Statická **kópia webu**

Návod na vytvorenie a nasadenie statickej verzie webového sídla. Postup je univerzálny pre ľubovoľný redakčný systém, ako ukážkový príklad slúži WordPress.

OBSAH

Úvod

Tento dokument popisuje, ako z webového sídla postaveného na redakčnom systéme (CMS) vytvoriť **statickú kópiu**, teda sadu obyčajných HTML, CSS a JavaScript súborov, a nasadiť ju na verejný webový server.

Statická kópia neobsahuje žiadny serverový kód (PHP, databázu, administračné rozhranie). Verejný server tak zverejňuje iba súbory, čím sa dramaticky zmenšuje útočná plocha: odpadajú zraniteľnosti CMS, zásuvných modulov, tém aj databázy. Redakčný systém funguje ďalej, ale iba vo vnútornej sieti alebo lokálne, verejnosť sa k nemu nikdy nedostane.

Návod je určený správcom webových sídiel a IT administrátorom. Predpokladá základnú znalosť práce v príkazovom riadku systému Linux.

Ako návod používať

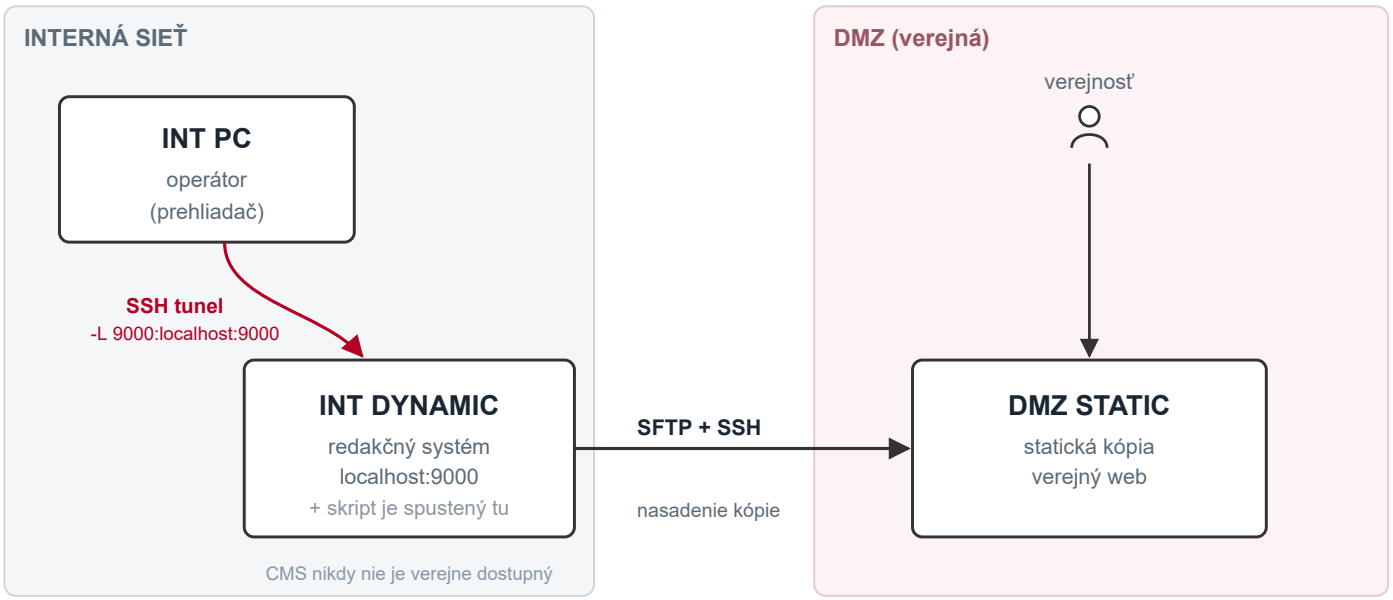
Samotný postup je **univerzálny**. Funguje pre ľubovoľný CMS, ktorý generuje stránky na serveri (WordPress, Joomla, Drupal, Typo3 a podobne), pretože web sa kopíruje tak, ako ho vidí návštevník. Na miestach, kde sú potrebné údaje špecifické pre konkrétny web, návod popisuje postup, ako si tieto údaje zistiť, a ako ukážku uvádza hodnoty z inštalácie WordPress.

