (the new) Golem Network Token (GLM), the migration proxy contract and the penetration test of the dedicated migration app performed by CertiK, based on commit `922728b63db7664a4a61051ae28fee506b95992f` of the `golemfactory/gnt2` repository,
- an internal audit and QA of the migration app

It is worth noting that the third contract involved in this migration (the original GNT contract) has been audited before: https://github.com/golemfactory/golem-contracts/blob/master/docs/TrailOfBitsContractsAudit.pdf

None of the audits and tests revealed any serious security issues that could impact the users migrating from the GNT token to the new token, GLM or to the users of the new token (GLM) later on.

None of the reported issues - either singularly or in combination - were deemed sufficient to request the update of the smart contracts or of the migration app's code and restart the audit process.

Golem MultiSig

Note: Whenever the Golem MultiSig account is mentioned in the document, it's referring to the Ethereum contract that is controlled by the individuals responsible for the original Golem Factory crowdfunding effort, which at the same time holds governance over the original GNT token. The Ethereum address of the contract is: 0x7da82c7ab4771ff031b66538d2fb9b0b047f6cf9.

Following are the details discussed in each audit.

Trail of Bits audit

The following issues were identified by the audit conducted by the Trail of Bits:

Permit is likely to be the target of phishing campaigns

https://github.com/golemfactory/gnt2/issues/140

Reported Severity: High

We acknowledge the importance of this issue and are taking serious measures to mitigate it.

We have already incorporated the provided advice into our documentation, social media campaign for the migration. We have two people trained to be able to look after the migration to avoid phishing.

Unfortunately, we cannot whitelist specific users (as Golem Factory) as per the nature of the migration. Migration should be opt-in and we cannot intervene unless there is a critical situation with the contract that would require us to stop the migration altogether. However, all safety measures are already either in place, planned or documented.

As part of this effort, and in the past, we have published a series of blogposts to help our users defend against phishing attacks:

- How To Protect Yourself From Scams
- Data breaches: how they affect people and what can we do to fight them
- GNT to ERC20 migration FAQs
- Making sense of the golem migration (will be published alongside the migration itself)

Additionally, we will double down on user support for the first weeks of the migration processes, have prepared video tutorials, written tutorials, and the UI has enough explanations and notices to make sure users can migrate safely.

Lack of on-chain minter verification allows for a backdoored deployment

https://github.com/golemfactory/gnt2/issues/139

Reported Severity: High

Again, we acknowledge the importance of this issue and appropriate measures have either already been undertaken or will be undertaken when needed.

The setting of the migration agent on the current GNT token can only be performed by the Golem Multisig account. Before any such transaction is signed, all parties involved will take extreme precautions to ascertain that the new contract is configured precisely as outlined in our deployment procedure:

https://github.com/golemfactory/gnt2/blob/docs/docs/deployment.md

It is important to note that, in the event of a severe issue, our migration process can be temporarily stopped and as a next step, redirected to a new target through the aforementioned proxy. While it's stopped, the users will not be able to migrate. We do not expect needing to use this function, but it has been contemplated and implemented.

Once sufficient time has passed after the commencement of the migration and provided no critical issues were identified during that time, we will revoke any ownership of these smart contracts so the process cannot be modified anymore.

Lack of chainID validation allows re-using signature across forks

https://github.com/golemfactory/gnt2/issues/138

Reported Severity: Medium

Generally, the `chain_id` _is_ used in the constructor so the replays are not possible between different networks (e.g. testnet vs mainnet).

As for any replays across mainnet forks - while we acknowledge that those are possible, we don't see that as an actual threat. What it would at most allow an attacker to do is to replay a permit for exactly the same contract that the owner of a given private key already deemed trustworthy enough to permit on a different fork.

There's no potential for a bait-and-switch type attack where an attacker could gain a permission to one contract and then substitute that contract for another on a different fork.

Front running on target update

https://github.com/golemfactory/gnt2/issues/136

Reported Severity: Medium

The only party able to set the migration agent and thus able to perform this kind of an attack is the Golem Multisig.

It's definitely contrary to the best interest of the owners of that multisig account to perform any actions to hurt the token and its holders.

Permit does not allow partial allowances

https://github.com/golemfactory/gnt2/issues/141

Reported Severity: Informational

By a design choice, `permit` will never be used that way.

We'll clearly recommend against issuing permits to individual private keys and thus permits will only be given to contracts. The contracts that Golem Factory GmbH will recommend issuing `permit` for, will be always selected by the development team after extensive due diligence.

migrateFrom ignores return value by target.mint(_from,_value)

https://github.com/golemfactory/gnt2/issues/137

Reported Severity: Informational

As we don’t intend to update the target implementation by this point, this issue is only of concern from a code purity perspective.

