

# Ochrana prezentací vytvořených v systému Visual PHP™ proti vnějším útokům

---

## 1 – Cross Site Scripting (XSS)

**Problém:** Nejobvyklejší a nejběžnější chyba zabezpečení webových aplikací. XSS vznikne v okamžiku, kdy aplikace odesílá uživatelská data webovému prohlížeči, aniž by nejprve tento obsah ověřila nebo zašifrovala. To umožní hackerům spustit škodlivé skripty v prohlížeči a unést uživatelské relace, znetvořit webové stránky, vložit nepřátelský obsah či řídit phishingové a malwarové útoky. Útoky jsou obvykle vykonávány prostřednictvím JavaScriptu, který umožňuje hackerům manipulovat s jakoukoli vlastností stránky. V nejhorším scénáři může hacker ukrást informace a vydávat se na webových stránkách banky za oprávněného uživatele.

**Příklad ze života:** PayPal se nedávno stal terčem útoku, při kterém útočníci přesměrovali návštěvníky webu této organizace na stránku s upozorněním, že jejich účty jsou ohroženy. Oběti byly přesměrovány na phishingový server a požádány o zadání přihlašovacích údajů k účtu PayPal, čísla SSN (slouží k identifikaci osob v USA) a podrobné údaje o kreditních kartách. Společnost Pay-Pal už chybu samozřejmě opravila.

**Ochrana Visual PHP™:** Integrovaný Firewall nedovolí zasílat formulářem nebo přímou URL žádný HTML kód. Díky tomu nelze znehodnotit stránku vložením obrázků, odkazů (typický příklad je spam diskuzí), nebo dokonce zdrojových kódů (např. JavaScript, který by byl spuštěn na straně prohlížeče). Všechny tyto pokusy jsou logovány (datum a čas, IP adresa útočníka, referer, zasílaná data).

## 2 – Injection Flaws

**Problém:** Pokud jsou data od uživatele odesílána do interpreterů (komponenta, která interpretuje příkazy zadané v textové podobě) jako součást příkazu nebo dotazu, zkoušejí je hackeři oklamat tak, aby nekontrolovaně vykonaly jejich příkazy. Tyto chyby umožňují útočníkům vytvářet, číst, aktualizovat a smazat jakákoli v aplikaci dostupná data. V nejhorším případě umožní tyto chyby zabezpečení útočníkovi zcela poškodit aplikaci a nosné systémy, a to dokonce i ty, které jsou ukryty hluboko za firewallem.

**Příklad ze života:** Ruští hackeři se v roce 2006 vlámali do vládního webu Rhode Islandu za účelem ukrást data o kreditních kartách. Sami tvrdí, že pomocí útoku SQL injection (vlození příkazů do dat pro SQL) ukradli 53 tisíc čísel kreditních karet, zatímco poskytovatel hostingových služeb tehdy prohlásil, že jich bylo pouze 4 113.

**Ochrana Visual PHP™:** Systém je naprogramován tak, aby nedovolil podstrčit výslednému SQL dotazu další instrukce. Veškerá data jsou escapována (jsou správně ošetřeny uvozovky a další

znaky umožňující napadení) před sestavením výsledného SQL dotazu. Zásilaná data také prochází firewallem, který typické pokusy o SQL injection ihned odhalí, další generování stránky zablokuje a zapíše tento pokus do logu (datum a čas, IP adresa útočníka, referer, zasílaná data).

### 3 - Spuštění škodlivého souboru

Problém: Hackeri mohou vzdáleně spustit kód, vzdáleně nainstalovat rootkity nebo zcela poškodit systém. Jakýkoli typ webové aplikace je zranitelný, pokud akceptuje názvy souborů nebo soubory od uživatelů. Tato chyba zabezpečení je nejběžnější u PHP, který je široce používaným skriptovacím jazykem pro vývoj webů.

Příklad ze života: V roce 2002 zjistil velmi mladý programátor, že web Guess.com je zranitelný vůči útokům, které by mohly umožnit krádež záznamů více než 200 tisíc zákazníků z databáze Guessu, a to včetně jmen, čísel kreditních karet a dat jejich expirace. Společnost přislíbila modernizovat zabezpečení následující rok, když byla vyšetřována úřadem Federal Trade Commission.