Celý proces má podobu opakovanej slučky: **stiahni** → **over** → **doplň** → **zopakuj**. Prvý prechod takmer nikdy nie je dokonalý, práve lokálne overenie (sekcia 7) odhalí, čo treba doplniť v sekciách 5 a 6. Po odladení sa celý postup zautomatizuje skriptom (sekcia 9) a ďalšie aktualizácie webu sú otázkou jedného príkazu.

Architektúra a bezpečnostný model

Zmysel celého postupu je bezpečnostný: redakčný systém so všetkou jeho útočnou plochou zostáva v internej sieti a verejnosť nikdy nekomunikuje priamo s ním, ale iba so statickou kópiou v DMZ. Postup preto pracuje s troma prostrediami:

- **INT PC** - pracovná stanica operátora v internej sieti. Odtiaľto sa cez SSH tunel prezerá a overuje redakčný systém.
- **INT DYNAMIC** - interný server s redakčným systémom, dostupným iba na slučke (`localhost:9000`), teda nikdy nie zo siete. **Na tomto serveri je spustený aj exportný skript.**
- **DMZ STATIC** - verejný server v DMZ, ktorý servíruje výslednú statickú kópiu. Jediné, čo je dostupné z internetu.



Tok: operátor cez SSH tunel (`-L 9000:localhost:9000`) prezerá redakčný systém na INT DYNAMIC; exportný skript je spustený na INT DYNAMIC a výslednú kópiu nasadí cez SFTP na DMZ STATIC; verejnosť pristupuje výhradne k DMZ STATIC.

Redakčný systém je na INT DYNAMIC naviazaný len na slučku (`127.0.0.1:9000`), takže nie je dostupný cez sieť. Dosiahnuteľný je dvoma spôsobmi: **exportný skript** je spustený priamo na INT DYNAMIC a pristupuje k nemu natívne cez `localhost:9000` ; **operátor** z INT PC otvorí SSH tunel a v prehliadači na svojom počítači tak vidí ten istý web. To je dôvod, prečo adresa `localhost:9000` funguje v oboch prípadoch.

```
# z INT PC: sprístupní redakčný systém z INT DYNAMIC na vlastnom porte 9000
ssh -L 9000:localhost:9000 uzivatel@int-dynamic
# potom v prehliadači na INT PC: http://localhost:9000
```

📋 **SSH TUNEL = LOKÁLNE PRESMEROVANIE PORTU**

Prepínač `-L 9000:localhost:9000` presmeruje port 9000 na vašom počítači na port 9000 tak, ako ho vidí vzdialený server (jeho slučka). Ide o *lokálne presmerovanie portu* - umožňuje pristupovať k službe, ktorá je zámerne dostupná len interne, bez toho, aby sa kdekoľvek vystavila do siete.

Pred začatím

⚠️ **PRACUJTE BEZPEČNE**

Redakčný systém, ktorý prehľadávate, je zdrojom obsahu. Práve jeho statickú kópiu zverejňujete. Podstatné je, aby táto inštancia **nebola počas prehľadávania vystavená verejnej prevádzke**: ideálne je dostupná iba vo vnútornej sieti (ako v tomto postupe, kde verejnosť vidí len statickú kópiu) alebo ako oddelená testovacia inštancia. Keďže skript používa `rm -rf` na pracovný adresár a `rsync --delete` na cieľový server, **pred prvým spustením si vytvorte zálohu** redakčného systému aj cieľového servera.

Čo budete potrebovať

- Funkčnú lokálnu inštanciu webu (v ukážke WordPress na `localhost:9000`)
- Nástroje `wget`, `curl`, `rsync`
- SSH prístup (ideálne s kľúčom) na cieľový statický server
- Cestu k súborom CMS na disku, a oprávnenie na ich čítanie (v ukážke `/var/www/html`)
- Verejnú doménu webu (v ukážke `https://csirt.sk`)

> ⚠️ **ZAPNITE „PEKNÉ" URL ADRESY (PERMALINKS)**
>
> V redakčnom systéme nastavte adresy stránok na podobu s cestou, nie s otáznikom (WordPress: **Nastavenia** → **Trvalé odkazy** → **Názov príspevku**). Je to nutná podmienka: bez nej stránky používajú adresy typu `?p=123`, RSS kanál na `/feed/` neexistuje a filter pascí prehľadávača (sekcia 3) by odmietol všetky odkazy.

