Zadanie 0
I-UPB: Úvod do počítačovej bezpečnosti

September 27, 2021

1 Úvod

Zadanie demonštruje realizáciu rôznych útokov na operačný systém a webové aplikácie bežiace v ňom. Informácie ako služby bežiace na cielenom OS a ch verzie umožňujú útočníkovi jednoducho zistiť známe zraniteľnosti a využiť ich v jeho prospech.

2 Realizácia útokov

2.1 Prieskum

Úlohou je zistiť IP adresu cieleného OS a služby bežiace v ňom.
Po spustení scanu na sieťovom rozhraní eth0 sme zistili IP adresu cieleného OS a boli zistené otvorené porty ktoré sú potencionálnym slabím miestom cieleného OS:

```
Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-23 09:56 EDT
Nmap scan report for 192.168.56.101
Host is up (0.50s latency).
Not shown: 96 filtered ports
PORT    STATE  SERVICE
22/tcp  open   ssh
25/tcp  open   smtp
80/tcp  open   http
631/tcp closed ipp
```

Ďalej vieme spustiť komplexnejší scan ktorý zistí OS, jeho verziu spolu s názvami a verziami služieb bežiacich na otvorených portoch.

```
sudo nmap -sV -O -PN 192.168.56.101
```

Výstup príkazu nám dal informácie o OS a službách na ňom bežiacich.

```
Nmap scan report for 192.168.56.101
Host is up (0.00098s latency).
Not shown: 996 filtered ports
PORT    STATE  SERVICE VERSION
22/tcp  open   ssh     OpenSSH 4.3 (protocol 2.0)
25/tcp  open   smtp    Sendmail 8.13.5/8.13.5
80/tcp  open   http    Apache httpd 2.2.0 ((Fedora))
631/tcp closed ipp

Running (JUST GUESSING): Linux 2.6.X|3.X|4.X (98%)
```

Zistili sme že OS je Linux a na OS beží webový server, mailový server a je možné sa ku OS pripojiť vzdialene cez protokol SSH. Webový server je Apache server takže môžeme predpokladať že je na ňom nasadená PHP aplikácia a zároveň vieme aj verziu Apache spolu s tím že distribúcia Linuxového OS je Fedora. Na základe týchto informácií môžeme ďalej smerovať náš útok.

2.2 Skenovanie zraniteľností

Pomocou nástroja Nikto sme dokázali zistiť veľmi jednoducho hneď veľa zaujímavých informácií o webovej aplikácií bežiacej v cielenom OS.

```
---------------------------------------------------------------------------
+ Target IP:          192.168.56.101
+ Target Port:        80
---------------------------------------------------------------------------
+ Server: Apache/2.2.0 (Fedora)
+ Retrieved x-powered-by header: PHP/5.1.2
+ The X-XSS-Protection header is not defined. This header can hint to the user
↪  agent to protect against some forms of XSS
+ Server may leak inodes via ETags, header found with file /robots.txt, inode:
↪  487720, size: 104, mtime: Tue Dec  9 18:39:44 2014
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE
+ Web Server returns a valid response with junk HTTP methods, this may cause
↪  false positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to
↪  XST
+ OSVDB-5034: /admin/login.php?action=insert&username=test&password=test:
↪  phpAuction may allow user admin accounts to be inserted without proper
↪  authentication. Attempt to log in with user 'test' password 'test' to verify.
+ OSVDB-682: /usage/: Webalizer may be installed. Versions lower than 2.01-09
↪  vulnerable to Cross Site Scripting (XSS).
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals
↪  potentially sensitive information via certain HTTP requests that contain
↪  specific QUERY strings.
+ OSVDB-3093: /mail/src/read_body.php: SquirrelMail found
```

```
+ OSVDB-9624: /admin/admin.php?adminpy=1: PY-Membres 4.2 may allow administrator
↪  access.
---------------------------------------------------------------------------
```

Výstup nám poskytol informácie o verzií jazyka PHP ktorý je použitý vo webovej aplikácii a tak isto nás aj upozornil že aplikácia môže byť náchylná na XSS a XST útok, SQL injection útok a fakt že na URL /admin/admin.php je možné získať administrátorský prístup ku webovej aplikácií. Zároveň sme boli upozornený že súbor robots.txt ktorý je určený pre Googlacke webscraper-y môže obsahovať odkazy na priečinky na súborovom systéme ktoré väčšinou obsahujú citlivé údaje alebo prostriedky ako realizovať útok.

