Jakub Čermák
Software fox, HAVIT, s.r.o.
cermak@havit.cz , jakub@jcermak.cz
Twitter @cermakj

Bezpečnost webových aplikací v ASP.NET

Agenda

- Obecné zásady, kryptografická primitiva
- Běžné útoky a chyby
  - A8 CSRF
  - Overposting
  - A3 XSS, CSP

Obecné zásady

- Nikdy nevěřte klientovi či klientské validaci
  - Hidden field, <option>, …
- Nikdy security by obscurity

- Používejte hotová kryptografické primitiva
  - Útoky na implementaci

Hotová kryptografická primitiva

- Běžné věci přímo v .NETu
- Jaké zvolit parametry?
  - -> SecurityDriven.NET Inferno

Hashujeme

- Kontrola integrity dat proti poškození
- Rychlé porovnávání velkých dat
- Hesla
  - Ne MD5, ne SHA1
  - SHA-2
  - PBDKF2 / Bcrypt / scrypt
  - Solíme
    - Kryprograficky silná náhodná čísla
      - Ne Random
    - Dlouhé, unikátní

DEMO: hashujeme hesla

Hashujeme s klíčem

- HMAC, symetrický podpis

- Zabezpečení přes nezabezpečený kanál
- Přihlašovací tokeny, ověřovací kódy, resety hesla, …

DEMO: hmac

Šifrujeme symetricky

- Šifrování a dešifrování pomocí stejného klíče
- AES, Blowfish

- Zabezpečení externího úložiště dat aplikace
  - Azure Storage

Šifrujeme asymetricky

- Veřejný klíč – pro šifrování
- Soukromý klíč – pro dešifrování
- RSA

- Posílání tajných dat po nezabezpečeném kanále

- Kombinace se symetrickou šifrou – asymetricky šifruji **náhodný** klíč pro symetrickou šifru
  - Rychlost
  - Více klíčů – více konzumentů (šifrování NTFS)

DEMO: šifrování

Nejčastější závažné chyby aplikací

- OWASP top 10

Cross site request forgery (OWASP A8)

- Naše aplikace umí mazat uživatele pomocí požadavku na /Admin/DeleteUser/
- Útočník donutí oběť na tuto URL přistoupit (zpráva na foru, ajax, iframe, ...) => oběť nevědomky smaže uživatele

- Metody obrany
  - Antiforgery token
  - Žádný GET

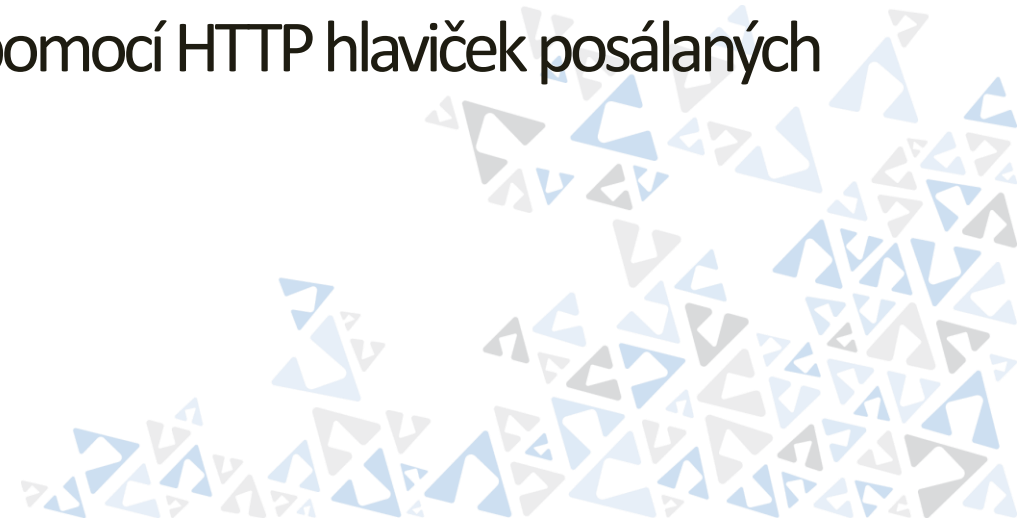
DEMO: csrf + antiforgery

X-Frame-Options

- Hlavička odpovědi omezující použití zdroje jako zdroj pro <iframe>
- Znají ji browsery a dle toho nezobrazí iframe
- Možné vlastnosti:
  - DENY - nikdy nepovolit
  - SAMEORIGIN – jen ze stejné domény
  - ALLOW-FROM uri – explicitní povolení vypsaných URI