Kedy tento postup nie je vhodný

Statická kópia má principiálne obmedzenia. Všetko, čo vyžaduje spracovanie na serveri, na nej prestane fungovať:

- **Formuláre** - kontaktné formuláre, komentáre, prihlasovanie
- **Serverové vyhľadávanie** - dá sa nahradiť vyhľadávaním na strane prehliadača z vopred vygenerovaného indexu (napríklad súbor `search.json`, pozri sekciu 5)
- **Personalizovaný obsah** - košíky, používateľské účty a podobne
- **Interaktívne prvky riadené JavaScriptom** - napríklad rozbaľovacie menu moderných blokových tém alebo posuvníky. `wget` nesleduje odkazy vnútri JavaScriptu (import modulov), takže ich runtime môže byť neúplný. Počas overovania (sekcia 7) tieto prvky vyskúšajte; ak ich téma používa, buď doplňte príslušné JS súbory z disku, alebo zvoľte tému s CSS variantom.

Weby, ktoré sa celé vykresľujú JavaScriptom v prehliadači (SPA aplikácie), sa nástrojom `wget` skopírovať nedajú. Typické prezentačné weby inštitúcií však uvedené obmedzenia spravidla nemajú, prípadne sa dajú vyriešiť, a práve pre ne je tento postup ideálny. Skript migruje celý web, a nie len zmenené časti, no pre statické sídla toto nepredstavuje problém, celá migrácia by mala prebehnúť do niekoľkých minút.

01

KAPITOLA PRVÁ

Princíp a príprava

Ako celý proces funguje a čo si pripraviť pred prvým spustením.

1 Ako proces funguje

Namiesto exportu z databázy sa web **prehľadá tak, ako ho vidí návštevník**: nástroj `wget` stiahne úvodnú stránku, nájde v nej všetky odkazy a rekurzívne pokračuje, kým neprejde celý web. Výsledkom je adresárová štruktúra so všetkými stránkami vrátane tém a výstupov zásuvných modulov.

1. **Stiahnutie webu** - rekurzívne prehľadanie lokálnej inštancie nástrojom `wget`
2. **Doplnenie medzier** - súbory, ktoré prehľadávanie nezachytí (binárne prílohy, RSS kanál), sa doplnia priamo z disku
3. **Prepis URL adries** - odkazy na lokálnu inštanciu sa nahradia verejnou doménou
4. **Lokálne overenie** - kontrola kópie pred nasadením
5. **Nasadenie** - synchronizácia na verejný server cez `rsync`

> 📋 **UNIVERZÁLNE JADRO, ŠPECIFICKÉ DETAILY**
>
> Kroky 1, 4 a 5 sú pre každý web rovnaké. Krok 2 a konkrétne pravidlá prepisu v kroku 3 závisia od daného webu. Návod preto v príslušných sekciách popisuje, ako si potrebné nastavenia zistiť. Ukážkové hodnoty v tomto dokumente pochádzajú z inštalácie WordPress.

2 Príprava pracovného prostredia

Zvoľte si pracovný adresár, do ktorého sa bude web sťahovať. Adresár sa pri každom spustení premaže, aby v kópii nezostávali súbory z predchádzajúcich exportov (napríklad medzičasom zmazané stránky).

```
# pracovný adresár exportu sa pri každom behu vytvára nanovo
WORK_DIR="$HOME/web/export"
rm -rf "$WORK_DIR"
mkdir -p "$WORK_DIR"
cd "$WORK_DIR"
```

Pred spustením overte, že lokálna inštancia webu je skutočne spustená. Bez tejto kontroly by chybný beh mohol nasadiť prázdny web:

```
curl -fsS --noproxy '*' -o /dev/null "http://localhost:9000/" \
  || { echo "CHYBA: lokálna inštancia nie je spustená"; exit 1; }
```

✅ **FIREMNÉ PROXY**

Ak vaša sieť používa proxy server, prepínače `--noproxy '*'` (curl) a `--no-proxy` (wget) zabezpečia, že požiadavky na `localhost` nepôjdu cez proxy.