2.2.1 Manuálne odhalenie zraniteľností

Nástroj Nikto nám poskytol návrhy na potencionálne zraniteľné miesta webovej aplikácie ktoré ako útočník musíme manuálne preskúmať. Ideálne je aby sme mali zaznamenaný každý HTTP request a mohli tieto dáta analyzovať. Aby sme dokázali odchytávať HTTP requesty tak si vieme nadstaviť lokálnu proxy pomocou nástroja Paros. Keď nám lokálne beží proxy tak môžeme začať preskúmavať web aplikáciu ktorá beží na cielenom OS. Ako nám už naznačil nástroj Nikto tak súbor robots.txt môže obsahovať cesty ku priečinkom ktoré môžu byť pre nás zaujímave. Po preskúmaní tohto súboru a URL ktorý obsahoval, sme zistili že niektoré URL sú zabezpečené no URL /sql nie je. Na tejto URL sme našli databázový create script ktorý nám odhaľuje schému databázy ktorú webová aplikácia používa. Túto informáciu vieme zneužiť.

2.3 Získanie administrátorského prístupu

Zistili sme že v databáze sa nachádza tabuľka s názvom user a stĺpcami user_name a user_pass. Keďže vývojár sa nie úplne riadil konvenciami pri pomenovávaní tabuliek a stĺpcov môžme si povedať že to bol nejaký junior a možno nám nechal aj cestu ku administrátorskému prístupu jednoduchým útokom SQL injection. Útok SQL injection funguje na princípe pokusu vyhodnotenia SQL dotazu na hodnotu ktorá nám umožní obísť zabezpečenie bez toho aby sme poznali prihlasovacie údaje do aplikácie.
Ak je na zabezpečenie na serveri použitý dotaz v tvare

```
$sql = \select * from users where username='$username' and
pass='$password'";
$results = mysql_query($sql);
```

Môžeme na miesto premenných $username a $password prepašovať kúsok SQL dotazu ktorý spôsobí že dotaz bude vyhodnotený s truthy výsledkom a získame administrátorský prístup.
Nástroj Nikto nám dal nápovedu že na URL /admin môže byť formulár na prihlásenie do systému ako administrátor. V tomto prípade môžeme do oboch text inputov na URL /admin zadať kúsok SQL dotazu ktorý vyhodnotí dotaz na truthy hodnotu. Do prihlasovacieho formulára sme zadali ' or 1='1 no stále sme sa nedokázali do aplikácie prihlásiť. Môžno programátor systému nebol až taký junior. Skúmame ďalej. Pozrieme sa do našej proxy. Vidíme že HTTP POST request na server odoslal údaje username=or11&password=or11. No to nie je to čo sme mi napísali. Niekde

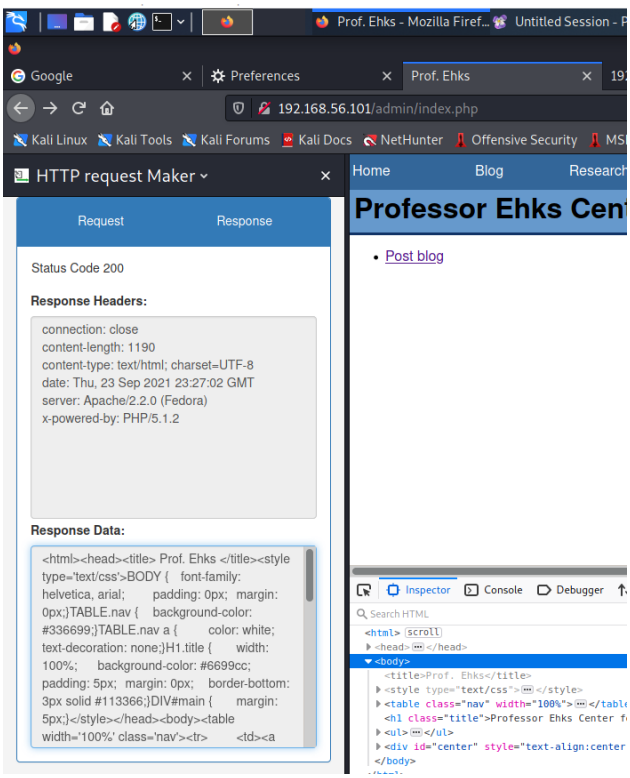
bol tento text zmenený, niekde ešte v browseri. Je zrejmé že na stránke je nejaký kus JS kódu ktorý mení hodnoty ktoré zadávame do formulára aby escapol znaky ktorý by nám dovolili takto jednoducho realizovať SQL injection.
Môzeme skúsiť odoslať tento POST request pomocou našej proxy kde nebude žiadny JS meniť náš vstup. Aj keď sme odoslali request bez toho aby nám JS zmenil vstup tak sa nám nepodarilo prihlásiť.
Predpokladáme že validácia nebude iba na strane klienta ale aj a strane servera.
Skúsime odoslať taký vstup ktorý spôsobí že SQL dotaz na serveri bude syntakticky neplatný, spadne a dúfame že programátor zabudol odkloniť chybové hlášky do logovacieho systému a zobrazí nám ich priamo na stránke. A naozaj, programátor zabudol na dôležitú vec a tím nám dal ešte viac informácií o systéme. Pomocou proxy sme poslali hodnotu "test" ako prihlasovanie meno a neposlali sme žiadne heslo. Toto spôsobilo chybu pri vyhodnotení SQL dotazu a dostali sme chybovú hlášku:

```
Problem with query:<br/><b>select user_id from user
        where user_name='' test'
                    AND user_pass = md5('')</b><br/>You have an error in your SQL
                    ↪ syntax; check the manual that corresponds to your MySQL
                    ↪ server version for the right syntax to use near 'test'
                       AND user_pass = md5('')' at line 2
```

