WORDCAMP US 2026

The Attacker That Never Logged In

Session Hijacking, Stolen Cookies, and the Blind Spot in WordPress Security

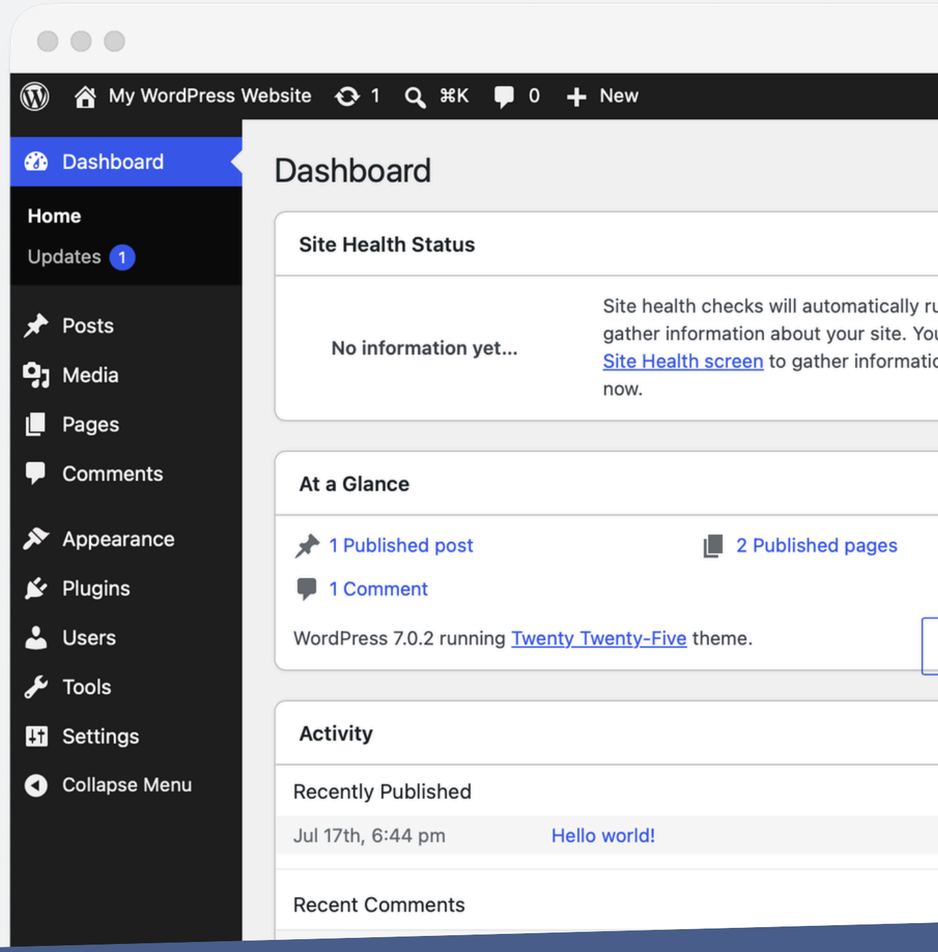
Robert Abela

Salted, hashed, and delivered

HOW SESSIONS WORK

An attacker has access to your WordPress dashboard without:

- Guessing your password
- Breaking your 2FA / MFA
- Bypassing your firewall / other security solution



HOW SESSIONS WORK

“Cybercriminals crave cookies (& sessions), not passwords.” To re-use your session!

- By stealing it(cookies,tokens)
- By riding it (tricking your browser)

HOW SESSIONS GET HIJACKED

“8.6bn

stolen session cookies in 2025”

— SpyCloud *2026 Identity Exposure Report*

“276 million credentials with active session cookies usable to bypass MFA”

— *Recorded Future 2025 Identity Threat Landscape Report*

“Identity threats accounted for 53% of the total detection volume in 2025, up from 20% in 2024”

— *Red Canary 2026 Threat Detection Report*

HOW SESSIONS WORK

Cookies are credentials

- Authentication cookies stored in the browser
- Sent with every request
- Whoever has the cookie or can make it fire is the logged in user

WordPress sessions & cookies are not the problem

Also used in:

