Zraniteľnosti druhého rádu vo webových aplikáciách

AUTOR: OLEH SOVA

VEDÚCI PRÁCE: RNDR. JUDR. PAVOL SOKOL

KONZULTANT PRÁCE: MGR. TERÉZIA MEZEŠOVÁ

Motivácia práce

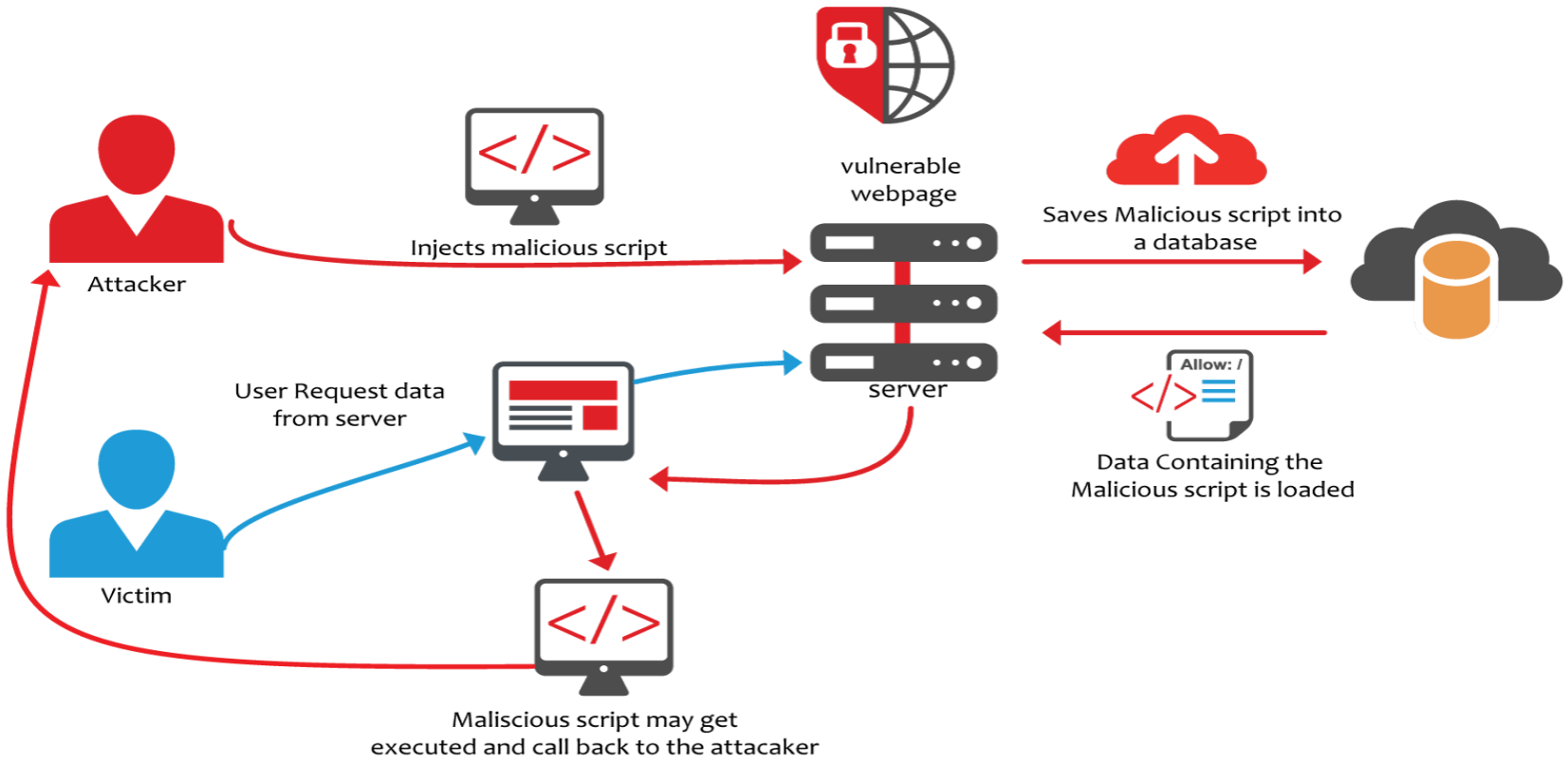
Webové aplikácie vytvorené v posledných rokoch (od jednostránkových až po multifunkčné aplikácie) získali veľkú popularitu v akejkoľvek sfére ľudskej činnosti. Táto rastúca popularita robí z webových aplikácií ciele pre mnohé typy útokov (napríklad: XSS, SQLi, DOS). Rastúci počet útokov vedie k vývoju výskumu v oblasti bezpečnosti webových aplikácií.

