ZAŠTITA WEB SERVISA U .NET OKRUŽENJU

Snežana Šućurović, *Institut Mihailo Pupin, Beograd*

Sadržaj – *U radu su opisana rešenja za autentifikaciju, autorizaciju i kriptovanje Web servisa u .NET okruženju. Cilj rada je bio da se ispitaju mogućnosti zaštite Web servisa, pre nego što se počne sa razvojem aplikacije koja bi se bazirala na ovoj tehnologiji.*

1. UVOD

Upravljanje sistemom heterogenih baza koristeći Web Servise ima brojne prednosti. Integracija informacija koristeći XML preko HTTP-a donosi uštede u vremenu, a samim tim i u novcu. Na samom početku razvoja ovog sistema postavilo se pitanje zaštite. Razvoj tehnologija na kojima se zasnivaju Web Servisi (*Simple Object Access Protocol* - SOAP, XML, HTTP), a takođe i zaštita, je vođen od strane World Wide Web Consortium (W3C). U ovom konzorcijumu najveću ulogu imaju Microsoft i IBM. Pošto standardi u zaštiti Web Servisa još nisu finalizovani, mi smo se opredelili za rešenje koje nudi Microsoft u .NET okruženju.

2. AUTENTIFIKACIJA

Autentifikacija, kao proces dokazivanja identiteta, je prvi korak u implementaciji zaštite u bilo kojoj aplikaciji. Preporučuje se da se koristi autentifikacija koju daje platforma na kojoj se aplikacija izvršava. U slučaju Web Servisa koji se izvršava u .NET okruženju, platformu čine: Windows operativni sistem, Internet Information System i .NET *Framework*. IIS nudi sledeća tri mehanizma za autentifikaciju:

- *Basic*
- *Digest*
- *Integrated Windows*

Basic autentifikacija se zasniva na korišćenju korisničkog imena i lozinke (*credentials*). *Web browser* prikazuje dijalog, u koji korisnik unosi korisničko ime i lozinku prethodno definisanog računa na Windows operativnom sistemu. Prednost *Basic* autentifikacije je to što je ona sastavni deo HTTP specifikacije, pa je time najveći broj *browser*-a podržava. Nedostatak ovog vida autentifikacije je što se *credentials* prenose bez kriptovanja i time napadači koji imaju mogućnost monitoringa mreže mogu da pročitaju lozinku. Zbog toga se *Basic* autentifikacija preporučuje samo ukoliko se koristi SSL (konfigurisani su digitalni sertifikati u *browser*-u i na Web serveru) ili se koristi iznajmljena linija.

Digest autentifikacija koristi *hash* algoritam da bi formirala heksadecimalnu reprezentaciju kombinacije korisničkog imena, lozinke, traženog resursa, HTTP metoda i slučajnog broja koji generiše server. Ovaj tip autentifikacije nije tako siguran kao Kerberos ili autentifikacija koja se zasniva na *public* kriptosistemu, ali je jača forma autentifikacije nego što je *Basic*. *Digest* autentifikacija je deo HTTP 1.1 specifikacije, što zahteva da *browser* podržava ovu specifikaciju. U praksi se pokazalo da Netscape, kao i jedan broj drugih *browser*-a ne podržava *digest* autentifikaciju prema HTTP 1.1 i zbog toga se ona preporučuje samo ukoliko se unapred zna da će svi *browser*-i biti Internet Explorer 4.0 i više verzije.

Integrated Windows autentifikacija ne šalje korisničko ime i lozinku preko mreže. Umesto toga *browser* šalje serveru *hash* korisnikovih *credentials*. Ranije se ovaj vid autentifikacije nazivao NTLM ili Windows NT *challenge/response* autentifikacija. *Integrated Windows* autentifikacija može koristiti Kerberos v5 autentikacioni protokol ili sopstveni *challenge/response* autentifikacioni protokol. Ukoliko je na serveru instaliran *Active Directory* servis i *browser* kompatibilan sa Kerberos v5 autentifikacionim protokolom, koriste se oba protokola, inače se koristi samo *challenge/response* protokol. Ograničenja ovog vida autentifikacije su: ne radi preko HTTP *proxy* konekcija i potrebno je da se otvore dodatni portovi na *firewall*-u, pošto *Integrated Windows* autentifikacija ne koristi port 80. Iz tih razloga Windows autentifikacija je najpogodnija za Intranet okruženje.