- Most web applications
- SaaS solutions
- Online services

What we'll cover

01 How WordPress authentication works

02 How sessions are stored

03 How sessions work in WordPress, and the risks

04 How sessions are hijacked

05 Protecting sessions and improving visibility

HOW SESSIONS WORK

Once you authenticate WordPress creates:

- Session token
  - Stored in the database
- 3 authentication cookies
  - Stored in browser

HOW SESSIONS WORK

Authentication & session cookies:

wordpress_[hash]

wordpress_sec_[hash]

wordpress_logged_in_[hash]

- Username (theidentifier)
- Expiration time
- Session ID (token)
- HMAC (security signature)

Cookie:

wordpress_logged_in_4c925661ace6df39a0ee1fa3e5244603=robert%7C1782395732%7CDEhDmvn3eeSV1GA5SWMSvcQ9Rhdp46Aky0rG2PjHpch%7C07a222f2bb50979ed366a4b9d8019cd50e722b57cla15f6b7c6bfe444439bed0;

Cookie hash in name: 4c925661ace6df39a0ee1fa3e5244603
Username: robert
Expiration (Unix time): 1782395732
Session token: DEhDmvn3eeSV1GA5SWMSvcQ9Rhdp46Aky0rG2PjHpch
HMAC: 07a222f2bb50979ed366a4b9d8019cd50e722b57cla15f6b7c6bfe444439bed0

HOW SESSIONS WORK

Session tokens in wp_usermeta

- One user session
- One browser or device

Edit: wp_usermeta

umeta_id		86
user_id		1
meta_key		session_tokens
meta_value		a:2: {s:64:"576c95906cf126c9260e7cac47ea97fe2f4d1da82b3 675571997ee495b7eaf1e";a:4: {s:10:"expiration";i:1784104519;s:2:"ip";s:10:"80. 57.82.7";s:2:"ua";s:117:"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/150.0.0.0 Safari/537.36";s:5:"login";i:1783931719;}s:64:"0eb 60dca3c300d07df97951e73f6d6f47697c8965e610eac58782 6ee0a57d847";a:4: {s:10:"expiration";i:1784116733;s:2:"ip";s:10:"80. 57.82.7";s:2:"ua";s:117:"Mozilla/5.0 (Macintosh;

Session token breakdown:

Hashed session token: 714f7d5fae78ac67d16a9efbd03289a44003df9c1701f391a168c9f43c31ff8b

