INFOTEH-JAHORINA Vol. 9, Ref. E-I-15, p. 482-486, March 2010.

TIPOVI NAPADA NA WEB SERVISE
COMMON TYPES OF ATTACKS ON WEB SERVICES

Ognjen Joldžić, *Elektrotehnički fakultet Banjaluka, Univerzitet u Banjaluci*
Zoran Đurić, *Elektrotehnički fakultet Banjaluka, Univerzitet u Banjaluci*

Sadržaj: *U ovom radu su analizirani sigurnosni problemi kod implementacije web servisa u informacionim sistemima. Prikazani su najpoznatiji sistemi za kategorizaciju sigurnosnih prijetnji za web servise. Opisani su najzastupljeniji tipovi konkretnih napada, način njihovog izvođenja, sigurnosni rizici i posljedice uspješnog izvođenja svakog od napada. Za svaki prikazani napad je opisan način za otklanjanje propusta i minimizaciju mogućnosti gubitka podataka ili oštećenja sistema.*

Abstract: *This paper presents an analysis of security issues when deploying web services for information systems. Some of the categorizations and catalogues for security threats aimed against web services are explained. The paper also describes a number of attacks against web services, means of their execution, security risks and finally, the possible consequences coming from a successful execution. For each of the respective attacks, a solution that should minimize the possibility of data corruption or any other system damage is presented in this paper.*

1. UVOD

Web servisi predstavljaju tehnologiju koja je u posljednje vrijeme stekla veliku popularnost u modernim informacionim sistemima. Funkcionalnosti koje ova tehnologija donosi predstavljaju ključne segmente većine web aplikacija i značajno povećavaju iskoristivost kompletnog informacionog sistema.

Imajući prethodno navedeno u vidu, da bi se osigurala interoperabilnost i dostupnost web servisa, jedan od najvažnijih uslova je visok nivo sigurnosti ovakvih aplikacija. Web servisi sa klasičnim web aplikacijama dijele jedan dio sigurnosnih rizika koji moraju biti riješeni na isti način kao i kod klasičnih web aplikacija, ali donose i čitav niz novih potencijalnih ranjivosti. Ove ranjivosti web servisa proizilaze iz arhitekture na kojoj su web servisi izgrađeni, a koja se u određenoj mjeri razlikuje od arhitekture klasičnih web aplikacija.

U ovom radu će biti prikazani najvažniji aspekti sigurnosti web servisa, kao i najpoznatije grupe napada usmjerenih prema servisno-orijentisanim aplikacijama. Za svaki opisani tip napada će biti prikazan i efikasan način zaštite web servisa i načini za uklanjanje sigurnosnog rizika koji svaki od spomenutih napada povlači za sobom.

2. OSOBINE I TIPOVI NAPADA NA WEB SERVISE

Zajednička osobina svih napada usmjerenih prema web servisima je da za cilj imaju smanjenje funkcionalnosti i dostupnosti servisa (u krajnjem slučaju i njegovo potpuno onesposobljavanje), te pribavljanje neovlaštenog pristupa zašićenim podacima ili dijelovima sistema. Veliki broj napada koji će biti opisani u sljedećim sekcijama nije specifičan samo za web servise, već se aktivno koristi i kod „klasičnih“ web aplikacija, pa su i tehnike zaštite analogne. Međutim, postoji određen broj napada koji koristi specifičnosti tehnologije servisno-orijentisanih sistema, i koji u skladu s tim ima i nešto uži spektar primjene.

Najčešći način grupisanja napada na web servise je prema načinu izvođenja i zajedničkim osobinama različitih napada. Prema tome, svi napadi na web servise se mogu svrstati u neku od tri osnovne grupe:

- napadi sa izmjenama XML (*eXtensible Markup Language*) sadržaja,
- direktni napadi na web servise i
- napadi na elemente infrastrukture servisno-orijentisanih aplikacija.

2.1. NAPADI SA IZMJENAMA XML SADRŽAJA

U ovoj grupi se nalaze svi napadi kod kojih se putem izmjene korisnog sadržaja XML poruke nastoji izazvati prekid dostupnosti funkcija web servisa (tipično *Denial-of-Service*), ili izazvati izvršenje malicioznog koda, kako na aplikativnom serveru, tako i na krajnjem sistemu sa kojim web servis komunicira. U ovu kategoriju spadaju sljedeći napadi i grupe napada: *Coercive parsing*, *Buffer overflow, External entity, Parameter tampering, Input validation* i *Error handling*.