02

KAPITOLA DRUHÁ

Vytvorenie statickej kópie

Stiahnutie webu a doplnenie súborov z disku.

3 Stiahnutie webu nástrojom wget

Jadrom celého procesu je jediný príkaz. Objemné binárne súbory (obrázky, PDF, archívy) sa zámerne vynechávajú. Sťahovať ich cez HTTP je pomalé a zbytočné.

```
wget --no-hsts --no-proxy -e robots=off \
     --recursive -l inf --no-parent --domains localhost \
     --page-requisites --adjust-extension --convert-links \
     --no-use-server-timestamps \
     --reject 'jpg,jpeg,png,svg,gif,webp,ico,pdf,zip,rar,7z,doc,docx,xls,mp3,mp4' \
     --reject-regex='(\?|/wp-login|/wp-admin|/xmlrpc)' \
     "http://localhost:9000"
```

Význam jednotlivých skupín prepínačov:

Prepínače	Účel
`--recursive -l inf --no-parent --domains localhost`	Rekurzívne prehľadávanie bez limitu hĺbky, ale iba v rámci lokálneho webu, nikdy nesleduje externé odkazy
`--page-requisites --adjust-extension --convert-links`	Stiahne aj CSS, JS a ďalšie súbory potrebné na zobrazenie; doplní prípony (`.html`); prepíše odkazy tak, aby kópia fungovala samostatne
`--no-use-server-timestamps`	Správne názvy sťahovaných súborov; časové značky súborov podľa času exportu
`--no-hsts --no-proxy -e robots=off`	Lokálnu inštanciu sťahujeme priamo, bez proxy a bez ohľadu na `robots.txt` (ide o vlastný interný web)
`--reject '...'`	Zoznam prípon, ktoré sa nesťahujú, doplnia sa z disku v sekcii 5
`--reject-regex='(\?|/wp-login|/wp-admin|/xmlrpc)'`	Odmietne adresy s otáznikom a administračné rozhrania. Predpokladá zapnuté „pekné" URL (pozri Pred začatím).

⚠️ **PASCA PREHĽADÁVAČA**

Ak sťahovanie zdanlivo nekončí a sťahuje adresy s `?query-...` alebo `wp-login.php?redirect_to=...`, `wget` uviazol v pasci: stránka s viacerými nezávisle stránkovanými blokmi generuje obrovské množstvo adries. Rieši to `--reject-regex` vyššie. Opačný príznak: prehľadávanie skončí po jedinom súbore `index.html`, čo znamená vypnuté „pekné" URL (pozri Pred začatím).

Výsledok nájdete v adresári pomenovanom podľa adresy inštancie (`localhost:9000/`). Premenujte ho na `html` **hneď teraz**, ešte pred ďalšími krokmi:

```
mv "$WORK_DIR/localhost:9000" "$WORK_DIR/html"
# zmažte prípadné zvyšky prerušených sťahovaní
find "$WORK_DIR/html" -name '*.tmp' -delete
```

📋 **PREČO PREMENOVAŤ HNEĎ**

Dvojbodka v `localhost:9000` nie je len kozmetika, nástroje `rsync` a `scp` čítajú `host:cesta`, takže dvojbodku v ceste považujú za názov vzdialeného servera a zlyhajú s „Connection refused". Po premenovaní na `html` problém zmizne a všetky ďalšie kroky pracujú s `html/`.

