Athena Research (2023)
ath 1.000001/cons-0001

Eliminating Scams and Phishing Websites from Web-3: Introduction to Snap Signals

Georges Chouchani[1]
[1] Athena Consulting

Abstract

Scams have been prevalent in the Web-3 ecosystem for a while, and from what it appears, they are here to stay. In 2022 alone, exploits and scams amounted to a total of 3.6 billion USD in lost user funds. Now many could think that phishing websites are only made for new users or "newbies", but even the most veteran Web-3 users have fell to similar scams.

Phishing websites are websites made by hackers to deceive users into believing that they are on an official website of a decentralized application, while actually when they interact with the website, they would be sending information or funds directly to hacker-controlled smart contracts or wallets.

This article will go over a potential method which we have called "Snap Signals" which aims to eliminate users interacting with phishing websites once and for all to promote a safer environment for retail users and investors.

1 Introduction

A Snap Signal is a signal that is sent from a decentralized application to a browser wallet to verify the validity of a website. For a website to be considered a valid website, it should go through a rigorous KYB process (to be discussed later on).

When a transaction is then submitted through the website, if a snap was not sent by the website to the wallet, a warning or error is raised to signal the issue. The website URL is then passed to the backend where artificial intelligence tools algorithms determine whether the website is a phishing website, or could potentially be a website that did not go through the KYB process.

A snap signal is sent when a user connects their browser wallet to a certain decentralized application. When this action is initiated, a snap hash is sent from the browser to the wallet, which then verifies the authenticity of that hash. If the hash is validated, the user is allowed to connect their wallet, if not an error appears that the website is not valid and could potentially be a scam.

Corresponding author(s): Georges Chouchani, Athena Consulting, georges@athenaconsulting.io

2 KYB Process

The first step of the whole process is for protocols/startups/L1s/L2s to identify and verify their authenticity. In order to do that, they will need to undergo a rigorous application process. This allows to ensure that only valid protocols register.

1. Provide proof of the registered domain.
2. Provide proof of ownership of a Github repo or organization.
3. Core team must join a video call to verify their identity.

When this is completed, the data is compiled into a JSON file and is hashed and stored in a merkle tree. The merkle tree is then stored in a backend and the root stored in a smart contract to be able to allow anyone to verify authenticated protocols, which will then be altered as new protocols are added to the system.

After KYB, the core team is given a hash that consists of the merkle proof that is going to be passed later on to the Snap Signal for verification. This hash should *NOT* be shared with anyone or else they can impersonate them during verification.

3 Merkle Trees

It is important to note that due to the nature of Merkle Trees, any addition to a tree changes the Merkle Proof generated in order to verify that a leaf exists in a tree, and therefor due to the dynamic action of validating and adding new protocols, one merkle tree would not make the cut.

A better option would be to store different MTs periodically (for example once a week) in batches under a unique identifier. Then, when a proof is generated for a certain batch of addresses, they are appended to a unique identifier that allows the access and use of this certain MT.

4 Smart Contracts

In order to give access to the verification system not only to wallets but to any developer that wants to integrate it into their system, a smart contract for each MT is created that stores the MT root and has a verify function that can be called to verify a merkle proof. Expanding on what was previously discussed in the previous section, there is going to exist multiple MTs in the system that represent a batch of addresses. In order to be able to keep track of all smart contract addresses, and allow them to match how they are stored in the database, we need a registry smart contract.

4.1 Snap Signal Registry

A snap signal registry is a smart contract that is responsible of deploying and keeping track of each individual MT smart contract that exists. When a new MT is created on the backend, the registry is called to create a new contract, store it in an indexed map with the index as the unique identifier of the MT and deploy the contract with a specific MT root. The registry also contains a function that allows a user to provide a MT proof directly to the registry, which then gets the index of the tree, and calls the

smart contract to verify the proof. This allows for easier and streamlined access to all merkle trees in the system.

5 The Snap Signal

The snap signal is the verification process when an action is submitted, in that case being connecting a wallet to a decentralized application. The snap signal takes as input the merkle proof hash that was initially given during the KYC process. Before connecting to an application, it verifies the proof to ensure that the snap was initially registered during the KYC process, if verified, the user is allowed to connect to the website, application etc... If not verified, an error is raised to the user that the website does not seem to be authenticated, and to proceed at their own risk.

6 Further Applications

Although this verification system was used to eliminate phishing in Web-3, the same system can be used for different use cases in the ecosystem that are deemed viable. For example, a similar verification system exists for token airdrops to avoid storing a large list of addresses on the blockchain. Another potential use case would be in IoT to register devices to interact with each other using snap signals without having to go through an authentication method each time.

7 Further Readings

1. Merkle Tree Solidity Smart Contracts
2. Web3 Phishing Has Finally Arrived
3. Merkle Airdrop