如何開發支持FIDO U2F 登錄的網站

Guo YK
光はいらね、水をください

前言

U2F (Universal 2nd Factor) 是Yubico, Yahoo 和Google 聯合開發的基於物理設備的雙因素認證協議，目前已經完成標準化，從屬於FIDO (Fast Identity Online) 聯盟名下。

特點

相較於其他雙因素驗證方案，U2F 有以下特點:

- 優勢
  - 相較於OTP (Google Authenticator, Authy, 短信驗證碼等)
    - 操作簡單，註冊和登錄均不需要輸入文字/掃描二維碼，只需要按一下設備上的按鈕
    - 安全性高，私鑰明文不會離開設備
  - 相較於其他基於物理設備的方案(各種U 盾)
    - 無需驅動/瀏覽器插件
- 劣勢
  - 需要購買硬件，Yubikey U2F 售價¥150 左右，U2FZero 物料費用$5 左右
  - 不兼容移動設備，只支持桌面瀏覽器(Chrome > 49，Opera > 42)

適用場景

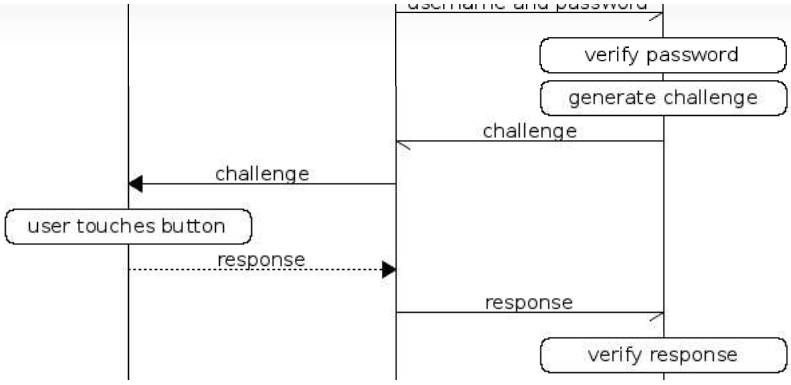
U2F 是嚴格基於物理設備的雙因素認證方案，相對於OTP，設備的交接和管理非常便利，適合**大型企業內部系統鑑權**（ERP，CRM 等）

此外，U2F 可以作為普通雙因素驗證方案的補充，為網站用戶提供更好的體驗（Google，Github, Dropbox，Docker Hub, Salesforce 等網站均已支持）

工作流程

U2F 安全性的核心在於不對稱加密算法，私鑰保存在設備上，簽名運算也在設備上執行，沒有任何手段可以獲取私鑰的明文。因此除非物理上獲取到了U2F 設備，否則是無法是無法破解U2F 認證流程的。

U2F 的工作流程和常見的不對稱加密算法認證體系類似，都是圍繞著“挑戰-響應” 展開的。



在硬件層面上，U2F 使用應用廣泛的HID 協議（鍵盤，鼠標等），支持USB、藍牙和NFC，確保在多種操作系統上，無需驅動，即插即用。

在瀏覽器層面上，Chrome 將HID 協議封裝成底層JavaScript API。

FIDO 官方提供了u2f-api.js 將瀏覽器底層API 封裝成高層API，通過一兩個調用，即可完成註冊和認證操作。

第三方封裝的高層API會有不同的實現，了解Yubico u2f-api.js 有助於了解U2F 開發的細節。

代碼實現

註冊

首先導入demo.yubico.com/js/u2f- ...，詳細的文檔可以在 這裡 查閱

典型的U2F 註冊流程，應該發生在用戶已經完成用戶名/密碼註冊之後，將U2F 設備綁定到一個用戶名下。

首先，後端生成隨機字符串，作為挑戰，記錄下來並發送到前端。

然後，前端使用u2f 對象，完成簽名

```
// AppId, 网站的 HTTPS 基地址
var appId = "https://demo.yubico.com";

// 构建参数
var params = {
 // 后端发送的随机字符串，作为挑战
 "challenge": "XXXXXXXXXXXXXXXX",
 // U2F 协议版本号，固定值
 "version": "U2F_V2"
};

// 调用 u2f.register
u2f.register(params.appId,
             [params],
             function(data) {
});
```

返回值data 是一個字典

返回值data 是一個字典。發生錯誤的情況下，data 只有errorCode 字段，定義如下（來自文檔）

```
    const short OTHER_ERROR = 1;
    const short BAD_REQUEST = 2;
    const short CONFIGURATION_UNSUPPORTED = 3;
    const short DEVICE_INELIGIBLE = 4;
    const short TIMEOUT = 5;
};
```

正常情況下，data 包含如下內容

```
{
  // 与服务器发起的 Challenge 内容相同
  challenge: "xxxxxxxxxxxxxxxxxxxxxxxxx",
  // 见下文
  clientData: "xxxxxxxxxxxxx",
  // 见下文
  registrationData: "xxxxxxxxxxxxxx",
  version: "U2F_V2"
}
```

clientData 為Base64 編碼後的JSON 字符串

```
{
  // 固定值， typ 没拼错
  typ: "navigator.id.finishEnrollment",
  challenge: "xxxxxxxxxxxxxxxxxxxxx",
  origin: "https://demo.yubico.com"
}
```

registrationData 為Base64 編碼後的二進制數據，內容按順序如下

- Head
  - 1 字節，固定值為0x05
- PubKey
  - 65 字節，應用證書公鑰，無壓縮P-256 NIST 橢圓曲線坐標數據
- PrivKeyHandle_Len
  - 1 字節，無符號整數，KeyHandle 的長度
- PrivKeyHandle
  - 長度由KeyHandle_Len 決定，私鑰句柄，見下文
- Main_PubKey
  - 長度不定，設備主證書公鑰，X.509 DER 二進制編碼的證書，同一批設備可能共用一個主證書
- Sig
  - 長度不定，簽名，使用SHA256-ECDSA (P-256 NIST) 算法

