Swarm Feeds

Javier Peletier
CTO @ Epic Labs
@jpeletier
jm@epiclabs.io

Agenda

- About Epic Labs
- What are Swarm Feeds?
- How do Swarm Feeds work?
- How do I use Swarm Feeds?
- API introduction
- Demos

swarm
epic labs

epic labs

#BeEpic!

A Software Innovation Center

Helping medium to large companies innovate

Experts in Media, Video Encoding & Machine Learning applied to Video

Blockchain consulting services & training

LightFlow

by epic labs

Ethergit

Swarm Feeds

What are Swarm Feeds?

A microblogging platform:

- You can post updates about a certain topic
- You can read other people's updates about a particular topic
- You can also see older posts

What are Swarm Feeds?

A key-value store, where:

- Each user can only write to their own key space.
- You can read yours and other users' key space
- You can retrieve older versions of the value for a key.

Addressing Feeds

A Feed is addressed with only two things:

- User's ethereum address
- Topic, which is a 32-byte array.

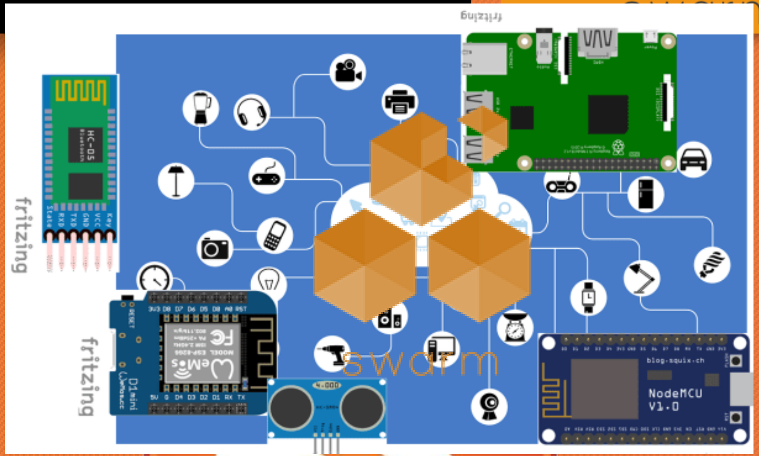
```
 type Feed struct {
  Topic Topic
  User  common.Address
}
```

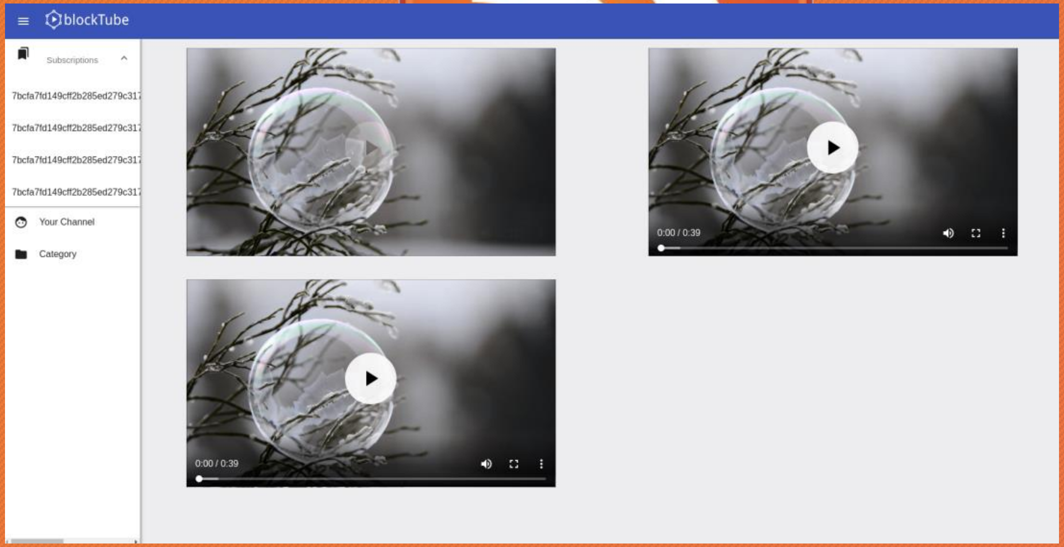
Thus, there is **one feed per topic per user**.