Cross-origin resource sharing

- Omezení přístupu k REST službám / zdrojům
- Př. Zdroj example.com/api/DeleteUser – AJAX rest api
- Mám aplikaci na myapp.com doméně
- Z aplikace chci AJAXem přistupovat na DeleteUser
- Výchozí NELZE – je na jiné doméně
- Je třeba explicitně povolit pomocí HTTP hlaviček posálaných cílovým zdrojem



CORS – posílané hlavičky

- Zjednodušený (GET-only, bez cookie):
- Origin: http//www.myapp.com (posílá browser)
- Access-Control-Allow-Origin: http://www.myapp.cz

- „Preflight check“ – předpožadavek OPTION na zdroj zjišťující oprávnění
- Access-Control-Allow-Origin: http://foo.com
- Access-Control-Allow-Methods: PUT, DELETE
- Access-Control-Allow-Credentials: true

DEMO: cors na CorsResource

CORS – podpora ve web api

- Nuget Microsoft.AspNet.Cors
- Konfigurace
  - config.EnableCors();
- Na controlleru
  - [EnableCors(origins: "http://app.cz", headers: "*", methods: "*")]

MVC Over-posting

- MVC binder vyplní všechny propertíes na přijatém modelu (nebo TryUpdateModel), které dostane i když jsme nedělali pro ně input
- Registrace uživatele, model = entita z DB, má vlastnost IsAdmin
- Útočník ručně přidá hidden input s hodnotou IsAdmin=true => proběhne binding => máme nového admina ☺

MVC Over-posting

- Nebindovat entity, ale ViewModely
  - Pro každou akci mít vlastní, bez závadných properties
  - V UI fasádě převést viewmodely na entity / business obj.
- Bindovat interface
  - TryUpdateModel<IUserInputModel>(user);
- Bind atribut na parametru
  - ViewResult Edit([Bind(Include = "FirstName")] User user)
  - ViewResult Edit([Bind(Exclude = "IsAdmin")] User user)
- ReadOnly atribut na property bindovaného modelu

DEMO: overposting registrace

XSS

- Typ injection; vkládání zlých věci do hodné stránky

- Vzdálené spouštění skriptů
- krádeže dat či identity
- CSRF

XSS (2)

- Persistentní (stored)
  - Ukládané do db
  - Typicky formuláře, fora, HTTP hlavičky při logování
- Nepersistentní (reflected)
  - Součástí URL
  - V mailech, na forech
  - Provede přesměrování - phishing, načte zlý skript

DEMO: XSS

XSS – hlídání vstupů

- Neřešit – admin stránky
- Zakázat
- Vlastní značovací jazyk – BBCode
- Sanitize
  - https://github.com/mganss/HtmlSanitizer
  - **Whitelisting tagů a atributů !**
    - Classjacking
    - Temná zákoutí html - <button form="test" formaction="javascript:XXX">X</button>
- **hlídat URL, cesty k souborům**
  - **http://srv.cz/downloadDoc.ashx?name=../Website/web.config**

Content Security Policy

- Omezení zdrojů přidruženého obsahu pomocí whitelistingu URL
  - Zákaz obrázků z nepovolených domén
  - Zákaz skriptů či inline skriptů
  - .....

- HTTP Hlavička
- **Content-Security-Policy:** default-src ‘self‘ cdn.moje.cz https://*.goodsite.cz ; script-src ‘none‘

CSP (2) - direktivy

- Child-src – iframe
- Connect-src – AJAX
- Font-src – fonty
- Form-action – omezení URL formulářů (experimental)
- Frame-ancestors – X-Frame-Options
- Img-src
- Media-src – audio, video
- Object-src -

CSP (3) - zdroje

- Cdn.domain.cz, *.domain.cz, https://xy.cz
- * - vše
- ‚none' – nic
- ‚self' – jen aktuální doména
- Data: - povolení data-uri
- https: - jen pomocí šifrovaného spojení
- ‚unsafe-inline'
- ‚unsafe-eval'

CSP - nonce

- Explicitní povolení konkrétních inline skriptů/stylů/...