Expiration: 1783632456

IP: 6.6.19.81

UA: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/149.0.0.0 Safari/537.36

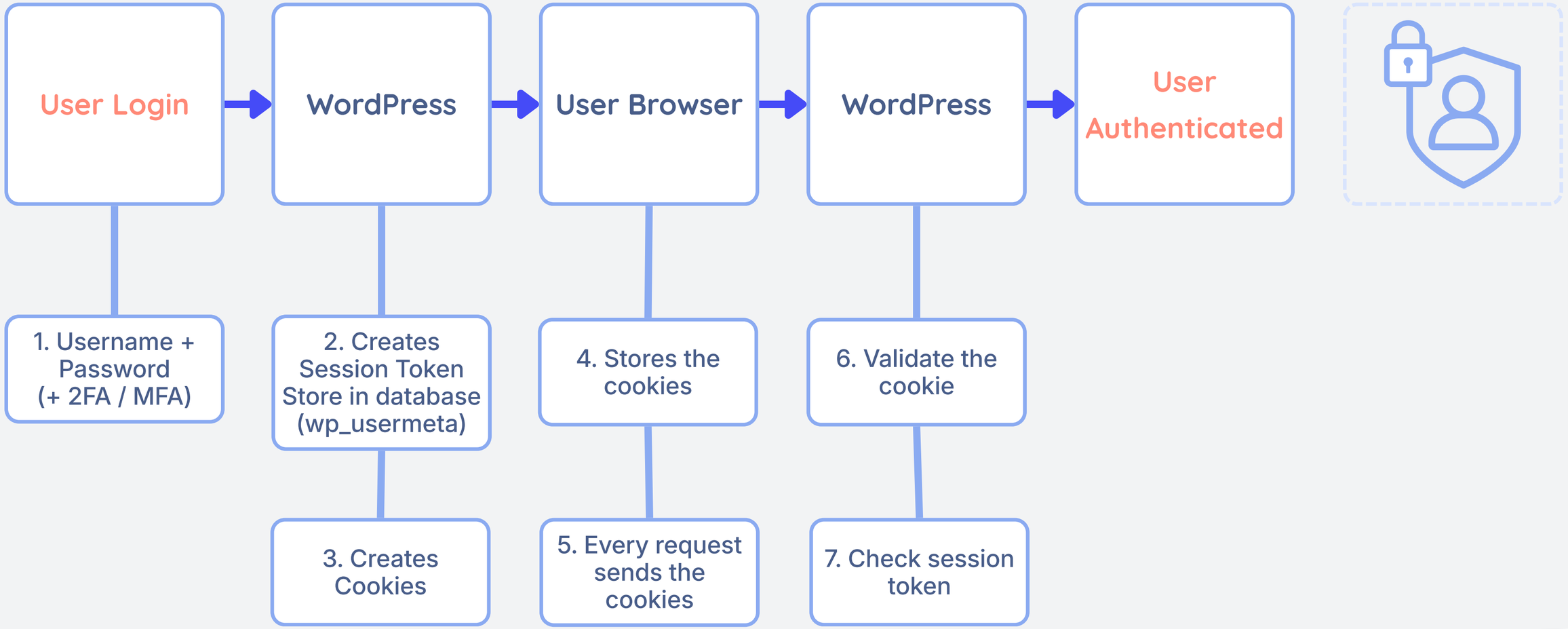
login: 1783589493

HOW SESSIONS WORK

What Happens on Every Request

Salted, hashed, and delivered

```
GET /wp-admin/edit.php?post_type=page HTTP/2

Host: test.melapress.com

Cookie:

wordpress_sec_f5003eacb230a787778c40d7aedd3047=jacixisigi3944%7C1784116733%7CA
bOSo3OOVC8zKrIfRRxvpftfYRdduUi2YIHv1XE4Tfk%7Cc92e8768fd1c5464d132ec9eea99e4073
2906bd8902029b52fa0640fe30ffdd3;

wordpress_test_cookie=WP%20Cookie%20check;

wordpress_logged_in_f5003eacb230a787778c40d7aedd3047=jacixisigi3944%7C17841167
33%7CAbOSo3OOVC8zKrIfRRxvpftfYRdduUi2YIHv1XE4Tfk%7Ccae29fb0e5cc13f0fcc179d63ff
d853455ab4a8b3589199195b8c80bf740ebc5;

wp-settings-time-1=1783943934


Sec-Ch-Ua: "Not-A.Brand";v="24", "Chromium";v="146"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "macOS"
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/146.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://test.melapress.com/wp-admin/
Accept-Encoding: gzip, deflate, br
Priority: u=0, i
```


User Login
WordPress
User Browser
WordPress
User Authenticated
1. Username + Password (+ 2FA / MFA)
2. Creates Session Token Store in database (wp_usermeta)
3. Creates Cookies
4. Stores the cookies
5. Every request sends the cookies
6. Validate the cookie
7. Check session token

HOW SESSIONS WORK

Users can have multiple sessions at the same time

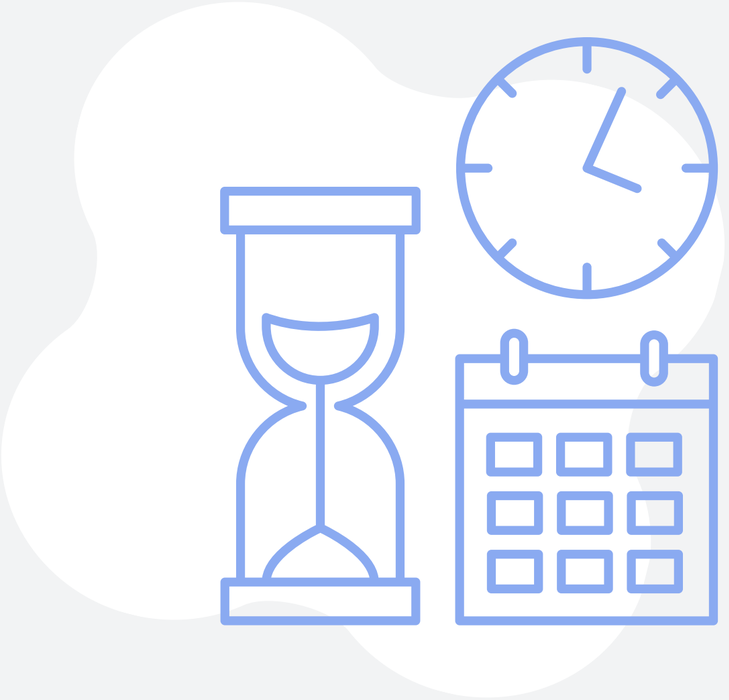
- Laptop
- Phone
- Work computer
- Multiple browsers