4 Špeciálny prípad: RSS kanál

Prepínače `--adjust-extension` a `--convert-links` sú skvelé pre HTML stránky, ale XML súbory nimi `wget` poškodí. RSS kanál by dostal príponu `.html` a prepísané adresy. Kanál preto stiahneme samostatne nástrojom `curl` a uložíme na správne miesto.

```
mkdir -p "$WORK_DIR/html/feed"
# WordPress ponúka kanál na viacerých adresách, skúšame ich po poradí
for feed_url in "http://localhost:9000/feed/" \
                "http://localhost:9000/?feed=rss2" \
                "http://localhost:9000/?feed=atom"; do
  curl --fail -sS --location --retry 3 --noproxy '*' \
       "$feed_url" -o "$WORK_DIR/html/feed/index.xml" && break
done
```

📋 **VŠEOBECNÝ PRINCÍP**

RSS kanál je len najčastejší príklad. Rovnako ošetrite akýkoľvek iný súbor, ktorý prehľadávanie poškodí alebo vynechá, napríklad `sitemap.xml` alebo súbory generované na požiadanie. Vzor je vždy rovnaký: stiahnuť samostatne cez `curl` a uložiť na očakávanú cestu. Ak inštancia požiadavku odmieta, pridajte hlavičku prehliadača (`-A "Mozilla/5.0 ..."`), niektoré bezpečnostné moduly blokujú predvolený User-Agent nástroja `curl`.

5 Doplnenie súborov z disku

Teraz treba doplniť súbory, ktoré `wget` odmietol (obrázky, fonty, PDF) a prípadné súbory, na ktoré nevedie žiadny odkaz. Namiesto vymenúvania jednotlivých priečinkov skopírujeme **celý `wp-content` jedným pravidlom**: rekurzívne prejdeme všetky adresáre, ponecháme len známe statické prípony a vynecháme PHP a vyrovnávaciu pamäť. Takto sa doplnia obrázky tém, prílohy modulov aj fonty bez ohľadu na to, kde ležia.

```
WP_ROOT="/var/www/html"   # koreň inštalácie CMS na disku
DST="$WORK_DIR/html"

rsync -a --mkpath \
  --exclude='cache/' \
  --exclude='*.php' \
  --include='*/' \
  --include='*.jpg'  --include='*.jpeg' --include='*.png' --include='*.gif' \
  --include='*.svg'  --include='*.webp' --include='*.ico' \
  --include='*.woff' --include='*.woff2' --include='*.ttf' --include='*.eot' \
  --include='*.pdf'  --include='*.json' --include='*.css' --include='*.js' \
  --include='*.mp3'  --include='*.mp4' \
  --exclude='*' \
  "$WP_ROOT/wp-content/" "$DST/wp-content/"
```

Pravidlá `rsync` sa vyhodnocujú zhora nadol: najprv vylúčime cache a PHP, potom `--include='*/'` povolí vstup do všetkých podadresárov, ďalej zoznam prípon určí, čo ponechať, a záverečné `--exclude='*'` zahodí zvyšok. Prispôsobte zoznam prípon podľa svojho webu.

WordPress však servíruje statické súbory jadra (štýly blokov, základné JS, fonty) aj z **druhého adresára `wp-includes/`**. Bez nich stratí najmä titulná stránka štýlovanie blokov. Skopírujeme ho rovnakým pravidlom:

```
rsync -a --mkpath \
  --exclude='*.php' \
  --include='*/' \
  --include='*.css'  --include='*.js'  --include='*.json' \
  --include='*.woff' --include='*.woff2' --include='*.ttf' --include='*.eot' \
  --include='*.svg'  --include='*.png' --include='*.gif' \
  --exclude='*' \
  "$WP_ROOT/wp-includes/" "$DST/wp-includes/"
```

📋 **UNIVERZÁLNY PRINCÍP: SKOPÍRUJTE VŠETKY ADRESÁRE SO STATICKÝMI SÚBORMI**

Iný CMS má tieto súbory inde, podstatné je skopírovať *každý* adresár, z ktorého web servíruje statické súbory (u WordPressu `wp-content` aj `wp-includes`). Ktoré to sú, spoľahlivo odhalí kontrola visiacich odkazov v sekcii 7: chýbajúci súbor v ceste `/wp-includes/...` znamená, že tento adresár treba doplniť.

✅ **OVERENIE KÓPIE SÚBOROV**

Otestujte pravidlo na malom strome, kým sa naň spoľahnete: skopírujte vzorku a overte, že prešli obrázky/fonty/JSON a *neprešli* PHP súbory ani obsah `cache/`. V praxi je najlepšou kontrolou lokálne prezretie webu (sekcia 7), chýbajúci súbor sa prejaví ako chyba 404 v konzole prehliadača a znamená, že do zoznamu prípon treba pridať ďalší typ.

