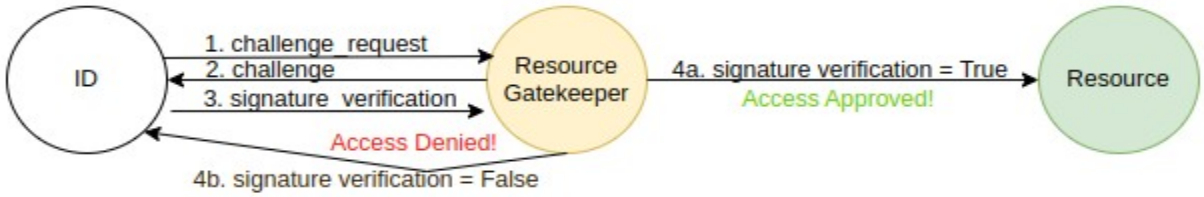
May 6, 2023

MoneroAuth Authentication



Worldwide Web Resources[1]

1. **challenge_request –** the entity requesting authentication to a resource sends a challenge_request message to the resource gatekeeper. This is typically in the form of a URL.

   Typically websites implement challenge_requests via a user action such as a user clicking a “signin” or “login” button.

   Resources should be uniquely identified via a MoneroAuth ID[2] that is referred to as a resource_identifier. Resources may be of many different types, i.e., a web app, machine/device, etc.[3]

2. **challenge message –** Once a challenge_request is received, the resource gatekeeper will provide a challenge message to the requester. The challenge message will look similar to:

   {"json":"2.0","method":"challenge","params":{"signature_verification":"http://demo.moneroauth.com/verify","challenge_string":"6933797398516"}}

   The challenge message will consist of a minimum of two parameters:
   - signature_verification – for world wide web resources this will be the URL where signature verifications will be sent.
   - challenge_string – this is a random string provided by the resource gatekeeper that the requester must digitally sign.

3. **signature_verification –** Once the requester receives the challenge message from the resource gatekeeper, the requester must digitally sign the challenge_string and provide a minimum of three parameters to the URL specified in the challenge message “signature_verification” parameter:
   - The challenge string that was signed
   - The ID of the signer
   - The signature

1 Resource gatekeepers may provide MoneroAuth authentication service via a number of different protocols.
2 A MoneroAuth ID is a Monero wallet address. They are one in the same.
3 See Appendix I

With the above three data parameters the resource gatekeeper will attempt to verify the signature provided. If the signature is verified (proving the requester holds the private key for the id), the resource gatekeeper will have successfully authenticated the requester and may proceed to authentication.

If the signature is not verified, then the resource gatekeeper will not allow access to the resource.

A typical world wide web signature verification URL would look similar to:

http://demo.moneroauth.com/verify?challenge=8142438938927&id=43mq8WjgSp9cdEztUZrsyNMjTNXy1QPJdWD8czEzzF8iPBVxFtMbS4EFVB1PpW2AapjY7nVsdp2nZGG2qcEzzp8A3AyKuts&signature=SigV2goDC4skaJP5G9HNQJiNUivWLX6tDhNoJQUYZpUDa5jHECdyVk73Kzy2bK6WeJZZcDa2gF6Jj9tHCG59rFbiPQAxr

A verified signature will always be a verified signature.

So if all the resource gatekeeper does is to verify a signature, then an ID that once successfully authenticated in the past, could continue to present past signature verification data on future authentication attempts with the expectation of successful authentication.

If a third party were to obtain another party's valid signature verification data, and the third party submitted that signature verification data for authentication, the third party could falsely authenticate if the gatekeeper only checked to see that the signature verification data was valid.

To avoid signature verification reuse, gatekeepers need to implement procedures to ensure a specific challenge string is not reused.

Appendix I

Machine-to-machine interfaces typically would send an explicit message similar to:

"json":"2.0","method":"challenge_request","params":
{“gatekeeper_resource_id”:”@resource_name:matrix.org"challenge_channel":"!
GORzWRbicCXEkMZHQH:matrix.org”,
“authorized_id”:”44HCjBysPb72ZeQFe8eaqCeH2wQSo6WAb8S1XskXSCYAEukxQC5DBr
hTkL1aLzbKy1aFF8d4SFmcq2qjPi4SVRd1AJ2cQM6”}

Where the specific resource is uniquely identified via the resource_id, the communications channel is uniquely specified, along with the id of the requester (authorized_id) being specified. For sophisticated resources (resource packages consisting of multiple resources managed by a single gatekeeper), a specific resource_id may be included to uniquely identify a specific resource among many available resources.

Additional details can be found in the Machine-to-Machine Authentication document.

Appendix II

The MoneroAuth protocol leverages Monero for ids, signatures and signature verification.

The authbot makes a *monero-wallet-rpc* ***sign*** call to sign the challenge_string:

https://www.getmonero.org/resources/developer-guides/wallet-rpc.html#sign

The resource gatekeeper attempting to verify a signature makes a *monero-wallet-rpc* ***verify*** call to verify a signature:

https://www.getmonero.org/resources/developer-guides/wallet-rpc.html#verify

Monero Reference Information:

Monero web site: https://www.getmonero.org/

Zero to Monero – Second Edition: https://www.getmonero.org/library/Zero-to-Monero-2-0-0.pdf