HOW SESSIONS WORK

Sessions expire only when

01 Expiration is reached (48 hours or 14 days)

02 User logs out



HOW SESSIONS WORK

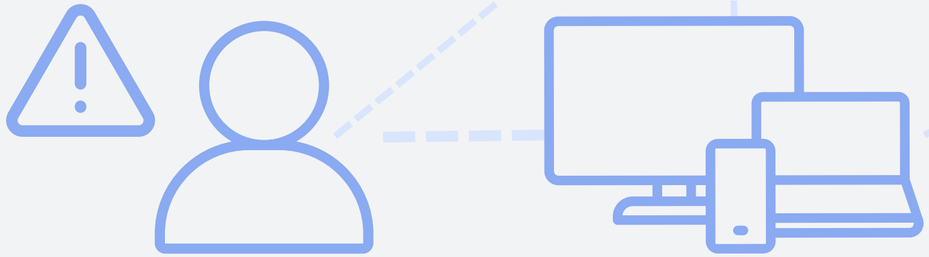
Every session is:

- An authenticated identity
- An access point
- A possible entry door

How can attackers steal, compromise, & hijack sessions?

HOW SESSIONS GET HIJACKED

Shared accounts:



- Shared (admin) accounts
- Lack of security hygiene
  - unlocked laptops
- Logging in on public computers
- Logging in on others' devices
- Many devices logged in

HOW SESSIONS GET HIJACKED

Compromised browsers & stolen devices:

- Malware
- Malicious (browser) extensions
- Browser profile theft
- Stolen laptops

