# bunsanweb and decentralized web

Decentralize Web Summit 2018

# Our view of the Web

A Network of User-Agents(browsers) and hyperlinked Resources

- Not a network of servers for their clients

Resource is a hyperlink collection with its URI to GET/PUT/POST/...

- Web-server is a HTTP protocol handing proxy for the Resources

# Web of something

Web of document: a browser communicates to document resources

- Hyperlink can directly link resources across their Web servers

Web of person: a browser communicates to person resources?

- Web Service as a proxy of multiple person resources with systems
- Browser just provides a personal information to POST as a resource on Web Servers

Web Service as intermediary between person and person

- => a central of inter person network

# Views toward decentralization

Decentralization of universal or monopolistic systems

- Decentralized incentive/motivation systems
- Decentralized persistent storage systems
- Fare contract systems for conflicted resources; such as name registry

Endpoint enhancement for decentralize

- Web browser enhanced with decentralizing features
- Programming functionalities provided for decentralized architectures

# Our view of decentralizing the Web

Return to **end-to-end** principle

- => reducing intermediary centric factors from endpoints

Web endpoint should also become an **endpoint of inter-person** systems

- Browser could become a **resource of myself**
- Browser could perform **endpoint-scripts for functions of the systems** broken into decentralized manner

**Web of Programs** with User-Agent

- Run programs **on each User-Agent** (browser)
  - mix add-ons in a browser
  - mix modules in a HTML you write
- Programs **for Web Resources**
  - process data from remote Resource
  - privide as Resource
  - (communicate to other program as a Resource)
- Share mixed/mixable programs **written from your wish**
  - same as documents on the Web

# What "bunsanweb" tackles

Endpoint-scripting

- mix server-side scripting features into client-side for **both accessing and producing as Web resources**

Universal event stream

- for communicating unspecified endpoint scripts **without depending specific centric channels**

Endpoint-relative hyperlinked space

- each endpoint views resources **from each local linked to the universal** on Web
  - e.g. the "personal data" is also relative resource for each person. it may link to "friends" on universal.

# What we've made

`anatta-engine`

- Prototype JavaScript package of runtime environments for endpoint-scripting
- Features:
  - Endpoint-scripting
  - Endpoint-relative hyperlinked space
- Runtime: node.js
  - Emulating browser `window`/`document` environment for JS runtime
- `grp`: successor as reverse proxy for scripts on vanilla browsers

`hashnet`

- Prototype JavaScript package of peer network for universal event stream
- Features:
  - Universal event stream
- Runtime: node.js and electron
  - Console UI and demo with electron app
  - Event as ES6 Proxy wrapped DOM Element

# bunsanweb: endpoint-scripting

Run programs on **vanilla** modern browsers

- Program as JavaScript codes **within HTML**
- Programming with the **standard JavaScript APIs** in Web browsers
- Implementations are **hidden** behind the standards as much; such as DOM `Events` or `fetch()`

Scripts **directly respond** as Web Resource via "general reverse proxy"

- With non-conflicted URI based on **public key hash** identity
- Responding with a standard `FetchEvent` of `ServiceWorker` (and `Request`, `Response`, `Blob`, ...)

Scripts handles remote resources with **loosely-coupled** way

- **HTML** as primary format for `href`
- Document as abstract hyperlink container: **URI links are special** from other property data
- Today, HTML DOM can be wrapped with **ES6 Proxy**

# bunsanweb:
# universal event stream

Content-based event stream

- For open network, event streams should not be limited by upper-side such as its publisher or published queue
- Event is filtered by its content at the endpoint
- Event is ordered when it embeds parent events as hyperlink

Sharing immutable event document into universal

- Identity (part of URI) as content hash
- Signed with actor's elliptic-curve crypto(ECC) key

Peer network to expand their universal

- Each peer has own universal space of events even if non-connected peers
- Federated peers make an union universal space with fetching event lists (as resource) each other

# bunsanweb:
# universal event stream (cont)

"contexts" **axes** to make a space of events

- Each tag in the "contexts" denotes some of events property structure **for filtering** or **for data processing**
- Existence of tags puts these event at the **position of a universal sphere**
- It also locates a **region of events** such as streams or actors

# bunsanweb:
# endpoint-relative hyperlinked space

Endpoint-script itself is same **everywhere**

- But its behavior is **different with local data values** where it runs
- For scripts being available anywhere, data **locations should be same**

**Standard structure for local** resources

- It is similar as "start page" of web browser
- Processing hyperlink relations **started from the relative top** resource

Initial local resources

- Key-pairs: for identity of universal event stream
- Personal profiles: as event actor
- Script URIs: to run for local resources

# Change with bunsanweb: open systems built on peer relations

- Decompose a system as person-based features and aggregated data features
  with these data ownership
- Add new peers for aggregated data from existing peers
- Make each features as endpoint-scripts to run on each peer for sharing their data as privacy protected way; signed or encrypted with ECC
- Open some of features to others, share events on universal event stream
- Or apply existing open features of other systems for enhancing, it also accepts events on universal event stream

# Connect to decentralized technologies

Decentralized Incentive system; such as crypto token systems

- For sustainability of network
- We think only a best-effort way of peers and reverse proxies
- To be applying crypto token systems for them
  - proof of burn, or self blockchain

