WEB_UK 生成逻辑分析报告

> 网站： https://www.youpin898.com/
> 分析日期：2025-11-16

---

1. 概述

WEB_UK 是悠悠有品网站使用的用户/设备唯一标识符，存储在浏览器的 localStorage 中。它是一个由**服务器端生成**的字符串，基于客户端收集的设备指纹信息。

示例值：

```
5FTId5j7gApxnWHIB8E1OqJL3NaHupTTFst9YgAhlnOjHHHrgTFqU49kNykh8cX1K
```

---

2. 完整生成流程

流程图

```
客户端                                          服务器
  |                                               |
  |-- 1. 收集设备指纹 ----------------------->|
  |   (Canvas、字体、浏览器信息等)                |
  |                                               |
  |-- 2. 生成随机 AES 密钥 ------------------>|
  |                                               |
  |-- 3. AES 加密设备指纹数据 ---------------->|
  |                                               |
  |-- 4. RSA 加密 AES 密钥 ------------------>|
  |                                               |
  |-- 5. POST /api/deviceW2 ----------------->|
  |   {encryptedData, encryptedAesKey}            |
  |                                               |
  |                                               |-- 6. RSA 解密 AES 密钥
  |                                               |
  |                                               |-- 7. AES 解密设备数据
  |                                               |
  |                                               |-- 8. 生成 WEB_UK
  |                                               |
  |<-- 9. 返回加密响应 ------------------------|
  |   {u: WEB_UK, deviceUk: WEB_DEVICE_UK}        |
  |                                               |
  |-- 10. 存储到 localStorage ---------------->|
      localStorage.setItem("WEB_UK", ...)
```

3. 详细步骤分析

步骤 1: 收集设备指纹信息

客户端通过 JavaScript 收集以下信息：

3.1 Canvas 指纹

```
getCanvasParams(params) {
  const canvas = document.createElement('canvas');
  const ctx = canvas.getContext('2d');

  // 设置画布尺寸
  canvas.width = Wt.width;
  canvas.height = (Ft.length - 1) * (Wt.lineHeight + 2);

  // 绘制文本
  ctx.font = Wt.font;
  ctx.textBaseline = 'top';

  Ft.forEach((text, index) => {
    ctx.fillText(`${text}: ${params[text]}`, 0, index * Wt.lineHeight);
  });

  // 转换为 base64 图片数据
  const fingerprint = canvas.toDataURL('image/png');
  this.canvasParams = fingerprint;
}
```

原理： 不同设备/浏览器渲染 Canvas 时会有细微差异，生成唯一指纹。

3.2 字体信息

```
getFontsParams() {
  const fonts = document.fonts;
  const fontList = [];

  fonts.forEach(function(font) {
    fontList.push(font.family);
  });

  return { fonts: fontList };
}
```

原理： 每个用户安装的字体不同，可作为识别特征。

3.3 额外参数

```
getExtraParams() {
  const existingUk = window.localStorage?.getItem("WEB_UK");
  let params = xt; // 基础参数对象
  const fontsParams = this.getFontsParams();

  if (existingUk) {
    params.uk = existingUk; // 如果已存在，带上旧的 UK
  }

  this.extraParams = { ...params, ...fontsParams };
}
```

步骤 2: 数据加密

2.1 生成随机 AES 密钥

```
// 生成随机字符串作为 AES 密钥
generateRandomString() {
  // 实现细节在混淆代码中
  return randomAesKey;
}
```

