1. Technical Overview/Call flow

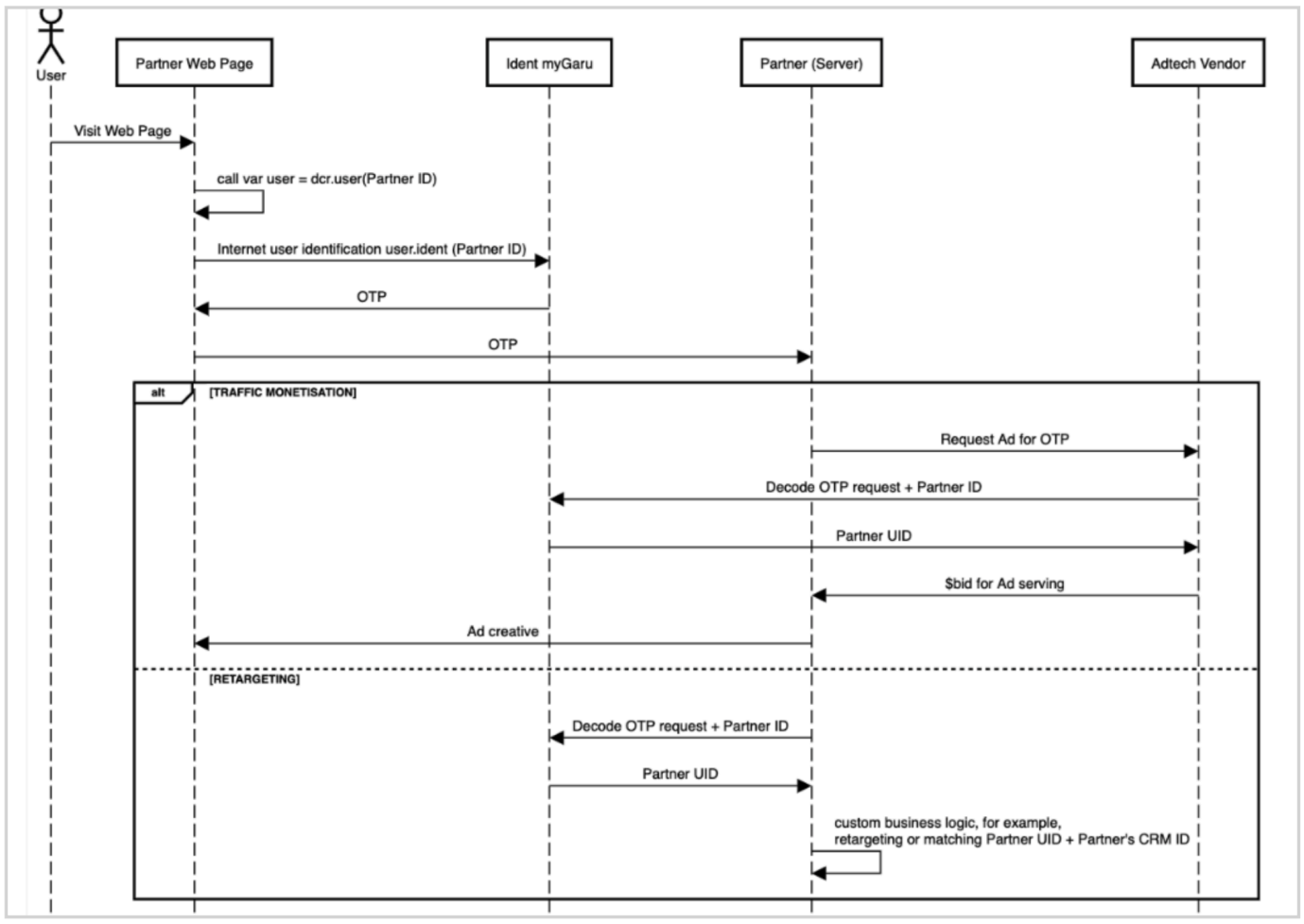

Figure 1. Call flow

- Partner implements provided JavaScript applet to a web page.
- When a User visits the web page JavaScript applet sends requests to Ident myGaru containing Partner ID.
- Ident myGaru responds to the web page with OTP.
- Partner’s web page sends OTP to Partner’s server.
- (Option 1) Partner shares OTP with connected Adtech vendors for requesting $bids on showing a targeted ad to User.
- (Option 1) Adtech vendor sends OTP and vendor’s own Partner ID to Ident myGaru.
- (Option 1) Ident myGaru returns a Partner UID as a User identifier related to the Adtech vendor.
- (Option 1) Adtech vendor checks available $bids related to the Partner UID and returns an appropriate $bid for showing an ad to a User.
- (Option 1) Partner chooses the best $bid and sends ad creative to the web page for showing it to a User along with content.
- (Option 2) Partner sends OTP and Partner ID to Ident myGaru.