Imagine you had a Twitter account for each possible topic!

Benefits & Applications

- Instantly alter content in Swarm without on-chain transactions
- Enable DApps to persist content easily
- Quickly retrieve older versions of content
- Enable social-type DApps
- Enable decentralized IoT



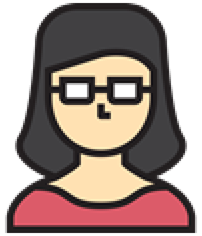
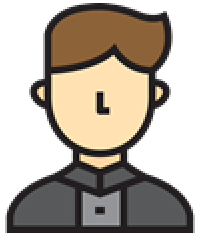

blockTube
Subscriptions
7bcfa7fd149cff2b285ed279c317
7bcfa7fd149cff2b285ed279c317
7bcfa7fd149cff2b285ed279c317
7bcfa7fd149cff2b285ed279c317
Your Channel
Category
0:00 / 0:39
0:00 / 0:39

Example feeds

epic›labs

swarm

TODAY:

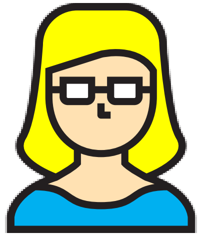
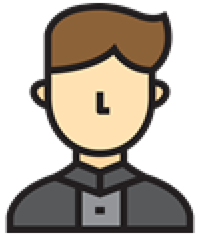
Topic	User 0xAAAA…	User 0xBBBB…	User 0xCCCC…
avatar			
local-weather	“sunny”	“cloudy”	
website		0xbacde124…	0x9876533…

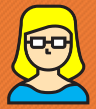
Thus, QueryFeed(0xAAAA, “avatar”) ->

QueryFeed(0xBBBB, “local-weather”) -> “cloudy”

You also have the time domain...

THE NEXT DAY...:

Topic	User 0xAAAA...	User 0xBBBB...	User 0xCCCC...
avatar			
local-weather	“sunny”	“rainy”	
website		0x12345678...	0x9876533...

QueryFeed(0xAAAA, “avatar”) -> 

QueryFeed(0xAAAA, “avatar”, **YESTERDAY**) ->

QueryFeed(0xBBBB, “local-weather”, **YESTERDAY**) -> “cloudy”

Posting to / reading a Feed

To **post** to a Swarm Feed, you need two things:

- A private key/address pair to sign your update
- The topic of your post

To **read** a Swarm Feed you need to know:

- The user's ethereum address
- The topic under which the user is posting
- (Optional) if interested in older posts, the timestamp to look up

Feed Topic

A Feed Topic is a user-defined **32-byte array**

- Can be anything: a string, a hash of something...
- Make sure users can easily derive the topic

For example, if it is known that people post their avatar to the "`avatar`" topic, given somebody's ethereum address you could retrieve their picture by looking up that feed.

Feed Payload

- Feed payload data is limited to `feed.MaxUpdateDataLength` bytes, or **3963** bytes.
- In general, you will want to post to the feed **the hash of the actual content**, rather than the content itself.