ASP .NET podržava *Forms* i *Passport* mehanizme autentifikacije, ali se ne preporučuje njihovo korišćenje u Web Servisima. *Forms* autentifikacija je mehanizam po kome se vrši redirekcija zahteva ka HTML formi, kroz koju korisnik prosleđuje korisničko ime i lozinku. Oni se šalju serveru u zaglavlju zahteva. ASP.NET vrši autentifikaciju takvih zahteva kroz validacione metode koje je programer specificirao. *Passport* je istovremeno i prvi MS Web Servis. Korisnik dobija login ekran u kome popunjava korisničko ime i lozinku, ali se koristi MS baza korisnika koja se čuva na centralnom mestu. I *Forms* i *Passport* autentifikacija koriste login ekran, ali korisnici Web Servisa ne mogu da taj ekran procesiraju i zbog toga se ove dva mehanizma ne preporučuju programerima Web Servisa.

Ako se programer odluči da ne koristi autentifikaciju koju daje platforma, potrebno je implementirati kastomizovan mehanizam. U tom slučaju bilo bi potrebno da se korisničko ime i lozinka prosleđuju u svakom pozivu metoda Web Servisa. Umesto ovoga, najčešće se korisničko ime i lozinka prosleđuju u zaglavlju SOAP poruke, a zatim Web Servis koristi ove podatke u procesu kastomizovane autentifikacije.

3. AUTORIZACIJA

.NET Framework i Windows platforma pružaju nekoliko tehnika za kontrolu pristupa sistemskim resursima. Resursi kojima se može pristupati su presek resursa za koje je korisnik

autorizovan od strane Microsoft Windows security sistema, kojima kod koji korisnik pokreće ima pravo pristupa prema *code access* zaštiti i, opciono, koju korisnik ima ulogu prema *role-based* zaštiti.

Direktorijum koji sadrži *root* .NET aplikacije je *root* logičkog *Universal Resource Identifier* (URI) *namespace*-a. ASP.NET aplikacija se može konfigurisati tako da se izvrše ograničenja pristupa aplikacionom URI zavisno od korisnikovog identiteta i uloge. Na primer, može se ograničiti pristup poddirektorijumima od aplikacionog *root*-a.

Prema Windows zaštiti, nakon što se izvrši autentifikacija korisnika, najveći deo koda koji korisnik pokreće ima pristup svim resursima kojima korisnik može da pristupa. Windows administrator može da kreira *discretionary access control* liste (DACL) koje kontrolišu pristup resursima ili objektima na mreži. DACL sadrže liste korisnika ili grupa korisnika kojima je dopušten pristup objektima kao što su fajlovi, štampači ili servisi. Slično tome, ukoliko se koristi *role-based* zaštita, prava pristupa se ne određuju prema identitetu korisnika već prema logičkoj ulozi koju ima u sistemu.

Pristup delovima ASP.NET aplikacionog URI može se ograničiti koristeći konfiguracioni fajl Web.config. Potrebno je u sekciji <*authorization*> ovog fajla navesti listu korisnika ili uloga ili oboje unutar *'allow'* i *'deny'* elemenata. Ovi elementi dodeljuju ili onemogućavaju pristup delovima URI.

Code access zaštita se koristi da bi se onemogućio pristup zaštićenim resursima. U tradicionalnim sistemima za autorizaciju, kod koji izvršava korisnik ima ista prava kao i korisnik. Ovaj pristup se pokazao kao nedovoljno siguran, zato što podrazumeva da je sav kod potpuno siguran u trenutku kada se izvršava. Postoji mnogo razloga zašto kod može biti nesiguran, na primer, može imati slabosti koje drugi, zlonameran kod (virus) može da iskoristi. Kod može da izvršava operacije kojih korisnik i nije svestan. Rešenje za ovaj problem je da se uspostavi mehanizam koji omogućava poverljivim korisnicima da izvršavaju nesiguran kod i takodje da se spreči da poverljivi kod slučajno ili namerno naruši sigurnost sistema i u .NET rešenje za ovaj problem daje *code access* zaštita. Prava, kao što je pristup fajlovima ili bazi, grupišu se u skupove i potom pridružuju *code access* grupama. Pre nego što se dopusti kodu da se izvrši, proverava se da li asembler pripada specificiranoj grupi koda. Ako se potvrdi pripadnost *code access* grupi, prava koja pripadaju ovoj grupi dodeljuju se asembleru. *Code access* zaštita uvodi pojam *evidence* kao skup informacija o identitetu i poreklu asemblera koje mogu uključivati:

- *Strong name* asemblera, koje se sastoji od jedinstvenog javnog ključa, imena i verzije
- Zone iz koje potiče asembler, kao što su lokalni računar, Internet ili Intranet
- Lokacije sa koje asembler potiče, koja može biti definisana kao URL ili lokalni direktorijum
- Kriptografski *hash* asemblera

Kada se asembler u toku izvršavanja *load*-uje, sakupljaju se *evidence* o asembleru i prezentuju *code access* sistemu za zaštitu. .NET *Framework* sadrži veliki broj *built-in* klasa za prava pristupa. Tako, na primer, *FileIOPermission* kontroliše pristup fajlovima i direktorijumima na fajl sistemu, *EnvironmentPermission* klasa kontroliše pristup *environmet* varijablama. Takođe, mogu da se implementiraju sopstvene klase, pomoću kojih se u *code access* sistemu za zaštitu definišu kastomizovana prava. Jedan od načina konfigurisanja *code access* sistema za zaštitu je definisanjem *security policy*. *Security policy* definiše hijerarhiju kodnih grupa i prava koja pripadaju *parent* kodnoj grupi moraju da budu u konjukciji sa dozvolama koje pripadaju samoj kodnoj grupi.

ASP.NET Web servisi izvršavaju se kao lokalne aplikacije. Web Services asembler prezentuje *code access* sistemu za zaštitu *evidence* da je 'Zona' jednaka 'My computer'. Ukoliko je na računaru default .NET *Framework* instalacija, prezentovanje ovog *evidence* dovodi do dodele asemblera *built-in* kodnoj grupi 'My computer Zone'. Ova kodna grupa ima skup dozvola sa imenom 'Full Trust', koji omogućavaju pun pristup resursima, koji se štite dozvolama. Međutim, najčešće se Web servisi smeštaju kod *Internet Service Provider*-a, koji neće dopustiti (konfigurisanjem *security policy*) pun pristup resursima. Zbog toga je potrebno da programer poznaje dozvole koje će njegov Web servis zahtevati. Postoji nekoliko načina pomoću kojih se može identifikaovati koje dozvole će kod zahtevati. Tu je najpre SDK dokumentacija koja sadrži listu potrebnih dozvola za svaku klasu. Zatim, moguće je testirati kod sa minimalnim skupom dozvola, kao što je 'Internet' skup dozvola, i zatim uočiti sve *Exceptions* koji se pojavljuju.

4. KRIPTOVANJE

Iako autentikacija i autorizacija onemogućavaju neovlašteni pristup resursima, ne mogu da spreče presretanje i čitanje podataka koji se prosleđuju između davaoca i korisnika Web servisa. Kriptovanje podataka se vrši da bi se obezbedio siguran transfer podataka.