2.2 AES 加密设备数据

```
encryptWithAES(data, aesKey) {
  // 使用 AES 算法加密设备指纹数据
  return encryptedData;
}
```

2.3 RSA 加密 AES 密钥

RSA 公钥:

```
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA2IYU9R1HrmaTTIw9BhdhTnxoR4gXUcu2g26YVX+6UpgKsGUVDum
UvvS67GxIPuRE9Y4L59uu7HkSVz61HZVIx+XqTANTH/sJf+eVmuaCnSgK+KZOKv64B9s1L0BXtpz3xpbKnjnmiRGDG0GKSx
ImcQZVzLzEaZc58olo3zXrIEfUBLeZ8bCD66PTqPS5+s9pAulS4mj/OZFcnoxhIR6h5L0XfZqdBaq6nQhYLCrrpKYmf4rzU
WefSvS1GKk+DgofrmUat+aOboEKLGH+02EndP1etmduJiEkSNXKzs1NddMt6IzWmwzKvT9oCQwZuuik8/0TdjNNc3kdD+c/
A0Q3pwIDAQAB
```

```
encryptWithRSA(aesKey, publicKey) {
  const rsa = new JSEncrypt({ defaultKeySize: 2048 });
  rsa.setPublicKey(publicKey);
  const encryptedAesKey = rsa.encrypt(aesKey);
  return encryptedAesKey;
}
```

步骤 3: 发送到服务器

API 接口:

```
POST https://api.youpin898.com/api/deviceW2
```

请求体：

```
{
  encryptedData: "...",       // AES 加密后的设备指纹数据
  encryptedAesKey: "..."      // RSA 加密后的 AES 密钥
}
```

核心代码：

```
async function sendData(encryptedData, generateKeys, request, env) {
  const aesKey = generateRandomString();
  const publicKey = "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA2IYU9R1Hrma...";
  const encryptedAesKey = encryptWithRSA(aesKey, publicKey);

  this.aesKey = aesKey;
  this.encryptedData = encryptedData;

  try {
    const response = await request.post("api/deviceW2", {
      encryptedData: encryptedData,
      encryptedAesKey: encryptedAesKey
    });

    // 解密响应
    const decryptedData = decryptData(response, aesKey);
    const parsedData = JSON.parse(decryptedData);

    const webUk = parsedData.u;
    const deviceUk = parsedData.deviceUk;

    // 存储到 localStorage
    localStorage.setItem("WEB_UK", webUk);
    localStorage.setItem("WEB_DEVICE_UK", deviceUk);

  } catch (error) {
    console.error("数据发送失败:", error);
  }
}
```

步骤 4: 服务器处理

服务器端操作（推测）：

1. **解密 AES 密钥**

   ```
   使用 RSA 私钥解密 encryptedAesKey
   ```

2. **解密设备数据**

   ```
   使用解密后的 AES 密钥解密 encryptedData
   ```

3. **生成唯一标识**

   ```
   根据设备指纹数据计算哈希或查询数据库
   生成/返回 WEB_UK 和 WEB_DEVICE_UK
   ```

4. **加密响应**

   ```
   使用 AES 密钥加密响应数据
   返回: { u: "WEB_UK", deviceUk: "WEB_DEVICE_UK" }
   ```

步骤 5: 客户端存储

```javascript
// 解密服务器响应
const decryptedResponse = decryptData(response, aesKey);
const data = JSON.parse(decryptedResponse);

// 提取两个标识符
const webUk = data.u;           // 用户唯一标识
const deviceUk = data.deviceUk; // 设备唯一标识

// 存储到 localStorage
localStorage.setItem("WEB_UK", webUk);
localStorage.setItem("WEB_DEVICE_UK", deviceUk);
```

存储位置:

- 键名: `WEB_UK` 和 `WEB_DEVICE_UK`
- 位置: 浏览器 localStorage (开发者工具 → Application → Local Storage)

---

4. 后续使用

4.1 初始化检查

```javascript
async function initUk() {
  const existingUk = localStorage.getItem("WEB_UK");

  if (!existingUk) {
    // 不存在则触发生成流程
    await generateUk();
  }

  return existingUk;
}
```

4.2 请求中携带

```
function getUk(options) {
  const isRefresh = options?.isRefresh;
  const existingUk = localStorage.getItem("WEB_UK") || "";

  if (!existingUk || isRefresh) {
    // 重新生成
    await this.sendData(...);
  }

  return localStorage.getItem("WEB_UK");
}
```

在后续 API 请求中，WEB_UK 会被附加到请求参数中：

```
extraParams.uk = localStorage.getItem("WEB_UK");
```