Feed Manifest

A Feed manifest is just a JSON that contains the bare minimum information to look up a feed, that is, again:

- User's ethereum address
- Topic

This JSON can be then uplodaded to the regular content-addressed Swarm and obtain a hash we can use in ENS and `bzz:/`

Example feed manifest:

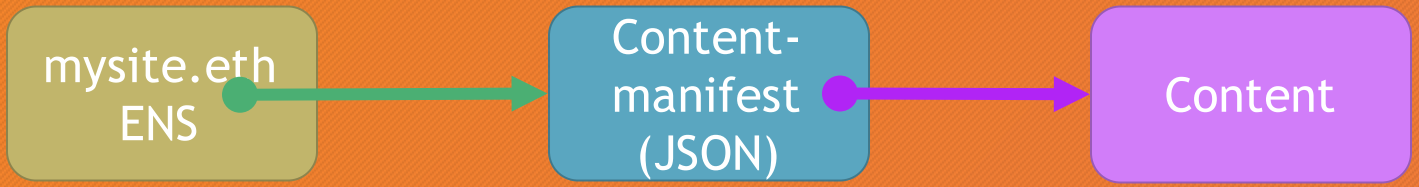
```
{
  "entries": [
    {
      "contentType": "ap            z-feed",
      "feed": {
        "topic": "0x7765627369746500000000000000000000000000000000000000000000000000",
        "user": "0x7a2e393025c567ec4089d34f393ae6b5c234536a"
      }
    }
  ]
}
```

ASCII for "website"

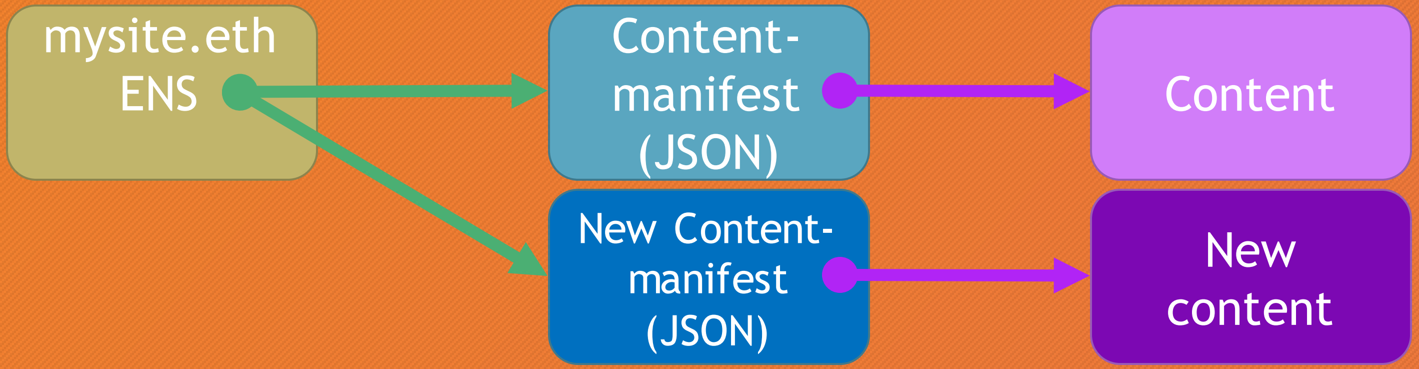
ENS and feed manifests

Before:

- Put in ENS hash of **content manifest** (output of `swarm up`)



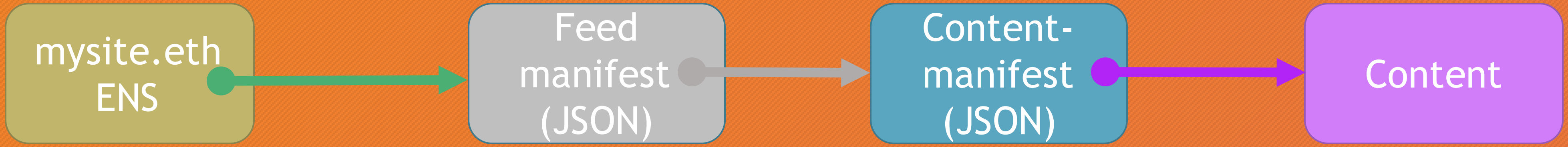
- To Update: Put in ENS the hash of a new content manifest. (requires another transaction)