As mentioned in the original issue “*This issue was classified as informational as the current `target` implementation reverts in case of failure.*”

Race condition involving “permit” nonces

https://github.com/golemfactory/gnt2/issues/135

Reported Severity: Informational

The reported severity and the description of the issue clearly state this type of attack could at most cause nuisance to a party that would like to have `permit` executed by a relay.

The requests to call permit will be short-lived, i.e. the issuer will always wait until the timeout has passed before trying to submit another permit request.

Thus, by the time the attacker could perform the attack, the original request would fail anyway and the issuer will just repeat the request with another relayer.

Lack of "setMigration" and "migrate" functions invites phishing attacks

https://github.com/golemfactory/gnt2/issues/134

Reported Severity: Informational

This is a design choice aiming at limiting the governance within the new Golem token’s contract.

Code Quality issues

https://github.com/golemfactory/gnt2/issues/142

The commented-on details are minor and wouldn't in themselves warrant re-opening the audit process.
Plus, we wanted to stay compatible with DAI unless there are very good reasons not to.

Manticore results

https://github.com/golemfactory/gnt2/issues/133

There were no actual issues identified by `Manticore`.

CertiK smart contract audit

Two issues of "DISCUSSION" severity have been identified:

The batching sidecar functionality could be implemented directly in the new GNT

Reported Severity: Discussion

It was our design choice during the requirements phase of the new Golem token to limit the scope of the new token contract to the absolute minimum. Thus, what the contract's implementation encompasses is just ERC-20 compliance and an ability to extend token's functionality via side contracts when needed.

To facilitate such extension, the new Golem token contract includes the `permit` and approve methods that are used to execute code from other contracts in the context of the new Golem token contract. The main difference being the ability to grant such a permission in a way that doesn't require the token holder to possess ETH tokens as well (permit is executed by a relay that pays the ETH cost of the transaction).

Usage of such a side contract requires an explicit permission (a signed Ethereum transaction) of each owner of an account that wishes to utilize the specific extension.

The owner of GNTMigrationAgent has power over the migration process.

Reported Severity: Discussion

Obviously, Golem Multisig account has a limited control over the migration process as discussed previously. Such control is needed to enforce contingency measures in case when, despite our extensive audit process, a critical flaw is identified in the migration that would introduce a risk to the new Golem token holders.

As mentioned in the original report: "*We don't see any easy fix to this, implying that this is necessary to facilitate the migration at least in the short-term.*"

Certik Migration App audit

Three issues have been identified by the migration app penetration test:

Clickjacking

Reported Severity: Low

As the app is designed to be strictly client-side, there's no way clickjacking could be used to affect any background state.

Effectively, to perform an attack, the victim would have to be lured to visit a modified site. In such a case though, the attacker could just as easily replace the whole site with a one that would perform malicious functions instead of facilitating the migration.

Thus, this particular issue has the same solution as preventing any other phishing attacks against the holders of the Golem token and/or Golem application users.

Bad transaction record causes application to freeze

Reported Severity: Informational

To leverage this shortcoming, a hypothetical attacker would need to be able to attack the user's browser. In that case, the potential harm is much greater than just causing a fixable freeze condition in the migration app.

Plus, it suffices to remove the browser's local storage for our migration app's website address to address the issue for an affected account.

As mentioned in the report: "*In theory, it will be very hard for an attacker to modify the response from "api.infura.io" and the chance for a corrupted transaction appears in the LocalStorage is very low.*"

Reconstruct source code with source map

Reported Severity: Informational

The allegation for this issue is that a potential attacker could gain insight into possible attack vectors by analyzing the front-end source code and that this risk should be mitigated by making it harder to access the app’s actual source code.

Given that the migration app’s code is accessible to the general public as part of the published repository alongside the new Golem token contract, we don’t feel it makes sense to obfuscate the code that way.

Internal QA

Token tests

Areas of interest

The following areas of interest were researched / tested:

1. Unit tests of smart contracts prepared alongside the contracts themselves
2. OpenZeppelin's tests of smart contracts
3. Executing the deployment, migration, and emergency stop procedures
4. Attaching functionality via permitted smart contracts (Batching)

Unit tests of Smart Contracts

The team responsible for developing the smart contracts has included the unit tests within the repository holding the new Golem Network Token contracts and the migration app.

We have prepared the list of test scenarios that should be included in the testing and compared it with existing unit test suite to identify any missing tests.

The tests present in the suite include:

- GNTMigrationAgent tests
- The new Golem Network Token (GLM)
- Various integration tests

Findings

Based on the aforementioned process, unit tests that verify the following scenarios have been added:

- migration should be possible to be stopped and restarted
- owner can not migrate
- only owner can set target
- old token cannot set target

`transferFrom` modifications do not break original OpenZeppelin unit tests.

Migration tool tests

The following issues were found and have subsequently been fixed:

- Application shows blank screen when metamask - or other web3 wallet provider - is not available.
- The tool should not make it possible to send a failing transaction when migration is on hold (target = 0x0)
- `/account` URL was not deployed (404 response).