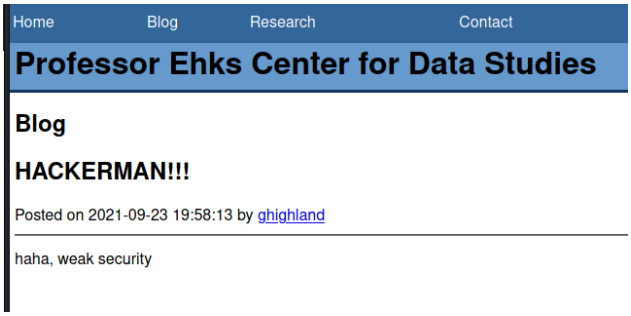
Teraz vidíme že SQL dotaz vypadá následovne:

```
select user_id from user where user_name=${username} AND user_pass = md5(${password})
```

Poďme to zneužiť.
Ako meno môžeme skúsiť: ' or 1='1 a heslo: ') or 1=1 #
Aby sme získali administrátorský prístup do systému v prehliadači tak si musíme nainštalovať vhodný plugin ktorý odchytáva a upravuje odchádzajúce requesty. Keď tak spravíme tak môžeme odoslať spomínané údaje. Vidíme že podľa poznatkov sa nám podarilo napísať SQL injection ktorá nám umožnila získať administrátorský prístup do systému.



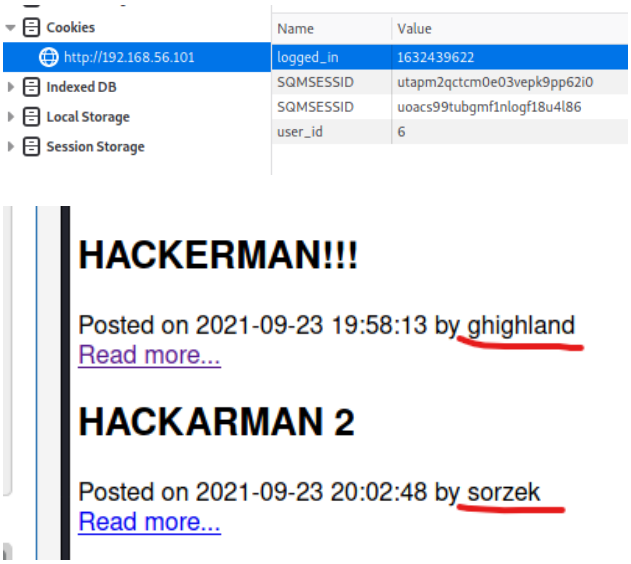
Teraz dokážeme v mene prihláseného používateľa vytvoriť blog post.



Za zmienku stoja aj súbory cookies ktoré sú uložené v pamäti prehliadača. Po prihlásení sa vytvorili dvoje cookies logged_in a user_id. Nás bude zaujímať skôr cookie user_id. Vidíme že hodnota key-value páru je čislo.

Skúsme ho zmeniť, zverejniť jeden článok a porovnať čo sa zmenilo.