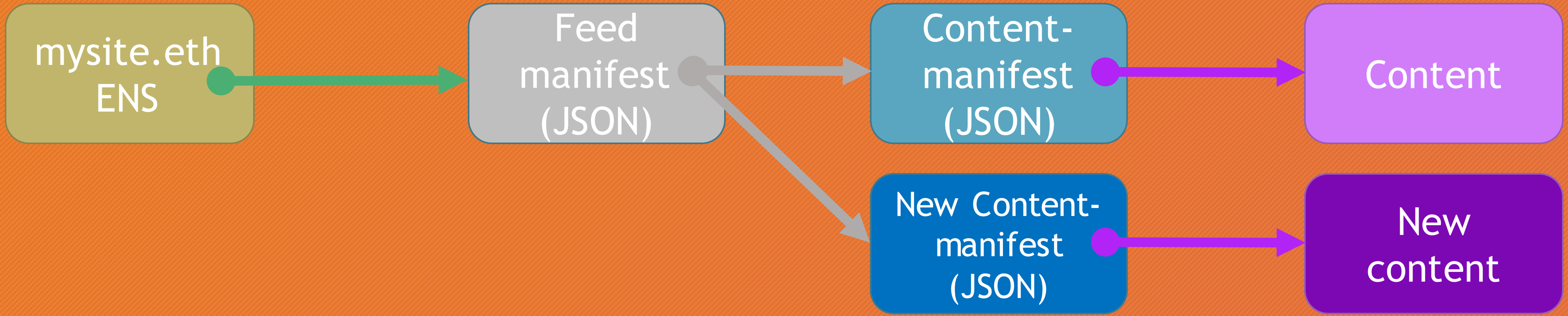
ENS and feed manifests

With Swarm feeds:

- Put in ENS hash of **feed manifest** (output of `swarm feed create`).



- To update: simply post to the feed the new content manifest hash



HTTP API

- To read updates:
  - `GET /bzz-feed:/?topic=0xAAA...&user=0xBBB...`
  - (update, if found, will come in the response body)

- To post updates:
  1. Learn feed status: `GET /bzz-feed:/?topic=0xAAA...&user=0xBBB...&meta=1`

  2. `POST /bzz-feed:/?topic=0xAAA...&user=0xBBB...&time=15...&level=xx...&signature=0xCCC...`
  - Update must be attached in the request body.

GO API

- To read updates:
  - Build (or reuse) a `feed.Query` object, indicating user+topic
  - Call `client.QueryFeed()` with your `Query` object. You will receive an `io.Reader` with the raw feed data.