📋 **ALTERNATÍVA: DEFINOVANÝ ZOZNAM**

Univerzálne pravidlo skopíruje aj súbory, na ktoré web neodkazuje (napr. médiá nepoužitých modulov). Ak potrebujete **auditovateľný zoznam** presne toho, čo sa nasadí, vymenujte priečinky jednotlivými `rsync` riadkami. Chýbajúci zdroj potom rieši tolerantný zápis `rsync ... || echo "WARN"`, aby jeden neexistujúci priečinok nezastavil celý beh.

📋 **KOPÍRUJÚ SA AJ NEAKTÍVNE TÉMY A MODULY**

Univerzálne pravidlo prejde *celý* adresár `wp-content`, takže skopíruje statické súbory aj z neaktívnych tém a vypnutých modulov, ktoré ostali na disku. Na fungovanie webu to nemá vplyv (nič na ne neodkazuje), ale na verejný server sa tak dostanú súbory, ktoré tam nemusia byť. Ak to prekáža (napr. z bezpečnostných dôvodov nechcete zverejniť súbory nepoužívaného modulu), vylúčte konkrétne adresáre pridaním `--exclude='themes/nazov-temy/'` alebo `--exclude='plugins/nazov-modulu/'` pred záverečné `--exclude='*'`. Ktoré adresáre existujú, zistíte príkazom `ls wp-content/themes` a `ls wp-content/plugins`.

✅ **VYHĽADÁVANIE NA STATICKOM WEBE**

Serverové vyhľadávanie na statickej kópii nefunguje, ale téma webu môže generovať index (napr. `search.json`), v ktorom vyhľadáva JavaScript priamo v prehliadači. Ak váš web vyhľadávanie potrebuje, toto je osvedčený vzor a keďže `.json` je v zozname prípon vyššie, index sa skopíruje automaticky.

03

KAPITOLA TRETIA

Úprava a overenie

Prepis lokálnych URL adries na verejnú doménu a kontrola kópie pred nasadením.

6 Prepis URL adries

Kópia stále obsahuje absolútne odkazy na lokálnu inštanciu (`http://localhost:9000/...`), najmä v adresách statických súborov, v RSS kanáli a v meta značkách.

- **HTML, CSS, JS, JSON** - navigačné odkazy už `--convert-links` spravil relatívnymi (napr. `about/index.html`) a na statickom serveri fungujú. Stačí teda *odstrániť* absolútny názov hostiteľa z adries súborov, čím sa z `http://localhost:9000/wp-content/x.jpg` stane `/wp-content/x.jpg`.
- **XML (RSS kanál)** - čítačky kanálov nevedia interpretovať relatívne adresy, preto v kanáli názov hostiteľa *nahradíme* verejnou doménou, aby boli adresy absolútne.

Ako zistiť, čo treba prepísať

Najprv si pozrite, v akých podobách sa lokálna adresa v kópii vyskytuje:

```
grep -rho "http://localhost[^\"' <)]*" "$WORK_DIR/html" | sort | uniq -c | sort -rn | head
```

Väčšinou pôjde o adresy statických súborov (často s neškodnou koncovkou `?ver=...`, ktorú **nechajte tak**, statický server ju ignoruje). Potom spustite dva prepisy, textové súbory a XML zvlášť:

```
export LC_ALL=C
# 1) HTML/CSS/JS/JSON - odstrániť hostiteľa (odkazy zostanú relatívne)
find "$WORK_DIR/html" -type f \
     \( -name '*.html' -o -name '*.htm' -o -name '*.css' \
        -o -name '*.js' -o -name '*.json' \) -print0 |
  xargs -0 sed -i 's#http://localhost:9000##g'
# 2) XML kanál - nahradiť hostiteľa verejnou doménou (odkazy budú absolútne)
find "$WORK_DIR/html" -name '*.xml' -print0 |
  xargs -0 sed -i 's#http://localhost:9000#https://csirt.sk#g'
```