- Uvedení sha256 vkládaného inline skriptu
- Uvedení nonce
  - **Content-Security-Policy:** default-src ‚self'; **script-src** ‘self‘ **'nonce-RAND‘**,
  - `<script type="text/javascript" nonce=„RAND">`
    - `alert('inline skript test');`
  - `</script>`

DEMO: nonce

CSP - reporting

- Direktiva report-uri
- Bonzuje všechna porušení CSP

- Content-Security-Policy-Report-Only
  - Pravidla nanečisto
  - Pouze reporting, neblokuje

DEMO: reporting

Insecure Direct Object References

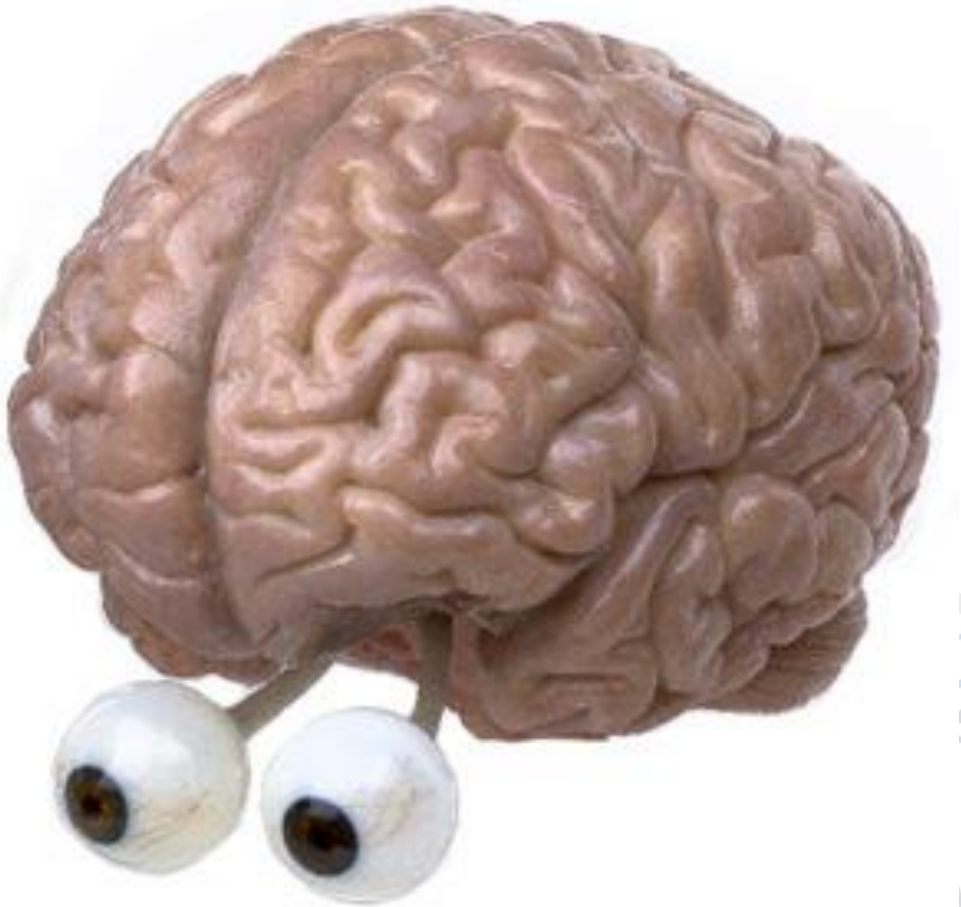
- Častá chyba
- Skrytí tlačítka, neověření přístupu na konkrétní stránku

- Missing Function Level Access Control
  - Varianta samého

OWASP Top 10 - nemusíme řešit

- A1 (SQL) Injection
  - Vždy používat SQL parametry nebo ORM
- A2 Broken Auth and Session Management
  - Vestavěné řešení ve frameworku
- A5 Security Misconfiguration
  - IIS a Win Server dosti restriktivní by default
- A6 Sensitive Data Exposure
  - Web.config chráněn by default, vše mít tam
  - Hashovat hesla, pozor na okolní kanály – zálohy, vývojové odlití produkční db
- A9 Using components with Known Vulnerabilities
  - Nuget

Tajná zbraň



Závěr

- Obecné zásady, kryptografická primitiva
- Běžné útoky a chyby
  - A8 CSRF
  - Overposting
  - A3 XSS, CSP

- Kontakt: Jakub Čermák; jakub@jcermak.cz