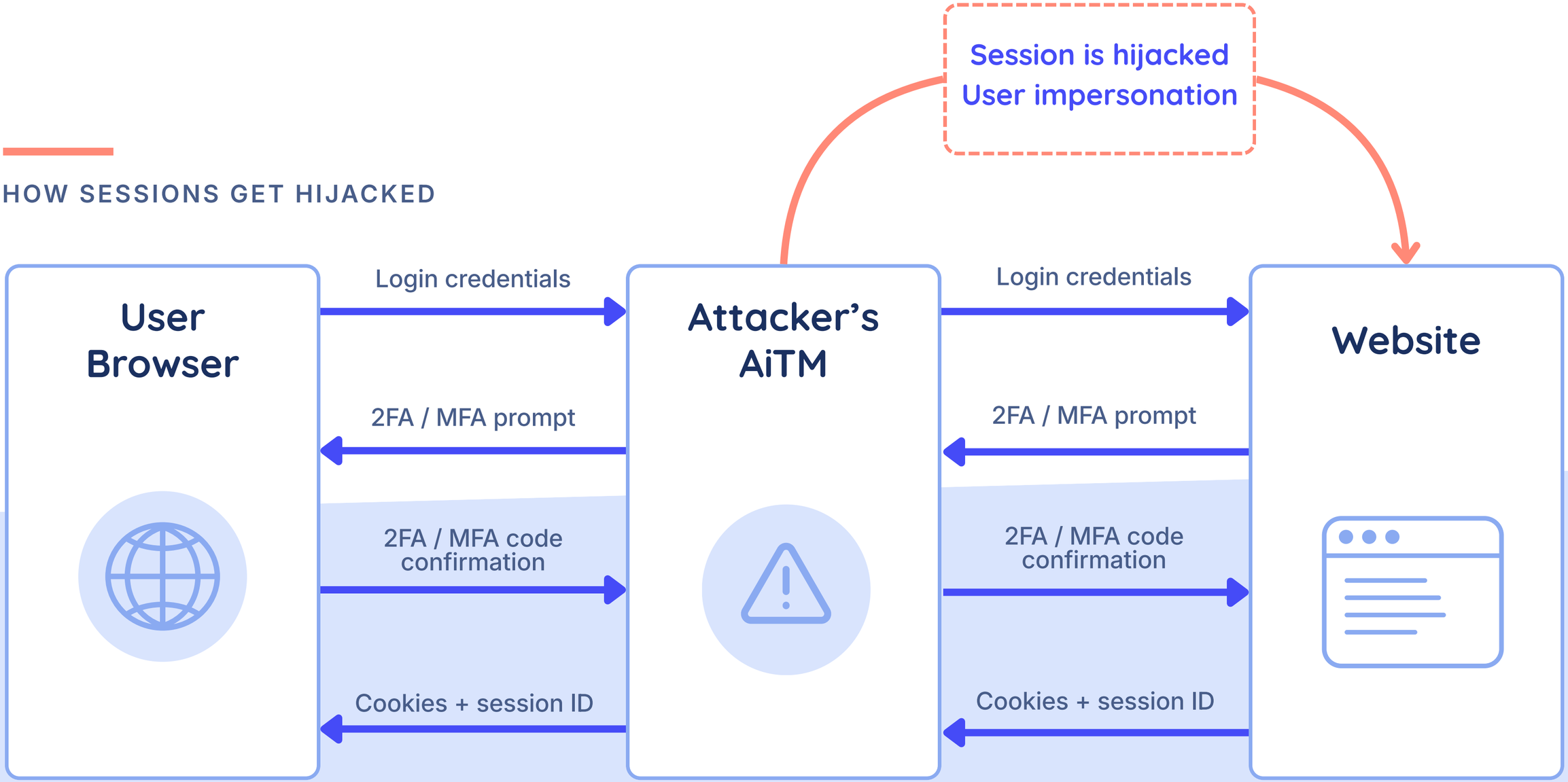
Dangers of public Wi-Fi & networks:

- Packet sniffing
- Evil twin Wi-Fi hotspot

Phishing for Sessions

- Phishing attacks (stealing credentials)
- Adversary-in-the-middle phishing attacks
  - Fake login page (reverse proxy)
  - Victim logs in (with MFA)
  - Attacker captures the session cookie