Takéto štúdie v zásade dokazujú pôvod zraniteľností v zdrojovom kóde a jeho štruktúre, a poukazujú na potrebu používať automatické nástroje na overovanie kódu, aby sa minimalizoval výskyt zraniteľností.

Zraniteľnosti druhého rádu vznikajú vtedy, keď aplikácia ukladá vstup do databázy bez overenia voči škodlivým príkazom a potom sa z databázy zobrazí používateľom stránky, alebo použije pri vnútornej operácií.

Jednou z takých zraniteľností je stored cross-site scripting (XSS). (ilustrácia zobrazené na ďalšom snímku)

Stored Cross-Site Scripting



Link pre obrázku

Stored Cross-Site Scripting

Útočník (hacker) nahrá škodlivý skript na aplikačný server, ktorý sa uloží do databázy (aplikácia ho považuje za štandardný vstup od užívateľa). Pri následných požiadavkách servera na databázu, databáza bude serveru vracať dáta obsahujúce tento škodlivý skript. Používateľ (obeť) tak pri kontakte s aplikáciou prijíma na svojom zariadení dáta, ktoré sú napadnuté alebo infikované škodlivým skriptom.

Potom útočník môže v závislosti od toho, na čo bol navrhnutý skript, získať kontrolu na dátami obete, alebo nad zariadením

Ciele práce

1. Analyzovať zraniteľnosti druhého rádu vo webových aplikáciách a možnosti ich detekcie.
2. Analyzovať a porovnať aktuálne prístupy k detekcii zraniteľností druhého rádu vo webových aplikáciách.
3. Návrh, implementácia a otestovanie nástroja pre testovanie zraniteľností druhého rádu pre webové aplikácie vytvorené v jazyku Javascript.

Postup práce

- analýza zraniteľností druhého rádu;
- vývoj automatizovaného systému na kontrolu štruktúry kódu, respektíve implementácia modulu pre existujúci systém na kontrolu zdrojových kódov v jazyku JavaScript;
- overiť účinnosť systému.

Analýza zraniteľností druhého rádu

Podľa populárnosti útokov na rôzne webové prílohy môžeme vydeliť tri základných kategórií zraniteľností, ktoré sa využívajú na rôzne ciele, od získania hesla poštovej skrinky do riadenia a zastavenia veľkých webových portálov:

- *Denial of Service(DoS)* útok
- *Cross-Site Scripting(*XSS) útok
- SQL injekcia

Denial of Service(DoS) útoká

Útoky DoS druhého radu sa stávajú možnými vďaka existencii skrytých zraniteľností bezpečnosti vo webových prílohách. Prvá etapa útoku je možná, pretože webová´príloha nepoužíva zodpovedajúci ochranný mechanizmus, ako napr. CAPTCHA, ktorý znemožňuje používateľom zapisovať cez bot. Teda, druhá časť útoku je možná, pretože príloha nedeaktivuje získanie databáz (napríklad, ohraničí ich veľkosť). Okrem toho, DoS útokom nemôžeme predísť použitím štandardných mechanizmov ochrany na základe siete.