Coercive parsing je napad koji koristi nedostatke u procesu parsiranja XML sadržaja od strane parsera ugrađenih u funkcionalnosti programskih jezika. U pravom smislu riječi, *Coercive Parsing* ne predstavlja izmjenu originalnog XML sadržaja, već vrši eksploataciju neadekvatno isprojektovanih web servisa. Provedeni testovi [2] pokazuju da procesorsko opterećenje prilikom parsiranja XML sadržaja kod kojih pojedini čvorovi dostižu dubinu od nekoliko nivoa (ili sa zahtjevima koji koriste veliki broj *namespace* deklaracija u svom zaglavlju), značajno raste, dostižući nerijetko i 100% na ciljnom sistemu. Kod takvih sistema, konstantna eksploatacija ovog nedostatka može dovesti do značajnog smanjenja dostupnosti funkcija web servisa i neefikasne potrošnje resursa. Nedostatak ovakve vrste se otklanja izmjenama u strukturi web servisa, odnosno smanjenjem kompleksnosti zahtjeva, kojim se omogućava efiksano parsiranje bez opasnosti po integritet sistema.

Buffer overflow je napad koji nije usko vezan za web aplikacije, već predstavlja rizik na svim sistemima kod kojih nije izvršena ispravna validacija ulaznih parametara servisa. Ovaj napad je kod web servisa u određenoj mjeri olakšan, jer potencijalnom napadaču na raspolaganju stoji WSDL specifikacija servisa koja propisuje očekivane parametre i njihove tipove za pozive pojedinih funkcija. Iz tog razloga je jednostavnije izvršiti usmjeravanje napada prema onim parametrima za koje postoji najveća vjerovatnoća pozitivnog rezultata za napadača. Numerički parametri su najpodložniji ovakvom napadu, dok se kod alfanumeričkih promjenljivih pokazuje velika tolerancija sadržaja ulaznih parametara (pojedini sistemi za parsiranje omogućavaju obrađivanje parametara dužih i od 1,000,000 znakova [3]).

External entity – external entity referenciranje predstavlja funkcionalnost XML infrastrukture koja omogućava ugrađivanje sadržaja koji potiču izvan originalnog XML dokumenta direktno u taj dokument. Ovo, jasno, otvara mogućnost zamjene originalno referenciranog sadržaja malicioznim kodom koji bi mogao rezultovati prekidom funkcionisanja web aplikacije ili kompletnog sistema [4]. Propusti ovog tipa se relativno uspješno rješavaju ograničavanjem pristupa za referencirane sisteme na entitete od povjerenja. Pozitivna praksa je kombinovanje dodatnih tehnika za ograničavanjem eventualnog *Man-in-the-Middle* napada, kako na aplikativnom sloju, tako i na ostalim slojevima OSI (*Open System Interconnection*) modela, gdje je to moguće.

Parameter tampering i *Input validation* – ovo su, vjerovatno najzastupljenije grupe napada koje su generalno primjenjive na sve web aplikacije. Slično kao i za buffer overflow, činjenica da su parametri dostupni kroz WSDL specifikaciju samo otežava posao obezbjeđivanja web servisa od neželjenih efekata izazvanih potencijalnim napadom. Iz širokog opsega mogućih načina izvođenja napada izdvajaju se: slanje specijalnih znakova koji imaju posebno značenje unutar XML specifikacije (tipično *XMLinjection*), sa ciljem da se izazove greška prilikom parsiranja sadržaja; te slanje posebno pripremljenih upita usmjerenih protiv sistema sa kojim web servis komunicira (tipično baza podataka; najzastupljeniji napadi iz ove podgrupe su *SQLinjection* [5] i *LDAPinjection*). Potrebno je naglasiti da pojedini parseri već imaju ugrađene tehnike za eliminisanje rizika od *XMLinjection* napada, čime ovaj napad na takve sisteme postaje relativno neupotrebljiv [6]. Kod *SQLinjection* napada takav sistem zaštite ne postoji, jer je sa stanovišta web servisa sadržaj parametara potpuno validan, s obzirom da se radi o napadu koji je usmjeren na pozadinski sistem.

Error handling – ova grupa napada kao rezultat ima informacije koje se mogu saznati o strukturi sistema u pozadini web servisa ili o prirodi greške kroz analizu povratnih vrijednosti pozvanih funkcija. Ove greške mogu uključivati podatke o bazi podataka (greške tipa: *ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)*), podatke o detaljima implementacije, programskom jeziku, web serveru ili operativnom sistemu koje bi omogućile napadaču da iskoristi napade specifične za datu platformu. Zajedničko rješenje za sve napade ovog tipa je obrada svih povratnih vrijednosti na ispravan način (kako bi se poruke o greškama vidjele samo tamo gdje su potrebne i bile dostupne samo administrativnom osoblju koje bi moglo analizirati i otkloniti eventualne probleme).