Ochrana Visual PHP™: Veškeré soubory, které jsou nahrávány na server, nesmí obsahovat příponu spustitelnou interpretem na serveru (vyloučeny jsou přípony .php, .phtml, php3, php4, php5, php6, a .pl). Každý soubor, který je nahráván na server navíc prochází integrovaným antivirem, který umí odhalit typické hackerské skripty umožňující ovládnout server.

### 4 - Nezabezpečené přímé odkazy na objekty

Problém: Útočník změní přímé odkazy na objekty, aby získal neoprávněný přístup k dalším objektům. To se stává, pokud adresy URL nebo parametry formulářů obsahují odkazy na objekty, jako jsou soubory, adresáře a databázové záznamy či klíče. Webové servery bank obvykle používají číslo účtu zákazníka jako primární klíč, a mohou tak vyrazit čísla účtů ve webovém rozhraní. Odkazy na klíče databází jsou často vystaveny. Útočník tyto parametry může využít k útoku jednoduchým hádáním nebo hledáním dalšího platného klíče. Často mají obyčejnou sekvenční podobu.

Příklad ze života: V roce 2000 byl hacknut web australského finančního úřadu uživatelem, který změnil DIČ obsažené v adrese URL, a získal tak přístup k údajům o 17 tisíc společnostech. Hacker tyto firmy e-mailem upozornil na chybu v zabezpečení.

Ochrana Visual PHP™: V systému nejsou pro jedinečné identifikátory použity sekvenční čísla (tedy například sekvence 1,2,3,4, 5,6,7,... což jsou typicky 32bitová čísla), ale celosvětově unikátní 256bitové UUID identifikátory (tedy například: ebe79a36-fa5c-102c-ab95-00e0814daf34). Každé generování tohoto UUID je unikátní a nelze odhadnout následníka.

### 5 - Podvržení požadavků mezi weby

Problém: Je jednoduchý a zničující – tento útok převezme řízení prohlížeče oběti během přihlášení k webu a odesílá škodlivé požadavky webové aplikaci. Weby jsou extrémně zranitelné částečně proto, že se pokoušejí ověřovat požadavky na základě souborů cookie dané relace nebo pomocí funkce „zapamatuj si mne“. Mezi potenciální cíle patří banky. 99 % aplikací na internetu neodolá podvrhu

požadavků mezi weby. Došlo ke skutečnému zneužití, když někdo přišel o peníze? Banky to zřejmě vůbec nezjistí. Těm se vše jeví jako legitimní transakce přihlášeného uživatele.

Příklad ze života: Hacker známý jako Samy získal koncem roku 2005 více než milion „přátel“ na webu MySpace.com pomocí červa, který automaticky přidával zprávu „Samy je můj hrdina“ na tisících stránkách webu MySpace. Samotný útok nebyl moc škodlivý, ale byl proveden tak, aby předvedl sílu kombinace skriptování mezi weby a podvržení požadavků mezi nimi. Další příklad, který se stal před více než rokem, ukázal zranitelnost služby Google tím, že cizí weby mohly změnit jazyková nastavení uživatele služby Google.

Ochrana Visual PHP™: Systém používá speciální token, který je vygenerován při přihlášení. Bez platného tokenu není možné provádět operace v administraci.

Doporučení: Pro maximální možné zabezpečení dále doporučujeme používat prohlížeč Mozilla FireFox a jeho rozšíření Prism, ve kterém lze přistupovat do administrace v samostatném vlákně. Díky tomu nejsou cookies sdílena s jiným vláknem prohlížeče.

## 6 - Únik informací a nesprávné zpracování chyb

Problém: Chybové zprávy, které aplikace generují a zobrazují uživatelům, jsou rovněž použitelné pro hackery, když narušují soukromí. Tyto zprávy totiž neúmyslně vyrazují informace o konfiguraci programu i interním zpracování. Webové aplikace často odhalují data o svém interním stavu prostřednictvím podrobných nebo ladicích chybových hlášení. Nežádka lze tyto údaje využít k zahájení nebo k zautomatizování silnějších útoků.