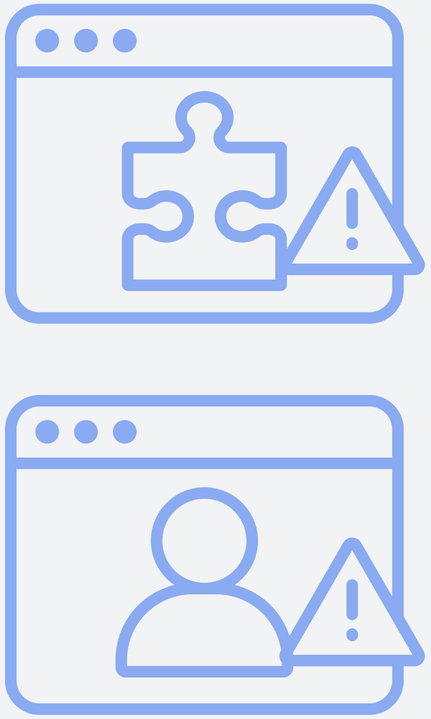
HOW SESSIONS GET HIJACKED



HOW SESSIONS GET HIJACKED

Outdated & exploitable software

- Known vulnerabilities in software
- Exploit code often publicly available

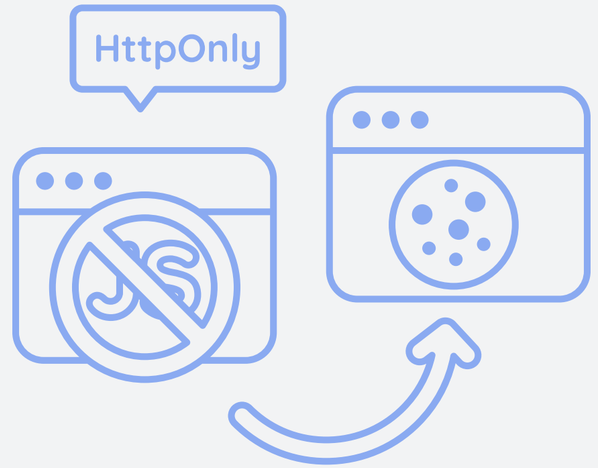


“By exploiting an XSS, attackers can steal your cookies”

WordPress sets the authentication cookies as HttpOnly

- JavaScript cannot read the cookie value
- document.cookie returns nothing

However, the cookie still travels with every request...



HOW SESSIONS GET HIJACKED

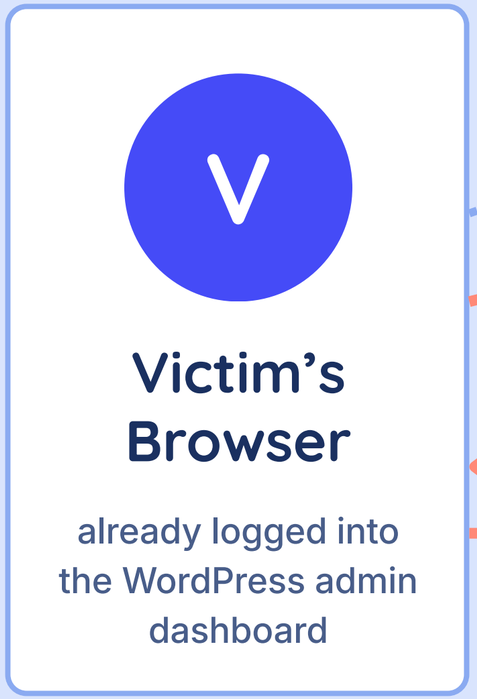
CSRF & Session riding

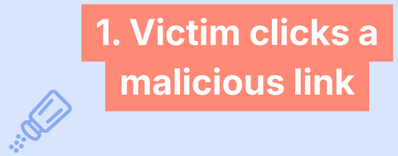
the attacker never touches the cookie but the attack uses them

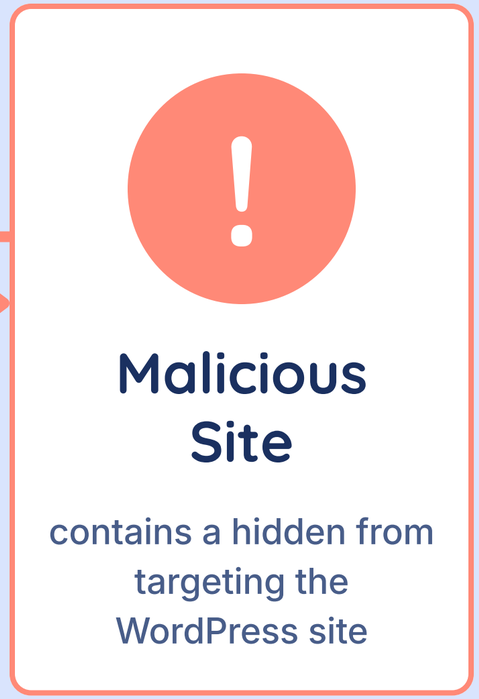
- Victim (admin) is tricked into visiting a malicious page
- Page silently submits a request to the WordPress site
- The admin's browser sends the cookies automatically
- Result: new administrator account created by the admin

Cross-Site Request Forgery (CSRF)

How a malicious link can trick your browser into sending a request you never intended