2.2. DIREKTNI NAPADI NA WEB SERVISE

Kao najvažniji napadi na web servise mogu se izdvojiti WSDL *scanning* i *Schema poisoning*.

WSDL *scanning* napad bazira se na otvorenoj prirodi WSDL dokumenta. WSDL [12] dokument je XML baziran dokument koji sadrži opis web servisa. Ovaj opis uključuje sve neophodne elemente za konzumaciju opisanog web servisa, kao što su: URL adresa web servisa, nazivi operacija web servisa, tipovi parametara operacija i njihovih povratnih vrijednosti. WSDL dokument se koristi od strane alata za automatsko generisanje programskog koda koji omogućava komunikaciju sa web serisom. Kao što se koristi od strane legitimnog korisnka, WSDL dokument može biti korišten i od strane potencijalnog napadača. Pored toga, detalji sadržani u WSDL dokumentu mogu poslužiti za otkrivanje metoda koje web servis posjeduje, a koje nisu opisane u WSDL dokumentu. Pogađanje naziva metoda koje nisu opisane u WSDL dokumentu, a koje web servis posjeduje, posebno dolazi do izražaja u situaciji kada je u WSDL dokumentu opisan veći broj metoda web servisa, te je na taj način izložen i obrazac imenovanja metoda. Zaštita od ove vrste napada može se izvesti ograničavanjem pristupa WSDL dokumentu, odnosno odgovarajućom autentikacijom. Ovakav način zaštite nije djelotvoran u slučaju kada legitimni, autentikovani korisnici pokušavaju da otkriju metode koje web servis posjeduje, a koje nisu opisane u WSDL dokumentu.

Schema poisoning napad bazira se na izmjeni XML schema datoteke. XML parseri koriste XML schema datoteku kako bi mogli razumjeti gramatiku i strukturu XML datoteke, odnosno kako bi izvršili validaciju XML datoteke. U kontekstu web servisa termin XML datoteka se odnosi na SOAP poruku. Cilj izmjene XML schema datoteke, od strane potencijalnog napadača, jeste omogućavanje procesiranja malicioznih SOAP poruke od strane parsera. Na ovaj način otvara se mogućnost ubacivanja posebno kreiranih SOAP poruka koje mogu dovesti do izvršavanja različitih komandi na aplikativnom serveru ili sistemu za upravljanje bazom podataka koju web servis potencijalno koristi.

2.3. NAPADI NA INFRASTRUKTURU WEB SERVISA

Postoji veoma velika sličnost web servisa i web aplikacija kada je u pitanju infrastruktura na kojoj se izvršavaju, kao i infrastruktura koju koriste pri izvršavanju. Na primjer, pristup web aplikaciji se ostvaruje korištenjem HTTP ili HTTPS protokola. Ovi protokoli koriste se i za prenos SOAP poruka između klijenta i web servisa. Isto tako, i web aplikacija i web servis obično ostvaruju pristup bazi podataka, odnosno sistemu za upravljanje bazom podataka. Na ovaj način, brojni napadi na web aplikacije primjenljivi su i na web servisi, na potpuno identičan način ili uz njihovu neznatnu izmjenu.

Kao najvažniji napadi na infrastrukturu web servisa mogu se izdvojiti napadi na autentikaciju i autorizaciju, napadi na web/aplikativne servere, MITM (*Man in the Middle*) i DoS (*Denial of Service*) napadi, pri čemu prve tri vrste napada nemaju bilo kakvu specifičnost koja je jedinstvena za web servise.

DoS napadi su napadi koji rezultiraju smanjenjem dostupnosti servisa ili njegovom potpunom nedostupnošću. Najznačajniji DoS napadi na web servise, ali i druge XML bazirane servise, su: *oversize payload* i *coercive parsing*. Coercive parsing napad je već objašnjen u sekciji 3.1.