.

Pretože majú zapísané databázy väčšinou malú veľkosť a sú dočasne rozdelené, DoS útoky druhého radu používajú malú prepúšťaciu schopnosť a môžu sa skrývať takým spôsobom, ktorý kontroluje anomálnu sieťovú kapacitu.

Cross-Site Scripting(XSS) útoká

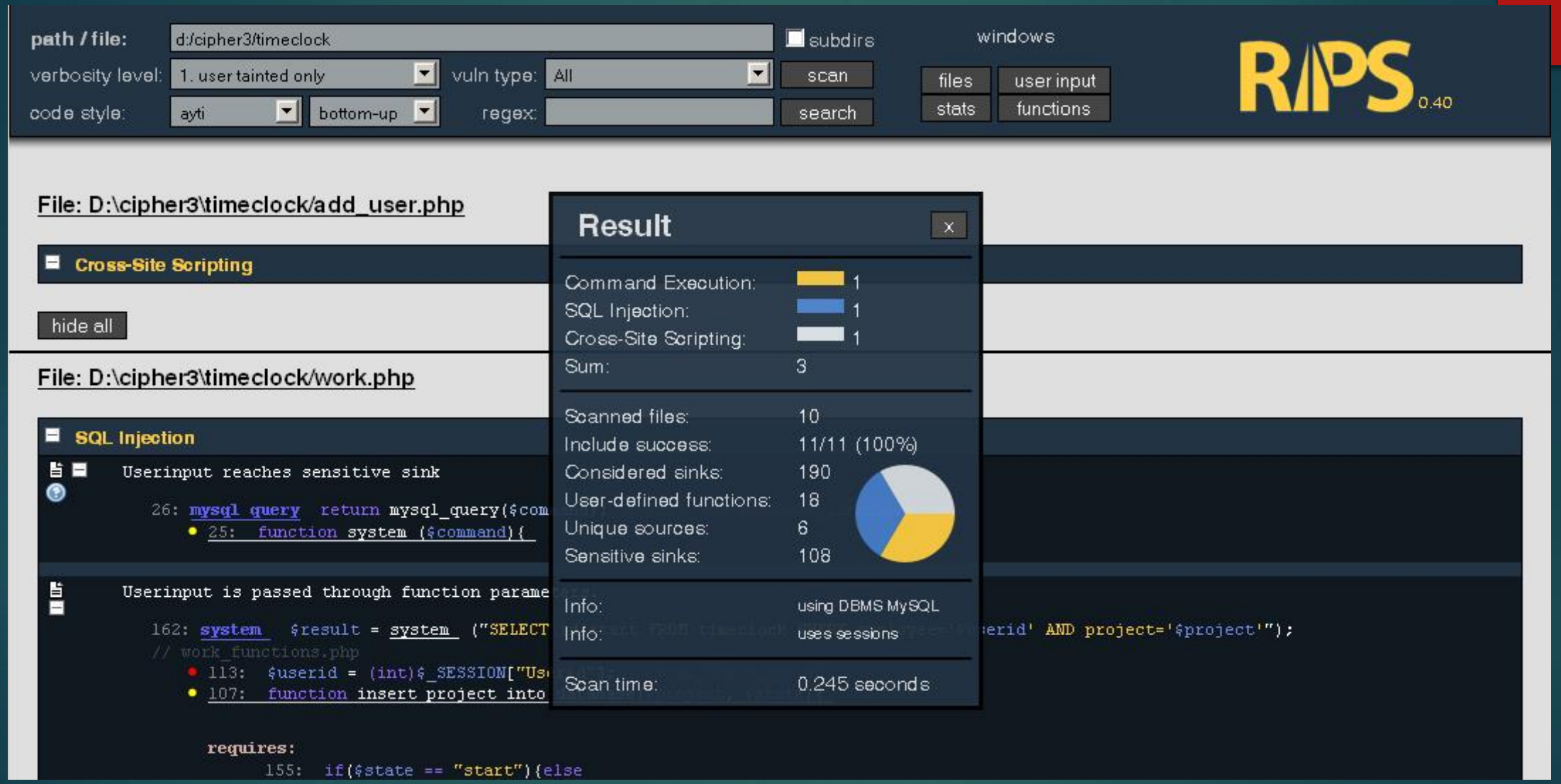
Špecifickosť takýchto útokov spočíva v tom, že škodlivý kód môže používať autorizáciu používateľa vo webovom systéme na získanie rozšíreného prístupu k nemu alebo na získanie údajov o autorizácii používateľa. Škodlivý kód môže byť vložený do stránky buď prostredníctvom zraniteľnosti na webovom serveri alebo prostredníctvom zraniteľnosti v počítači užívateľa.