- (Option 2) Ident myGaru responds with Partner UID as a User identifier related to the Partner.
- (Option 2) Partner can link received Partner UID to internal User's IDs and/or use it for ad targeting campaigns through Adtech vendors integrated with myGaru. In this case, Adtech vendors use a similar flow as described in Option 1.

3. Lite Integration (Onion ID)

Lite integration enables Partners to use Onion ID identification with minimal effort and doesn't require any software deployment.

3.1 Network allowlist & Partner ID emission

After the contract is signed, the Partner must provide IP Address (es) your systems will use to call the Onion ID Ident Service.

- We accept IPv4/IPv6, single IPs
- If your egress IPs change, notify us at least 2 business days in advance.

myGaru will whitelist these IPs and issue your **Partner ID** (your unique identifier in our platform, which is required to enable Onion ID).

3.2 JS Applet/SDK integration

JS Applet must be embedded into the Partner's website in the header section to enable Onion ID identification. JS Applet is loaded asynchronously, and the ID retrieval is queued to execute after the script is available. The global myGaru object is created if it does not exist, and a command queue is established to handle the getId function. This approach is non-intrusive, maintains user experience, and adheres to privacy compliance, providing effective monetisation.

Note**: At this point, myGaru provides JS Applet for Web environment, Android/iOS SDK will be released in **Q1 2026

JS Applet script:

```
<script src= 'https://cdn.mgaru.dev/static/myGaruStandalone.js'
async></script>
<script>
    var mygaru = window.mygaru || {};
    mygaru.cmd = mygaru.cmd || [];
    mygaru.cmd.push(function() {
         const user = mygaru.init(PARTNER_ID);
         user.ident().then(function(OTP){
             console.log('THIS IS MYGARU TOKEN',OTP)
    })
</script>
```

***Note**: **PARTNER_ID** must be replaced by the ID provided by myGaru Manager*

JS Applet hosting options:

- By default, the script is hosted on myGaru CDN: https://cdn.mgaru.dev/static/myGaruStandalone.js'
- Partner can also host the script on a custom CDN. If the script is self-hosted on a custom Partner's CDN, myGaru will alert you about new releases. Please refresh your hosted copy promptly to stay compatible and supported.

3.3 OTP Decoding API

Partners who are authorised to use identification from myGaru (have a personal Partner ID and whitelisted IPs) can request decryption of OTP received by them or by any other participant within myGaru. Responding to such a request system provides them with Partner UID, a user identifier related to the Partner.

Request:
POST/v2/decode HTTP/1.1

Host: ident.mygaru.com
Content-Type: application/json

```
{"otp": "ipKtuZ1EWovDiwx0aEdij/TZuBxDWWs/k0n8BBHZwsoDD5WbIKwpZGA3LkbsyS0Fn0tWK86sb3G9RgGP"}
```

JSON object structure:
otp - string. OTP value provided by Ident Service

Response:
Content-Type: application/json

JSON Object structure:
uid - string. User Identifier associated with OTP in request
unixtime - string. Unix time stamp when OTP was generated.

Status code	Description
200 - HTTP **OK**	OTP and associated Partner UID found in DB and provided in the response
400 - HTTP **Bad Request**	The request body is malformed or missing required values
403 - HTTP **Forbidden**	The request for OTP decoding was made from an IP address that is not whitelisted; OTP decoding is only possible from pre-agreed whitelisted IP addresses
410 - HTTP **Gone**	OTP lifetime is expired

Note: *In case of a non-200 status code the body is always empty.*

4. Full Integration (Onion ID+DCR Client)

To unlock the full value of the myGaru solution, partners must deploy the **DCR Client** component within their own premises in addition to the Onion ID integration. This software enables direct monetisation of your first-party data, and grants privacy-safe access to other partners' contributed data on the platform - so you can build combined audiences (with no PII exposure) and run audience analytics across properties. Partners have full control over their data: advanced access/visibility/pricing management settings.

In short: more yield from your own data, scalable data collaboration, and activation-ready segments without compromising compliance.

DCR Client Requirements:

Platform/OS	64-bit Linux (Ubuntu, Debian, Redhat Enterprise Linux or derivatives (Rocky, Alma or Oracle Linux for example).
CPU	2C/4T CPU (or 4 vCPU for virtualised environment) at ≥ 2 GHz
RAM	8 GB
Disk/memory	100 GB Solid-State Disk (SSD);
Network	100Mbps network connectivity