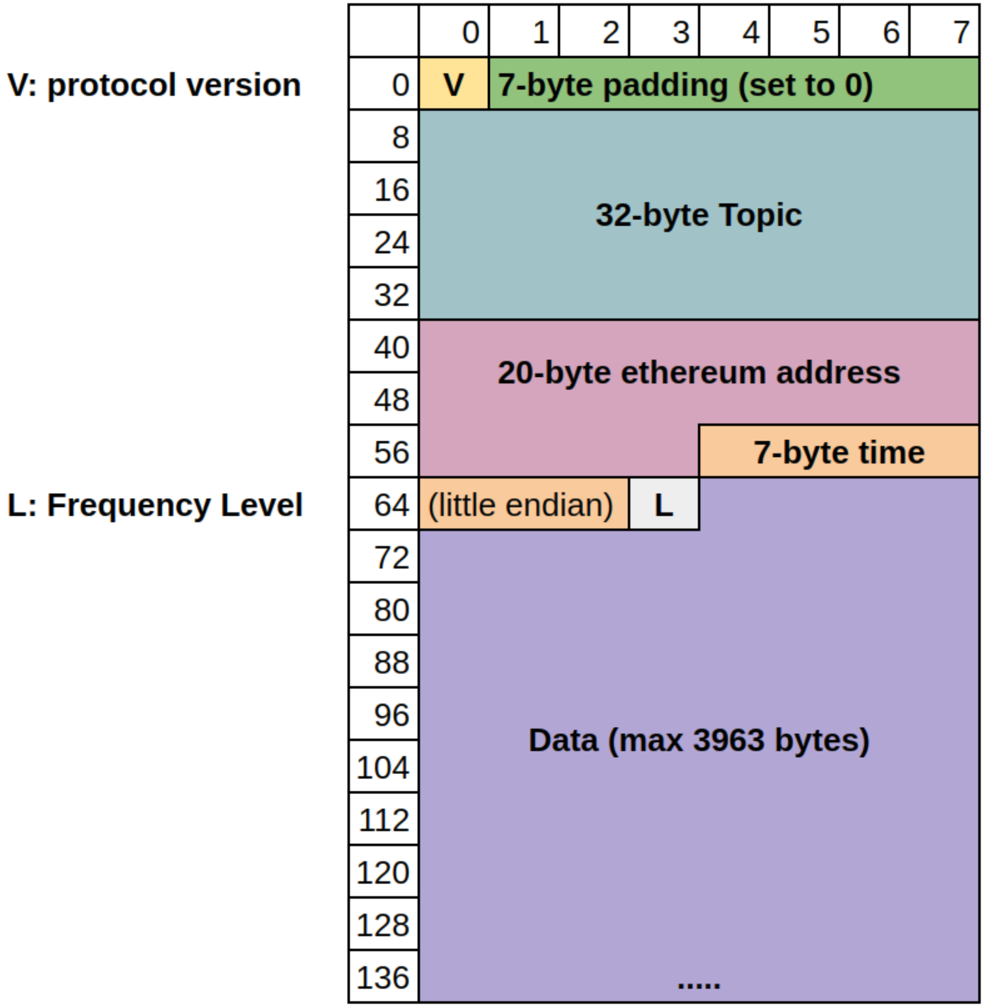
- To post updates:
  1. Build (or reuse) a `feed.Query` object
  2. Obtain a `feed.Request` object with `client.GetFeedRequest()`
  3. Attach data with `Request.SetData()`
  4. Sign with `Request.Sign()`
  5. Call `client.UpdateFeed()` with your signed `Request`

Digest & Feed Signatures

Feed signature is the Ethereum ECDSA 65-byte signature of the digest of the feed update structure.

Digest is the **keccak256** hash of the feed update structure

Signing code available in Go, JS and C++ (new).