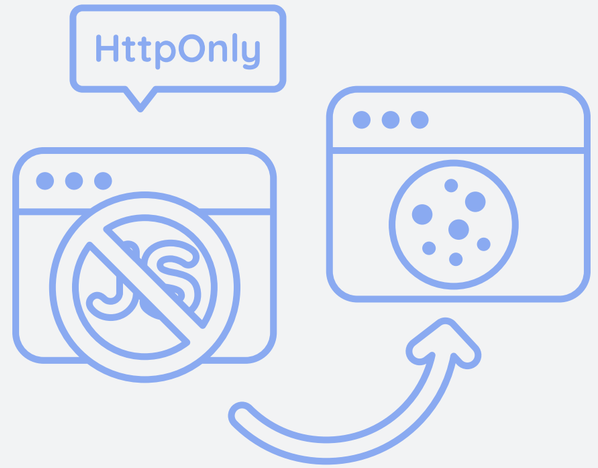
Results
WordPress creates a new admin user - believing the victim requested it on purpose. The attacker now has full site access.

Salted, hashed, and delivered

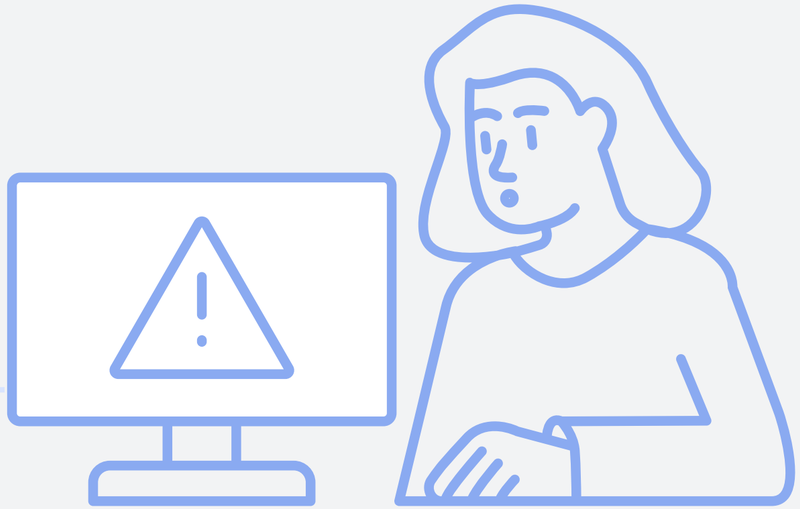
HOW SESSIONS GET HIJACKED

A chain of exploits doesn't need to break the cookie jar

- Vulnerable plugin/theme leads to server-side flaws
- Session token retrieved server-side (not via the browser)
- HttpOnly never enters the picture



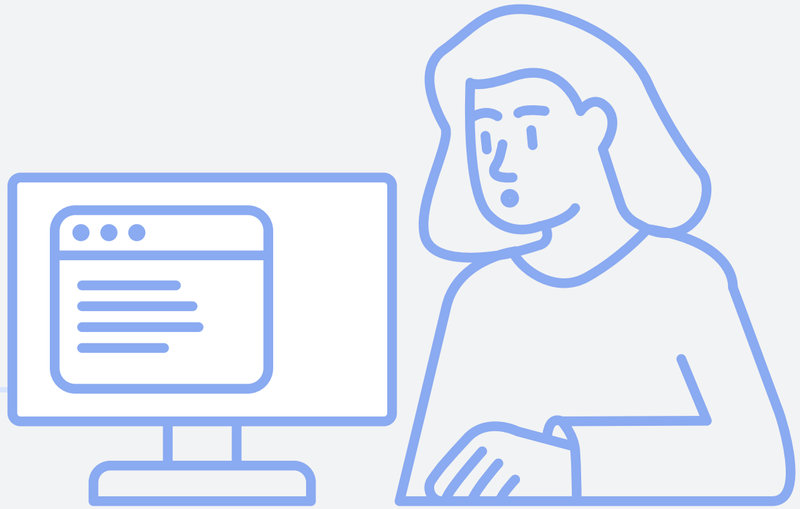
Holistic security



- HttpOnly, encrypted cookies, firewalls, 2FA.
- Each one blocks a piece of the attack.
- None blocks all of it.
- Security is layers, not a switch you flip.

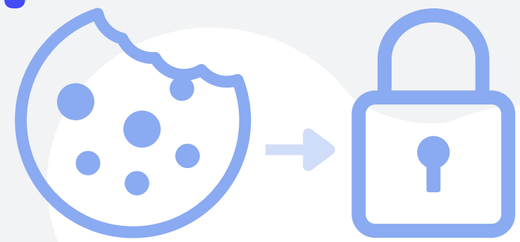
HOW SESSIONS GET HIJACKED

The weakest link...