Oversize payload napad [8] se izvodi relativno jednostavno zbog činjenice da određeni XML parseri koriste veliku količinu radne meorije pri procesiranju XML poruka.Tako količina memorije potrebna za parsiranje jedne SOAP poruke može biti mnogo veća od veličine same poruke. Ovo je posebno izraženo kod web servis okruženja koja koriste DOM [7] parsere, koji čitaju kompletnu SOAP poruku, parsiraju je i transformišu u objektnu reprezentaciju u radnoj memorijiv (DOM stablo). Iz ovih razloga ekstremno velike SOAP poruke mogu dovesti do „potrošnje“ radne memorije koja je dodijeljena datom procesu. Čak i u slučaju kada web servis okruženje koristi SAX parser [10] i dalje postoji mogućnost za ovu vrstu napada. XML schema element [11] posjeduje atribut maxOccurs čijom vrijednošću se definiše maksimalan broj pojavljivanja datog elemenata u XML dokumentu. Kod većine alata za generisanje WSDL dokumenta iz izvornog koda (na primjer, Java2WSDL) vrijednost ovog atributa za nizove se automatski postavlja na „unbounded“. Ovakva deklaracija dozvoljava SOAP poruke koje sadrže neograničen broj elemenata. Jasno je da ovakve poruke mogu dovesti do uspješne realizacije *oversize payload* napada. Ovaj tip napada se može spriječiti ograničavanjem maksimalne veličine dolazne SOAP poruke, pri čemu se sve veće poruke odbacuju.

3. MEHANIZMI ZA UNAPREĐIVANJE SIGURNOSTI WEB SERVISA

Postoji niz tehnika koje se koriste u cilju poboljšanja sigurnosti web servisa. Počevši od samog web servisa, kao atipične web aplikacije, u cilju sigurnosti sistema je ispravljanje svih eventualnih propusta na nivou samog

programskog koda, čime se u startu sigurnost kompletnog informacionog sistema podiže na znatno viši nivo.

S druge strane, u slučajevima u kojima nisu moguće izmjene u strukturi aplikacije iz bilo kojeg razloga, na raspolaganju su rješenja koja ne zavise od aplikativne platforme. Ova rješenja su primjenjiva na sve web aplikacije bez obzira na njihovu namjenu i način implementacije.

U prvoj grupi se nalaze *firewall* rješenja koji rade na aplikativnom sloju OSI referentnog modela. Iako se primjenom aplikativnog *firewall*-a web aplikaciju može zaštititi od poznatih napada, poput SQL *injection* napada, potrebno ih je dodatno konfigurisati kako bi bile zadovoljene specifične potrebe aplikacije. Napadi na web aplikacije su znatno raznovrsniji i komplikovaniji od napada na mrežne čvorove, te je samim tim i konfiguracija *firewall* rješenja koje radi na aplikativnom nivou složeniji proces nego konfiguracija mrežnog *firewall* uređaja. Mrežni *firewall* uređaj operiše sa relativno malim brojem jednostavnih entiteta, poput IP adresa i portova, dok aplikativni *firewall* operiše sa svim tipovima ulaza koje web aplikacija može da prihvati. Da bi se web aplikacija u potpunosti zaštitila korištenjem aplikativnog *firewall*-a potrebno je identifikovati sve ulazne tačke aplikacije i tipove podataka koji su dozvoljeni na tim ulaznim tačkama. Zadatak aplikativnog *firewall*-a dodatno je usložnjen postojanjem brojnih načina generisanja jednog napada. Broj načina generisanja jednog napada može biti višestruko uvećan primjenom tehnika kodovanja na maliciozni ulaz. Pored toga, postoje i napadi koji su specifični za samu web aplikaciju. Primjer takvog napada je smanjenje cijene poručenog proizvoda u web prodavnici. Ovakav, i slične napade, aplikativni *firewall* ne može detektovati jer ne poznaje poslovnu logiku aplikacije.

Kada se govori o načinima zaštite primjenjivim na sve web servise, prvenstveno se misli na nekoliko dodatnih specifikacija koje pripadaju grupi WS-Security. Ove specifikacije propisuju sigurnosne mehanizme koji nisu vezani za korisni sadržaj SOAP poruke, već utvrđuju način prenosa poruka sigurnim kanalima ili način međusobnog potvrđivanja identiteta učesnika u komunikaciji. Većina standarda u ovoj grupi je razvijena i specificirana od strane IBM-a, a neki od najznačajnijih su:

- autentifikacija – ovom tehnikom se omogućava filtriranje klijenata koji imaju pravo pristupanja funkcionalnostima web servisa. U okruženjima u kojima web servis predstavlja internu aplikaciju uske oblasti primjene, ova nadogradnja na osnovnu specifikaciju omogućava dodatnu zaštitu resursa servisa od neovlaštene upotrebe. U opštem slučaju, u tijelu SOAP poruke se nalazi dodatak na osnovu kojeg primalac može jednoznačno utvrditi identitet pošiljaoca, bilo da se radi o digitalnom potpisu ili nekom drugom načinu potvrđivanja autentičnosti. Autentifikaciju omogućava specifikacija pod nazivom WS-Trust [14],
- XML Signature – predstavlja W3C preporuku koja sadrži XML sintaksu za rad sa digitalnim potpisima. Ovaj standard je poznat i pod nazivima *XMLDsig*, *XML-DSig* i *XML-Sig*. Iako se funkcionalnost digitalnog potpisa kod XML-a bazira na PKCS tehnologiji, zbog prirode samog XML-a postoje neke razlike u odnosu na uobičajeni postupak generisanja digitalnog potpisa. Razlike su najočiglednije kod rada sa *hash* algoritmima, jer se prema XML specifikaciji prazni karakteri zanemaruju, ali utiči na sadržaj *hash* otiska, i samim tim i digitalnog potpisa,
- WS-Trust – ekstenzija na WS-Security standard koji propisuje procedure za ostvarivanje sigurne komunikacije za razmjenu SOAP poruka [14]. Model razmjene podataka koji definiše WS-Trust omogućava web servisu da odbaci svaku poruku koja ne ispunjava standarde definisane sigurnosnom politikom ustanovljenom prilikom implementacije,
- WS-Policy – ekstenzija koja omogućava definisanje sigurnosne politike za ostvarivanje sigurne komunikacije između web servisa i klijenata. Sigurnosna politika može uključivati sve aspekte komunikacije, od izbora transportnog protokola i načina autentikacije, do QoS podešavanja [16]. WS-Policy omogućava jednoznačnu notaciju za definisanje, bez obzira na prirodu sigurnosne politike,
- WS-SecureConversation [13] – nadogradnja na osnovnu specifikaciju slična TLS-u, koja propisuje tehnike generisanja i razmjene ključeva u cilju kreiranja sigurnog kanala za prenos poruka,
- XML enkripcija – minimalni sigurnosni zahtjevi koje preporučuje W3C [15] sadrži popis sigurnosnih algoritama minimalne jačine i kompleksnosti koji bi se morali koristiti pri enkripciji XML sadržaja SOAP poruke da bi se ostvario dovoljan nivo sigurnosti komunikacije.

4. ZAKLJUČAK

Web servisi predstavljaju specifičnu grupu web aplikacija i u kompletnom informacionom sistemu često imaju dvojaku ulogu: sa jedne strane treba krajnjem korisniku da stave veliki broj funkcionalnosti sistema na direktno raspolaganje, dok istovremeno moraju osigurati

sistem od zloupotrebe. Upravo iz tog razloga se aspektu sigurnosti web servisa mora pokloniti posebna pažnja kako bi se eliminisao bilo kakav rizik od gubitka podataka ili doveo u pitanje integritet kompletnog sistema.

U ovom radu dat je pregled najvažnijih napada na web servise, i to: napada sa izmjenama XML sadržaja, direktnih napada na web servise i napada na infrastrukturu web servisa. Pored toga, u ovom radu opisani su i mehanizmi za unapređivanje sigurnosti web servisa.

5. LITERATURA

[1], http://cve.mitre.org/

[2], Asurvey of attacks onweb services, Classification and countermeasures, Meiko Jensen · Nils Gruschka · Ralph Herkenhoener

[3], http://codeidol.com/html/effective-xml/Choose-SAX-for-Computer-Efficiency/Choose-SAX-for-Computer-Efficiency/

[4], http://www.securiteam.com/securitynews/6D0100A5PU.html

[5], Sigurnost web aplikacija, Zoran Đurić, Ognjen Joldžić, Infofest 2009.

[6], http://java.sun.com/j2se/1.4.2/docs/api/javax/xml/parsers/SAXParser.html

[7], Hors AL, Hegaret PL, Wood L, Nicol G, Robie J, Champion M, Byrne S (2004) Document Object Model (DOM) Level 3 Core Specification, W3C Recommendation, http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/

[8], Lindstrom P (2004) Attacking and Defending Web Service. A Spire Research Report

[9], Brett McLaughlin. Java and XML Data Binding. O Reilly, 2002.

[10], SAX Project, http://www.saxproject.org/

[11], H.S. Thomson et al. XML Schema Part 1: Structures Second Edition, W3C Recommendation, http://www.w3.org/TR/xmlschema-1/

[12], Erik Christensen et al. Web Services Description Language (WSDL), W3C Note, http://www.w3.org/TR/wsdl

[13] http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-secon/ ws-secureconversation.pdf

[14] http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html

[15] http://www.w3.org/TR/xmlenc-core/#sec-Algorithms

[16] http://www.w3.org/Submission/2006/SUBM-WS-Policy-20060425/