Příklad ze života: Únik informací snadno nastává po zpracování chyby. Bývá zaznamenán také po výskytu chyb zabezpečení, kdy jsou důvěrná data ponechána zcela na oddiv. Debakl společnosti ChoicePoint začátkem roku 2005 spadá zhruba do této kategorie. Záznamy 163 tisíc zákazníků byly vyraženy poté, co si zločinci vydávající se za legitimní zákazníky organizace ChoicePointu vyhledali podrobnosti o jednotlivcích uvedené v databázi personálních informací této organizace. ChoicePoint následně omezil prodej produktů obsahujících důvěrná data.

Ochrana Visual PHP™: Ladicí informace, varování a chyby jsou zobrazovány pouze na IP adresy vývojářů. Útočník proto nemůže na základě těchto hlášení zjistit blíže strukturu databáze a uspořádání systému souborů.

## 7 - Porouchané ověřování a správa relací

Problém: Pokud aplikace selže při ochraně relací a pověření, může dojít ke krádeži uživatelských a administrátorských účtů. Dávejte pozor na narušení soukromí nebo funkční omezení řízení oprávnění a zodpovědnosti. Chyby v hlavním mechanismu ověřování nejsou vzácné, ale nedostatky se častěji vyskytují ve formě slabých pomocných funkcí ověřování, jako je přihlašování, odhlašování, správa hesla, časový limit, funkce „zapamatuj si mne“, tajné otázky a aktualizace účtu.

Příklad ze života: Společnost Microsoft musela v roce 2002 odstranit chybu zabezpečení služby

Hotmail, která dovoľovala tvůrcům škodlivého kódu JavaScriptu ukrást uživatelská hesla. Tato chyba byla zjištěna prodejcem síťových produktů a e-mailům obsahujícím trojské koně dovoľovala změnit uživatelské rozhraní služby Hotmail a nutit uživatele k tomu, aby několikrát zadali své heslo, které bylo bez jejich vědomí odesláno hackerům.

Ochrana Visual PHP™: Citlivé informace jako například hesla jsou ukládána výhradně jako hash (pouze jednosměrné kódování, hesla proto nelze dekódováním získat zpět). Systém navíc umožňuje ukládat další citlivé informace pomocí šifrování).

## 8 - Nezabezpečené kryptografické úložiště

Problém: Mnoho webových vývojářů důvěrná data v úložištích nešifruje, přestože je kryptografie důležitou součástí většiny webových aplikací. Ale i když je šifrování použito, je často navrženo nedostatečně a využívá nevhodné šifry. Tyto chyby mohou způsobit vyzrazení důvěrných dat a umožnit narušení.

Příklad ze života: Nedávný únik dat společnosti TJX vyzradil 45,7 milionu čísel kreditních a debetních karet. Průzkum kanadské vlády kritizuje firmu TJX za to, že neupgradovala svůj šifrovací systém předtím, než se stala cílem elektronického odposlouchávání, které začalo v červenci 2005.

Ochrana Visual PHP™: Citlivé informace lze kódovat rozličnými algoritmy (3DES, DES, TripleDES, ENIGMA, IDEA, RIJNDAEL\_256, RC6, SAFER128, SERPENT\_256, THREWAY, TWOFISH256)

## 9 - Nezabezpečená komunikace

Problém: Podobně jako u bodu 8 (nezabezpečené kryptografické úložiště) se jedná o nesplnění požadavku šifrovat kvůli ochraně důvěrné komunikace síťové přenosy. Útočníci mohou přistupovat k nechráněným konverzacím, včetně přenosů pověřovacích dat a důvěrných informací. Z tohoto důvodu například normy pro kreditní transakce požadují šifrování informací o kreditních kartách přenášených přes internet.

Příklad ze života: Opět společnost TJX. Vyšetřovatelé se domnívají, že ke krádeži dat bezdrátově přenášených mezi přenosnými zařízeními pro kontrolu ceny, pokladnami a počítači obchodů použili hackeři teleskopickou anténu a notebook, jak píše Wall Street Journal. Bezdrátová síť za 17,4 miliardy dolarů patřící prodejněmu řetězci byla méně zabezpečená než síť většiny domácností. TJX například používal šifrování WEP namísto robustnějšího WPA.