- "Hey, can you check this real quick"
- Urgency + trust beats technical skill every time
- Targets aren't always admins
- One click, on one device, on one bad day

Why these type of attacks?
What is the impact?



Admin session hijacked?

A hijacked non-admin user account?

- Reconnaissance
- Social engineering
- Privilege escalation
- Internal visibility
- Vulnerability chaining

Hijacked accounts go unnoticed because:

- No login alert
- No 2FA / MFA triggered
- Session remains valid
- No visibility of active sessions
- No activity logs
- No controls and checks

Focus on hardening is VERY important:

- Credentials hygiene
- 2FA / MFA / Passkeys
- Brute force protection
- Firewall & Virtual Patching
- Patch management / keep everything up to date

POST-LOGIN SOLUTIONS

Post login security & monitoring

- Active sessions
- Login origins
- Devices
- Session lifetime
- Unusual session activity

POST-LOGIN SOLUTIONS

Activity logs allow you to:

- Keep an eye on user activity
- Do forensic work
- Notifications for specific user actions

POST-LOGIN SOLUTIONS

User sessions management & policies

- See who is logged in real-time
- Limit concurrent user sessions
- Implement inactivity timeout
- Shorter default session duration

POST-LOGIN SOLUTIONS

User accounts management

- Delete inactive users
- Use temporary / time limited logins for 3rd parties
- Add security questions / 2FA for sensitive actions
- Restrict user login times
- Apply Principle of least privileges for user roles

POST-LOGIN SOLUTIONS

Device recognition:

- Limit IP addresses/machine combinations
- Device recognition alerts
- Improve user notifications

POST-LOGIN SOLUTIONS

Technical fixes only work if the human layer is covered:

- Recognize possible attacks
- Encourage a reporting culture
  - "I think I clicked something" should be safe to say
- Simulated phishing & other tests like fire drills

POST-LOGIN SOLUTIONS

Follow best practises:

01 Credentials hygiene

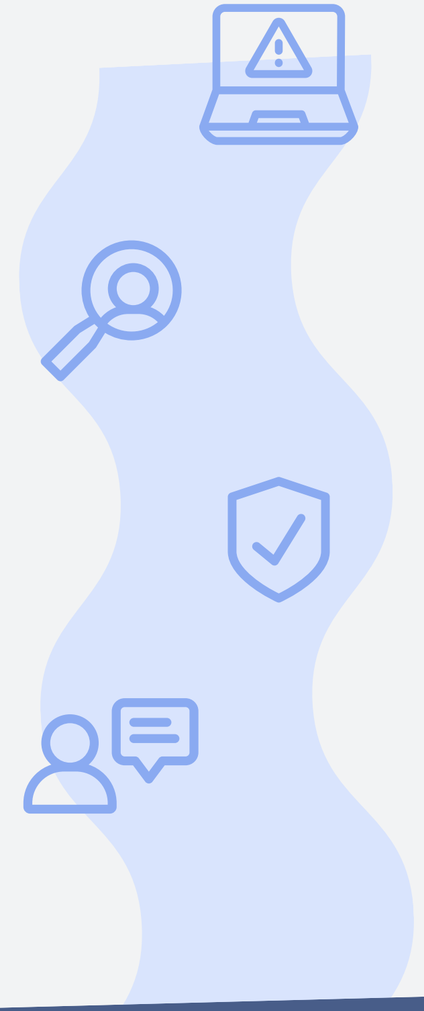
02 Use VPN

03 Only use assigned devices

04 Keep everything up to date

POST-LOGIN SOLUTIONS

To recap:



- User sessions are an attack surface
- User session visibility is critical
- Improve security:
  - Session management & policies
  - Website users / account management
  - Devices recognition & user notifications
  - Keep users informed
- User education goes a long way

STAY IN TOUCH

Thank You

Robert Abela

CEO & Founder

melapress

WordCamp US 26

Salted, hashed, and delivered