Sig 使用Main_PubKey 簽名，原始內容如下（拼接二進制數據）

- 1 字節固定值，0x00
- SHA256(AppId)
- SHA256(ClientData)
- PrivKeyHandle
- PubKey

最終，後端在驗證完簽名後，將PrivKeyHandle, PubKey 和Main_PubKey 記錄下來，並與用戶關聯，用於日後的驗證。

驗證

後端生成隨機數，作為挑戰，並記錄下來，然後和PrivKeyHandle 一起，發送到前端。

```
// Challenge, 后端生成的随机数
var challenge = "xxxxxxxxxxxxxxxx";
// 参数
var params = {
  // U2F 协议版本号，固定值
  "version": "U2F_V2",
  // PrivKeyHandle, 先前记录的私钥句柄
  "keyHandle": "XXXXXXX"
};
// 调用 u2f.sign
u2f.sign(appId, challenge, [params], function(data) {
});
```

和註冊類似，在失敗的時候，data 包含一個errorCode

成功的時候，data 包含如下內容

```
{
  // PrivKeyHandle, Base64 编码
  "keyHandle":"xxxxxxxxxxx",
  // 见下文
  "clientData":"xxxxxxxxxxx",
  // 见下文
  "signatureData":"xxxxxxxxxx"
}
```

其中，clientData 字段為Base64 編碼後的JSON

```
{
  // 固定值，typ 没拼错
  typ: "navigator.id.getAssertion",
  // 先前服务器发起的 Challenge
  challenge: "xxxxxxxxxxxxxxxxxxxxx",
  origin: "https://demo.yubico.com"
}
```

signatureData 字段為Base64 編碼後的二進制數據，內容按順序如下:

- Flag
  - 1 字節，第0 比特位表示認證是否成功
- Counter
  - 4 字節, Big-Endian 無符號整數(UInt32)，簽名計數器
- Sig
  - 長度不定，SHA256-ECDSA (P-256 NIST) 簽名

Sig 使用PubKey 簽名，原始數據如下（拼接二進制數據）：

- SHA256(AppId)
- Flag
- Counter
- SHA256(ClientData)

最終，服務器在驗證完簽名後，認可用戶的身份，執行下一步操作。

在整個流程中，為了防止設備在不同的網站間追踪，U2F 設備會為不同的網站生成不同的密鑰對。為了保證單台U2F 設備支持無限多的網站登錄，密鑰對中的私鑰保存在U2F 設備上是不可能的，因為芯片容量有限，且非常珍貴，因此才有了PrivKeyHandle 這一參數。PrivKeyHandle 是用來讓U2F 設備“回想”起私鑰的，而具體的內部實現方式，由各廠商自己決定。

取。註冊時生成的私
，服務器把

而Yubikey 和U2FZero （以及其他廠商） 使用了一個更加複雜的方案，私鑰由隨機數、AppId、設備主密碼經過複雜的算法派生出來，PrivKeyHandle 中只包含一個MAC (Message Authentication Code） 和隨機數，保證私鑰不會離開設備。

參考資料:

Yubico's Take on U2F Key Wrapping | Yubico

Yubico/java-u2flib-server: Java server-side library for U2F

fidoalliance.org/specs/ ...

我的博客鏈接:如何開發支持U2F 的網站

編輯於2017-02-23 18:19

網絡安全 RSA 加密 互聯網

7 條評論 默認 最新

Bananananana
進行這部分的開發測試，必須要買一個yubikey麼?
2020-03-05 回复 贊

一隻被貓俘獲的狼
樓主可以試試HyperFIDO Key 很好用又便宜性價比非常高
2019-06-13 回复 贊

洛小白
為啥業內鑑權都用otp很少用u2f?
2018-06-05 回复 贊

馬前炮
請問Yubikey在哪裡買的?
2017-02-24 回复 贊

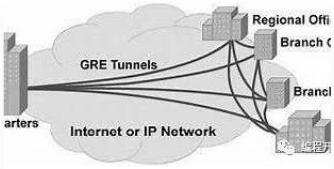
Guo YK 作者
淘寶
2017-02-24 回复 贊

喵嗚要給力
請問支持FIDO UAF的登錄的網站或者客戶端容易實現麼
2017-02-23 回复 贊

Guo YK 作者
UAF 我沒有看，有機會我也研究一下。寫U2F 純粹是因為手上有一個YubiKey。
2017-02-23 回复 1

LandØ
以程序的視角看待世界，解構世界

推薦閱讀



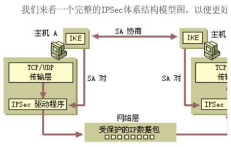
Docker搭建簡易VPN(支持IKEv2、IPsec協議)

青衫大叔,,,,

什麼是以太坊的域名服務ENS

ENS 問世以來，有很多用戶競相註冊與續費，但也有很多用戶還不知道什麼是ENS，該如何操作。以下通過詳細的步驟教你如何註冊屬於自己的以太坊域名。什麼是以太坊的域名服務ENS？看到ENS，...

Tokenview



網絡安全之IPSec vpn

Mr.Six666