---

5. 技术要点总结

5.1 加密方案

加密类型	用途	算法
RSA	保护 AES 密钥传输	2048 位公钥加密
AES	加密设备指纹数据	对称加密（具体模式未明）

5.2 设备指纹技术

指纹类型	采集方法	唯一性
Canvas 指纹	绘制特定图形并转为 base64	高
字体指纹	枚举已安装字体	中
浏览器参数	UserAgent、分辨率等	低

5.3 存储机制

- **存储位置：** `localStorage`
- **持久性：** 长期有效（除非用户手动清除）
- **作用域：** 限定在 `youpin898.com` 域下

---

6. 代码位置

6.1 主要 JavaScript 文件

```
https://www.youpin898.com/umi.f036b7c3.js
```

6.2 关键函数位置（在混淆后的代码中）

函数名	功能
`getCanvasParams()`	生成 Canvas 指纹
`getFontsParams()`	获取字体信息
`getExtraParams()`	组装额外参数
`encryptWithRSA()`	RSA 加密
`encryptWithAES()`	AES 加密
`sendData()`	发送数据到服务器
`getUk()`	获取/生成 UK

6.3 API 接口

```
POST https://api.youpin898.com/api/deviceW2
```

请求格式：

```
{
  "encryptedData": "...",
  "encryptedAesKey": "..."
}
```

响应格式（解密后）：

```
{
  "u": "WEB_UK值",
  "deviceUk": "WEB_DEVICE_UK值"
}
```

---

7. 安全性分析

7.1 安全措施

✅ **传输加密：** 使用 RSA + AES 混合加密
✅ **HTTPS：** 所有通信通过 HTTPS
✅ **密钥随机：** 每次生成新的 AES 密钥
✅ **代码混淆：** JavaScript 代码经过混淆处理

7.2 潜在风险

⚠️ **设备指纹追踪：** 可能侵犯用户隐私
⚠️ **跨浏览器追踪：** 基于硬件特征可能实现跨浏览器追踪
⚠️ **持久化标识：** localStorage 长期保存，难以清除

---

8. 如何清除 WEB_UK

方法 1: 开发者工具

1. 打开浏览器开发者工具（F12）
2. 进入 **Application → Local Storage**
3. 选择 `https://www.youpin898.com`
4. 删除 `WEB_UK` 和 `WEB_DEVICE_UK` 键

方法 2: 控制台命令

```
localStorage.removeItem('WEB_UK');
localStorage.removeItem('WEB_DEVICE_UK');
```

方法 3: 清除浏览器数据

清除网站数据会同时删除所有 localStorage 内容。

---

9. 总结

WEB_UK 生成逻辑核心：

1. 📊 客户端收集设备指纹（Canvas、字体等）
2. 🔐 使用 AES + RSA 双重加密保护数据
3. 📡 发送到服务器 `/api/deviceW2`
4. 🎯 服务器生成唯一标识符 WEB_UK
5. 💾 存储到浏览器 localStorage
6. 🔄 后续请求中携带 UK 用于用户识别

应用场景：

- 用户身份识别
- 设备指纹追踪
- 反欺诈检测
- 用户行为分析

---

10. 附录

完整的 RSA 公钥

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA2IYU9R1HrmaTTIw9Bhd
hTnxoR4gXUcu2g26YVX+6UpgKsGUVDumUvvS67GxIPuRE9Y4L59uu7HkSVz61HZ
VIx+XqTANTH/sJf+eVmuaCnSgK+KZOKv64B9s1L0BXtpz3xpbKnjnmiRGDG0GKS
xImcQZVzLzEaZc58olo3zXrIEfUBLeZ8bCD66PTqPS5+s9pAulS4mj/OZFcnoxh
IR6h5L0XfZqdBaq6nQhYLCrrpKYmf4rzUWefSvS1GKk+DgofrmUat+aOboEKLGH
+02EndP1etmduJiEkSNXKzs1NddMt6IzWmwzKvT9oCQwZuuik8/0TdjNNc3kdD+
c/A0Q3pwIDAQAB
-----END PUBLIC KEY-----
```

localStorage 示例数据

```
{
  "WEB_UK": "5FTId5j7gApxnWHIB8E1OqJL3NaHupTTFst9YgAhlnOjHHHrgTFqU49kNykh8cX1K",
  "WEB_DEVICE_UK": "..."
}
```

文档生成时间： 2025-11-16
分析工具： Claude Code + Playwright Browser
网站版本： umi.f036b7c3.js