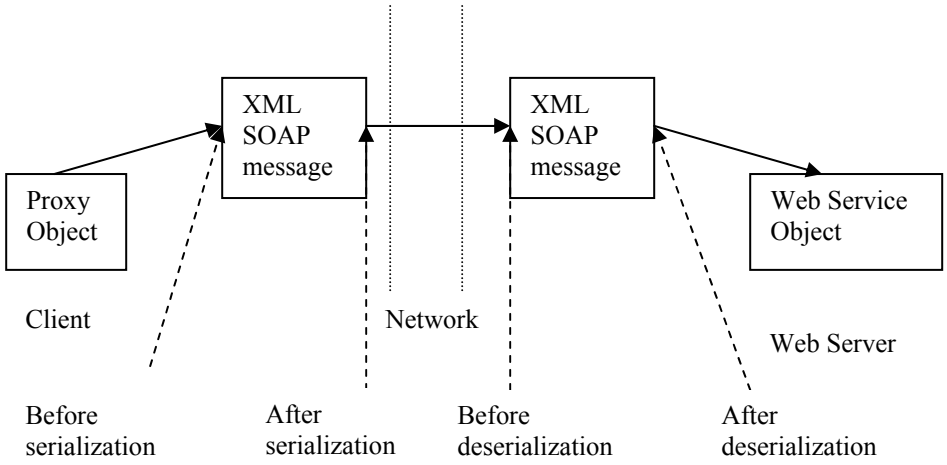
Kriptovanje je vremenski zahtevna operacija i zbog toga je potrebno biti pažljiv u izboru šta se kriptuje. Postoji nekoliko mogućnosti:

- Kriptovanje cele poruke. Iako se ovo može uraditi na relativno jednostavan način, performanse postaju veoma loše. Sa druge strane, malo je verovatno da svaka komunikacija zahteva apsolutnu privatnost.
- Kriptovanje samo tela poruke. Na ovaj način, postižu se bolje performanse nego u prvom slučaju, ali je često to i dalje više kriptovanja nego što je potrebno.
- Kriptovanje samo zaglavlja poruke. U Web servisima autentikacione informacije se najčešće prosleđuju u SOAP zaglavljima. Istovremeno, kriptovanje zaglavlja nije vremenski zahtevna operacija.
- Kriptovanje odabranih poruka. Ovaj način kriptovanja zahteva najviše programerskog rada, ali generalno

govoreći daje najbolji kompromis između performansi i zaštite.

- Bez kriptovanja. Ako podaci koji se šalju nisu osetljivi, treba izbeći slabljenje performansi do kojeg dovodi kriptovanje.
- Podela Web servisa. Ideja je da se interfejs servisa podeli u grupe poruka koje zahtevaju kriptovanje i koje ne zahtevaju kriptovanje. Tako podeljen interfejs može se implementirati koristeći dva Web servisa. Na ovaj način, mogu se zaštiti samo oni metodi koji to zahtevaju.

Dva najčešće korišćena načina za kriptovanje poruka u .NET Web servisima su: *Secure Sockets Layer* (SSL) i kastomizovane SOAP ekstenzije. Korišćenje SSL je relativno jednostavan način da se kriptuje celokupna komunikacija. Zahteva da se na klijentu ukonfigurišu privatni ključ i digitalni sertifikat koji sadrži javni ključ, a koji je dobijen od sertifikacionog tela. Korišćenje kastomizovanih SOAP ekstenzija omogućava veću kontrolu kriptovanja poruka. Da bi se koristile kastomizovane SOAP ekstenzije, potrebno je da se u projekat uključi *EncryptionExtension.dll*, i zatim da se u metodu koji se kriptuje navede kao atribut, parametar koji ukazuje da li se kriptuje zaglavlje ili telo poruke. SOAP poruka se kriptuje nakon serijalizacije i dekriptuje pre deserijalizacije (Slika 1.).



Sl.1. *Transformacije SOAP poruke*

5. ZAKLJUČAK

Web servisi se sve više nameću kao rešenje za distribuirane aplikacije. Međutim, rešenje pitanja zaštite u mnogome uslovljava njihovu primenu. U ovom radu opisano je rešenje koje daje .NET platforma. Opisana su rešenja za autentifikaciju, autorizaciju i kriptovanje. Cilj rada je bio da se ispitaju mogućnosti zaštite Web servisa, pre nego što se počne sa razvojem aplikacije.

LITERATURA

[1] WLDJ - Liquid Data XQuery-Based Enterprise Information Integration, http://sys-con.com/weblogic/articleprint.cfm?id=241

[2] Developing XML Web Services using Microsoft ASP.NET, Course Book

Abstract – This paper presents solutions for authentication, autorisation and encryption of Web Services in .NET environment. Paper aimed to examine Web Services security before developing Web Services based application.

WEB SERVICES SECURITY IN .NET ENVIRONMENT

Snezana Sucurovic