7 Lokálne overenie

Toto je najdôležitejší krok celého postupu - spätná väzba, ktorá z prvého nedokonalého exportu urobí kompletný. Kópiu si pred nasadením prezrite lokálne:

```
cd "$WORK_DIR/html"
python3 -m http.server 8080
# web je teraz dostupný na http://localhost:8080
```

Skontrolujte postupne:

1. **Navigáciu** - preklikajte hlavné menu, podstránky, jazykové mutácie
2. **Obrázky a súbory na stiahnutie** - v konzole prehliadača (karta Sieť) nesmú byť chyby 404; každá nájdená chyba môže znamenať nový typ prípony do zoznamu v sekcii 5
3. **Interaktívne prvky** - rozbaľovacie menu, posuvníky (pozri obmedzenia v úvode)
4. **RSS kanál** - `http://localhost:8080/feed/` musí vrátiť platné XML s *absolútnymi* verejnými adresami
5. **Zvyšky lokálnych adries** - v HTML, CSS a XML nesmie zostať žiadna:

```
grep -rl "http://localhost" "$WORK_DIR/html" \
  --include='*.html' --include='*.css' --include='*.xml'
```

Nakoniec skontrolujte **visiace odkazy** - odkazy na súbory, ktoré v kópii neexistujú:

```
cd "$WORK_DIR/html"
grep -rho '\(src\|href\)="[^"]*"' . | grep -oE '/(wp-includes|wp-content)[^"]*' \
  | sed 's/?.*//' | sort -u | while read -r p; do
    [ -e ".$p" ] || echo "CHÝBA: $p"
  done | head -30
```

Výsledok by mal obsahovať len súbory, ktoré vedome neriešite (napr. JS moduly interaktívnych prvkov). Čokoľvek iné, chýbajúci štýl alebo obrázok znamená medzeru v kopírovaní súborov (sekcia 5).

> ✅ **SLUČKA: OVER → DOPLŇ → ZOPAKUJ**
>
> Ak overenie odhalí chýbajúce súbory alebo neprepísané adresy, doplňte príslušný riadok v sekcii 5 alebo pravidlo v sekcii 6 a celý export spustite znova. Po dvoch-troch iteráciách býva kópia kompletná, a keďže doplnky zapisujete rovno do skriptu, každý ďalší export je už bezchybný na prvý pokus.

04

KAPITOLA ŠTVRTÁ

Nasadenie a automatizácia

Synchronizácia na verejný server a zostavenie celého postupu do skriptu.

8 Nasadenie na server

Overenú kópiu prenesieme na verejný server (DMZ STATIC). Podľa bezpečnostnej politiky DMZ sa **prenos súborov robí cez SFTP**. Ten istý server však prijíma aj SSH príkazy, ktoré použijeme na vymazanie starej kópie a nastavenie práv. Postup má tri kroky:

```
REMOTE="uzivatel@dmz-static"
REMOTE_DIR="/var/www/web-static"

# 1. bezpečnostná poistka: nič nenasadzuj, ak export nie je kompletný
[ -s "$WORK_DIR/html/index.html" ] || { echo "CHYBA: prázdny export"; exit 1; }

# 2. zmaž starú kópiu a nahraj novú cez SFTP
ssh "$REMOTE" "rm -rf '$REMOTE_DIR/'*"
sftp "$REMOTE" <<SFTP
put -r $WORK_DIR/html/* $REMOTE_DIR/
bye
SFTP

# 3. nastav vlastníctvo a práva, aby web server súbory prečítal
ssh "$REMOTE" "chown -R uzivatel:www-data '$REMOTE_DIR' && chmod -R u=rwX,g=rX,o=rX '$REMOTE_DIR'"
```

⚠ **PRÁVA NA ČÍTANIE SÚ NUTNÉ**

Súbory prenesené cez SFTP môžu mať prísne práva, ktoré webový server neprečíta. Výsledkom je chyba **403 Forbidden**. Preto krok 3 nastaví vlastníctvo na skupinu `www-data` a práva na čítanie (`chmod -R u=rwX,g=rX,o=rX` , veľké `X` pridá právo vstupu iba adresárom, nie súborom). Bez tohto kroku sa nasadený web nemusí zobraziť.