V: protocol version

L: Frequency Level

DEMO – updating a website using Feeds

epic›labs

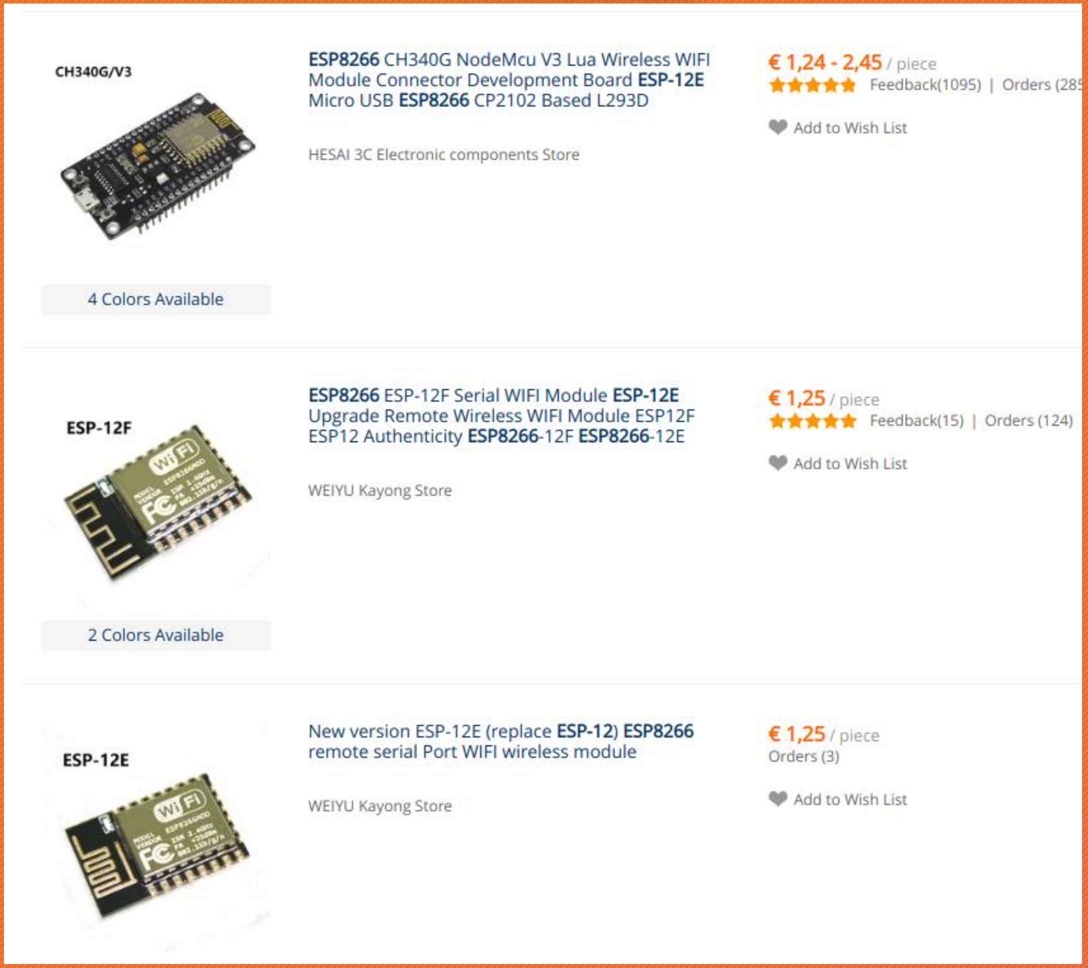
swarm

Swarm Feeds IoT demo!!

Feeds provides an excellent building block for IoT

Demo: get a cheap ESP8266 to sign and publish a temperature feed to Swarm!

Dashboard: https://swarm.epiclabs.io/bzz:/4dea5857577cc7c93bd8affa04acfdf8c98295aae888118bfae092b652509f98/



ESP8266 CH340G NodeMcu V3 Lua Wireless WIFI Module Connector Development Board **ESP-12E** Micro USB **ESP8266** CP2102 Based L293D

HESAI 3C Electronic components Store

€ 1,24 - 2,45 / piece
Feedback(1095) | Orders (28

ESP8266 ESP-12F Serial WIFI Module **ESP-12E** Upgrade Remote Wireless WIFI Module ESP12F ESP12 Authenticity **ESP8266**-12F **ESP8266**-12E

WEIYU Kayong Store

€ 1,25 / piece
Feedback(15) | Orders (124)

New version ESP-12E (replace **ESP-12**) **ESP8266** remote serial Port WIFI wireless module

WEIYU Kayong Store

€ 1,25 / piece
Orders (3)

Areas of improvement:

epic labs

swarm

- Accelerate lookup algorithm
- Simplify signing feeds in different platforms
- Push for an official JS client
- Improve HTTP API
- User mount point in `/bzz:/`

Thanks!   Q&A

Feeds Guide: **https://swarm-guide.readthedocs.io/en/latest/usage/feed.html**

Appendix: Validators in Swarm

Validator is a function that accepts or rejects a chunk

- Content-addressed validator:
  - `valid = key == H(value)`

- Feeds validator:
  - value: user, topic, epoch, payload, signature
  - `valid == ecrecover(H(user, topic, epoch, payload), signature) == user && key == H(user, topic, epoch)`