Smart contracts; such as Ethereum, chainspace

- For universal fairness on whole of network
- We think it is enough with peers for separated aggregated data
  - Accepting aggregated data peers is depends on judgment of each peer
- Replacing these aggregating peer with smart contracts between peers

# Connect to decentralized technologies (cont)

Decentralized persistent storage; such as IPFS, Dat

- For sharing content data, not URI
- On universal event stream, peer primary manages event list as URLs of events
- We think event data is also stored in peers (as they made there)
- It can use decentralized storages to sharing event data

Blockchain itself

- For strict ordering of transactions/events
- We think events are just bunch of events or partially ordered enough; such as git branches/forks
- But there is head of each local peer as one of branches/forks
- Blockchain for events may be required for realizing smart contracts
- It may use proof of burn, or local votes with event actor ids

# Info

Project

- https://bunsanweb.github.io/ (work in progress)
- https://github.com/bunsanweb/
  - Documentations: https://github.com/bunsanweb/bunsanweb/

Repos

- anatta-engine: https://github.com/bunsanweb/anatta-engine
- hashnet: https://github.com/bunsanweb/hashnet
- grp: https://github.com/bunsanweb/grp

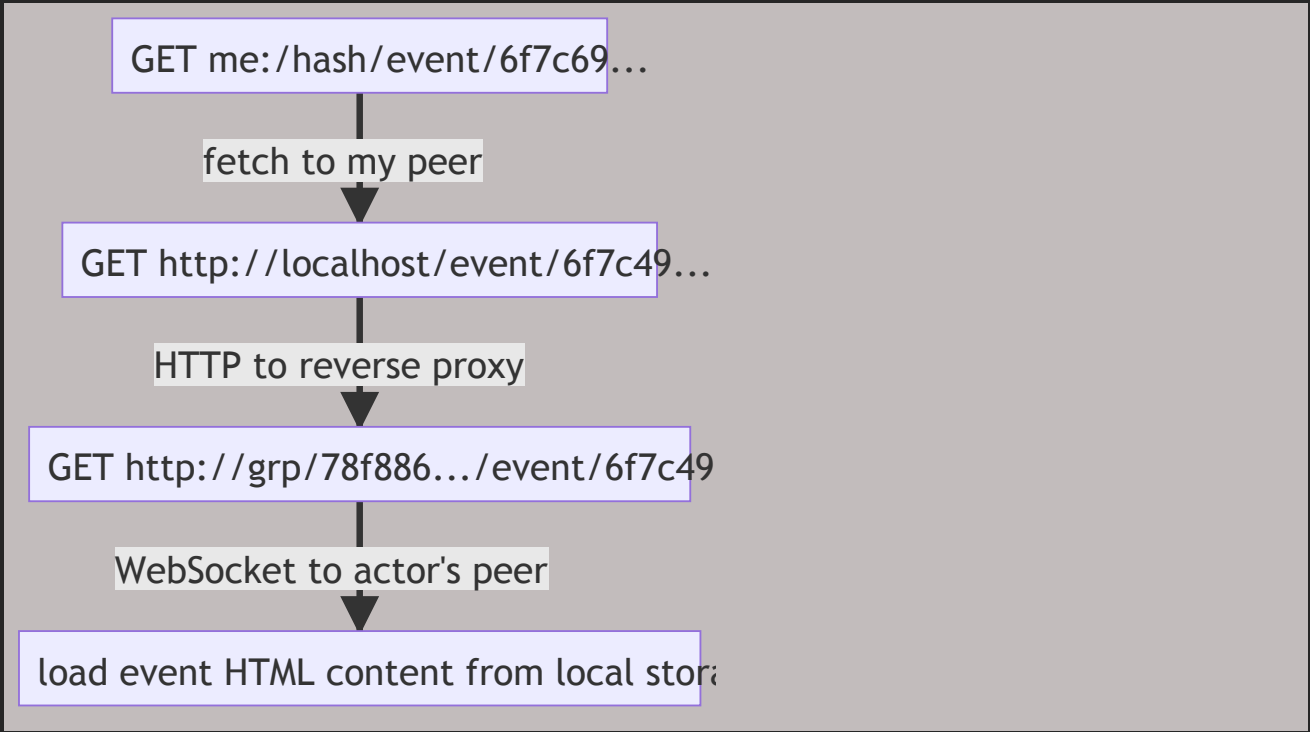
# A1: Features behind the standard APIs

```
<html>
  <head><script type="module">
import ReverseTarget from "./grp.m.js";
(async function main() {
  const target = await ReverseTarget.connect("http://localhost:3000/");
  target.addEventListener("fetch", ev => {
    ev.respondWith((async () => {
      const body = `Hello World! from a Browser Tab: ${ev.request.url}`;
      return new Response(body, {status: 200, headers: {
        "content-type": "text/plain;charset=utf-8",
        "access-control-allow-origin": "*",
      }});
    })());
  }, false);

  const a = document.quertSelector("#link");
  a.href = `${proxyUrl}${target.ident.id}/`;
  a.innerHTML = `open proxy page: ${a.href}`;
})().catch(console.error);
  </script></head>
  <body><a id="link" target="_blank"></a></body>
</html>
```

With `Response`, body also accepts standard `File`, `Blob` , and `ReadableStream`

# A2: Universal event content between peers with grp



# A3: endpoint relative link space from local "me:" to universal