SQL injekcia

Zraniteľnosť SQL injekcia (SQLi) vzniká, keď web príloha dynamicky generuje SQL -požiadavku nekontrolovaným vpísaním užívateľa. Tu môže škodca potenciálne zaviesť svoj vlastný syntax SQL pre samostatnú zmenu požiadavky. V závislosti od prostredia môže škodca potenciálne vyberať dôležité údaje z databázy, zmeniť údaje alebo kompromitovať web server.

Zisťovanie zraniteľností druhého rádu

Jedným z najpopulárnejších nástrojov analýzy statického kódu na automatické zisťovanie zraniteľností vo webových prílohách je RIPS. V nasledujúcich bodoch budú popísané zásady analýzy, v ktorej sa používa RIPS.


path / file:
d:/cipher3/timeclock
subdirs
windows
verbosity level:
1. user tainted only
vuln type:
All
scan
files
user input
code style:
ayti
bottom-up
regex:
search
stats
functions
RIPS 0.40
File: D:\cipher3\timeclock/add_user.php
Cross-Site Scripting
hide all
File: D:\cipher3\timeclock/work.php
SQL Injection
Userinput reaches sensitive sink
26: mysql_query return mysql_query($com
25: function system ($command){
Userinput is passed through function parame
162: system $result = system ("SELECT
// work_functions.php
113: $userid = (int)$_SESSION["Us
107: function insert_project into
erid' AND project='$project'");
requires:
155: if($state == "start"){else
Result
x
Command Execution: 1
SQL Injection: 1
Cross-Site Scripting: 1
Sum: 3
Scanned files: 10
Include success: 11/11 (100%)
Considered sinks: 190
User-defined functions: 18
Unique sources: 6
Sensitive sinks: 108
Info: using DBMS MySQL
Info: uses sessions
Scan time: 0.245 seconds

RIPS používa kontextovú aj procesnú analýzu toku údajov. Daný nástroj používa základné funkčné a sumárne reporty [30] pre efektívnu analýzu spätného toku údajov [6]. Po prvé, pre zdrojový kód sa generuje riadiaci potok (CFG), vytvorený zo spojených základných blokov, ktorý určuje funkcie, rady a metódy v kóde. Potom sa každý CFG analyzuje zhora dole, napodobňujúc spojené bázové bloky jeden za druhým.

Automatizovani systémy na kontrolu kodu

Zeon.js console

Putting the semi colon back in js

```
parse time: 1 ms
post process time: 4 ms (2, 0, 0, 0)
output generation time: 1 ms
display time: 2 ms
```

```
1
```

Literatúra

- 1) DAHSE, Johannes; HOLZ, Thorsten. Static Detection of Second-Order Vulnerabilities in Web Applications. In: USENIX Security Symposium. 2014. p. 989-1003.
- 2) OLIVO, Oswaldo; DILLIG, Isil; LIN, Calvin. Detecting and exploiting second order denial-of-service vulnerabilities in web applications. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM, 2015. p. 616-628.

Ďakujem za pozornosť!