Ochrana Visual PHP™: Systém plně podporuje přenos dat pomocí SSL šifrování. Díky tomu jsou data chráněna i ve slabě zabezpečených sítích.

## 10 - Nefunkční omezení přístupu k adrese URL

Problém: U některých webových stránkách se očekává omezení přístupu pouze na malou skupinu privilegovaných uživatelů, jako jsou například správci. Často však tyto stránky nemají žádnou skutečnou ochranu a hackeři mohou adresu zjistit pomocí inteligentního hádání. Řekněme, že adresa

URL odkazuje na číslo ID „123456“. Hacker si může říci „A copak najdu na ID 123457?“. Útoky zaměřené na tuto chybu zabezpečení jsou nazývané vynucené prohlížení (forced browsing) a zahrnují hádání odkazů a techniky využití hrubé síly k nalezení nechráněných stránek.

Příklad ze života: Bezpečnostní díra na webu konference Macworld Conference & Expo minulý rok umožnila uživatelům získat vstupenky „Platinum“ za téměř 1 700 dolarů a zvláštní přístup na přednášku Steva Jobse, vše zdarma. Chybou byl kód, který ověřoval privilegia na klientském počítači, nikoliv však na serveru. To umožnilo získat lidem vstupenky prostřednictvím JavaScriptu v prohlížeči namísto serveru.

Ochrana Visual PHP™: Systém provádí pokaždé autentizaci při přístupu na stránku, která se má zobrazit pouze přihlášeným uživatelům. Tajné URL adresy bez autentizace nejsou v tomto systému použity.

# Doplňující ochrany Visual PHP™

---

## 11 – Ukradení session id

**Problém:** V případě, že jsou session id zasílána přes URL, může dojít k vyrazení identifikátoru kliknutím na odkaz, směřujícím na stránky útočníka. Skript na straně útočníka získá session z HTTP hlavičky Referer. Další chybou může být přímé zaslání URL obsahující session někomu jinému (např. emailem nebo jinými sociálními sítěmi – Skype, ICQ, ...). Příjemce tak získá kliknutím na tento odkaz přímý přístup do rozhraní, které má být přístupné pouze původnímu uživateli.

**Ochrana Visual PHP™:** Systém ukládá session do cookies, aby nedošlo k vyrazení session při přeposílání URL adresy např. emailem nebo jinou sociální sítí, nebo kliknutím na externí odkaz a zjištěním hodnoty referer. Navíc je získání session z cookies zabráněno díky ochraně proti XSS, kdy nelze vložit do stránek JavaScript, umožňující zjistit hodnoty z cookies a zaslat je útočníkovi.

## 12 – Získání emailů ze stránek

**Problém:** Na většině webů jsou emaily vystaveny v nechráněné podobě. Robotům procházejícím tyto stránky jsou tak lehce nabídnuty emaily, na které jsou zasílány spamy. Jakmile se email jednou dostane do databáze spamů, množství spamů zasílaných na tento email neustále vzrůstá.

**Ochrana Visual PHP™:** Všechny emaily vystavené na stránkách jsou v zakódované podobě. Nelze je tedy jednoduchou formou získat. Díky implementaci pomocí JavaScriptu současně zůstává zachována funkčnost v podobě otevření emailového klienta kliknutím na odkaz tohoto emailu.

## 13 – Plnění formulářů spam roboty

**Problém:** Různí roboti prochází stránky a vyhledávají v nich formuláře. Z těchto formulářů si zjistí názvy formulářových polí a poté periodicky zasílají informace do těchto formulářů (typické je spamování diskuzních fór, formulářů "Napište nám", atd...). Pokud nejsou tyto formuláře ochráněny opisem znaků z obrázku, může dojít k periodickému zasílání nechtěných informací.

**Ochrana Visual PHP™:** Systém do formulářů přidává skryté pole, které robot vyplní, ale skutečný návštěvník nikoli. V případě, že došlo k zaslání libovolné hodnoty z tohoto pole, dojde k blokadě dalšího zpracování stránky a zápisu do logu firewallu.