📋 **KOMPROMIS: PRÁZDNE OKNO POČAS NAHRÁVANIA**

Keďže SFTP neprenáša inkrementálne, celý web sa nahráva nanovo a medzi vymazaním a koncom nahrávania je verejná kópia krátko neúplná. Je to daň za povinný prenos cez SFTP. Prvý krok (kontrola `index.html`) aspoň zabráni tomu, aby sa web vymazal, keď je export pokazený. Ak by prázdne okno prekážalo, dá sa nahrávať do dočasného adresára a na záver ho prehodiť. V tomto návode volíme jednoduchšiu podobu.

9 Automatizácia skriptom

Po odladení slučky z kapitoly 3 zostavte celý postup do jedného skriptu. Kompletný odladený skript pre WordPress je súčasťou prílohy tohto dokumentu.

```
#!/usr/bin/env bash
set -euo pipefail          # pri prvej chybe skript skončí, nikdy nenasadí pokazenú kópiu

# ── konfigurácia: všetky hodnoty špecifické pre web na jednom mieste ──
LOCAL_HOST="localhost:9000"
PROD_URL="https://csirt.sk"
REMOTE="uzivatel@staticky-server"
REMOTE_DIR="/var/www/web-static"
WP_ROOT="/var/www/html"
WORK_DIR="$HOME/web/export"
LOG_FILE="$HOME/web/export.log"

log() { echo "$(date +%FT%T) $*" | tee -a "$LOG_FILE"; }

log "export: začiatok"
# 0. kontrola, že lokálna inštancia je spustená (sekcia 2)
# 1. wget - stiahnutie + reject-regex, mv html, zmazať .tmp   (sekcia 3)
# 2. curl - RSS kanál                          (sekcia 4)
# 3. rsync - doplnenie súborov z disku         (sekcia 5)
# 4. sed - prepis URL: HTML odstrániť, XML nahradiť  (sekcia 6)
# 5. ssh rm + sftp put + ssh chown - nasadenie  (sekcia 8)
log "export: koniec"
```

Zásady, ktoré skript dodržiava:

- `set -euo pipefail` - akákoľvek neošetrená chyba beh zastaví; jediná tolerovaná výnimka je návratový kód 8 pri `wget` (sekcia 3)
- Všetky hodnoty špecifické pre web sú v konfiguračnom bloku na začiatku - prispôsobenie inému webu znamená upraviť len tento blok a zoznamy zo sekcií 5 a 6

- **Všetky cesty v konfigurácii musia byť absolútne** (napr. `/var/www/html` alebo `"$HOME/..."`). Skript v priebehu behu prejde do pracovného adresára (`cd`), takže relatívna cesta (napr. `./wordpress`) by sa vyhodnotila voči nemu a kopírovanie by potichu zlyhalo.
- Každý beh zapíše do denníka časovú značku začiatku a konca

Pravidelnú aktualizáciu statickej kópie potom zabezpečí `cron`, napríklad každú noc o 3:00:

```
0 3 * * * /usr/local/bin/staticka-kopia-webu.sh >> /var/log/web-export.log 2>&1
```

Záver

Po dokončení tohto návodu máte plne statickú kópiu webu na verejnom serveri, redakčný systém bezpečne skrytý vo vnútornej sieti a skript, ktorý celý export zopakuje jedným príkazom. Aktualizácia webu má odteraz jednoduchý rytmus: upravte obsah v redakčnom systéme, spustite skript (alebo počkajte na naplánovaný beh) a zmeny sa objavia na verejnom webe.

Pri prenose postupu na iný web postupujte podľa slučky z kapitoly 3: prvý export, lokálne overenie, doplnenie zoznamov v sekciách 5 a 6, opakovanie - a po dvoch-troch iteráciách zapíšte výsledok do konfigurácie skriptu.

Kompletný skript

Kompletný ukážkový skript `staticka-kopia-webu.sh` je udržiavaný v git repozitári CSIRT.sk. Aktuálnu verziu spolu s históriou zmien nájdete tu:

https://github.com/CSIRT-SK/staticka-kopia-webu

Pred použitím upravte konfiguračný blok na začiatku skriptu (všetky cesty musia byť absolútne) a podľa potreby zoznamy v sekciách 5 a 6.