Vidíme že sa zmenil autor. To znamená že toto číslo bude asi primárny kľúč v tabuľke user a keď zmeníme hodnotu tohto cookie tak vieme zverejňovať články v mene niekoho iného. Na základe tohto by sme mohli zistiť všetky mená registrovaných používateľov systému.

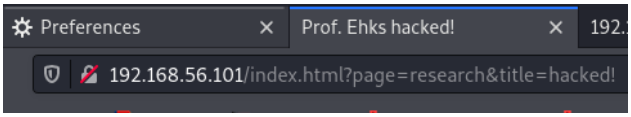


2.4 XSS - cross site scripting

Nástroj Nikto nás upozornil že webová aplikácia môže byť nachylná na XSS útok. Keď sme vytvárali blog posty v mene niekoho iného tak sme si mohli všimnúť tvar url:

```
http://192.168.56.101/index.html?page=blog&title=Blog
```

Keď preklikávame stránku tak vidíme že query parametre page a title v URL sa mení podľa toho akú podstránku navštívime. Skúsme ich upraviť a uvidíme čo sa stane. Keď upravíme parameter page na náhodnú hodnotu tak sa nám zobrazí iba prázdna stránka (tento parameter je použitý na navigáciu v aplikácií) čo nie je pre nás zaujímavé. Skúsme zmeniť hodnotu parametra title.

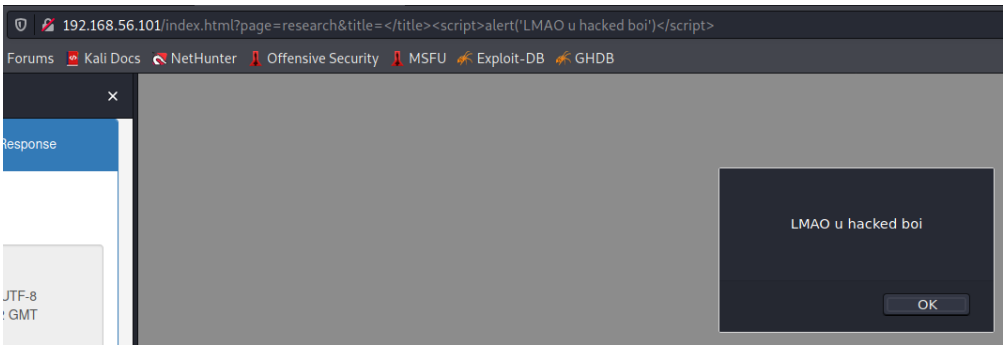


Vidíme že dokážeme zmeniť časť názvu stránky. Skúsme tam poslať nejaký JS.

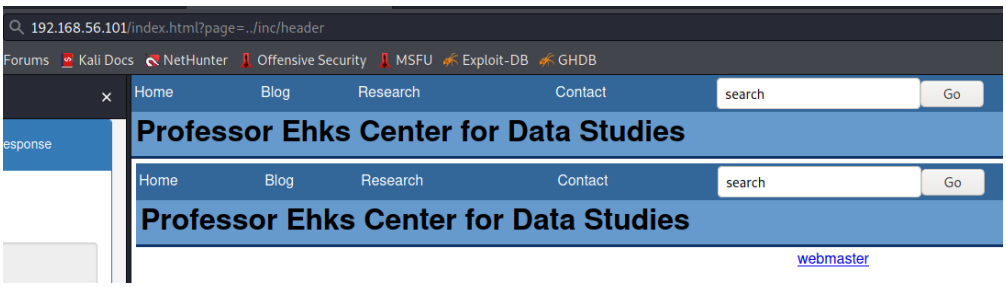
Dokázali sme spustiť kus JS kódu. Týmto spôsobom môžeme spustiť akýkoľvek JS kód ktorý dokáže presmerovať obeť útoku na akúkoľvek inú stránku ktorá môže byť potencionálne škodlivá. Jediné čo musíme spraviť je začať hodnou query parametra title je uzavrieť HTML tag ¡/title¿ a napísať akýkoľvek JS za to.

2.5 File include zraniteľnosť

V predchádzajúcej kapitole sme spomenuli že query parameter page je požitý pre navigáciu v aplikácií. Ak nie je táto navigácia ošetrená proti útokom tak môžeme realizovať file include útok pomocou ktorého môžeme zobraziť časti stránky ktoré by sa nemali zobrazovať používateľovi. Štandardným spôsobom pri vývoji PHP stránok je importovať do stránky už existujú kód ktorý vyrenderuje často sa opakujúce časti stránky, napríklad hlavička alebo päta. Pomocou brute force



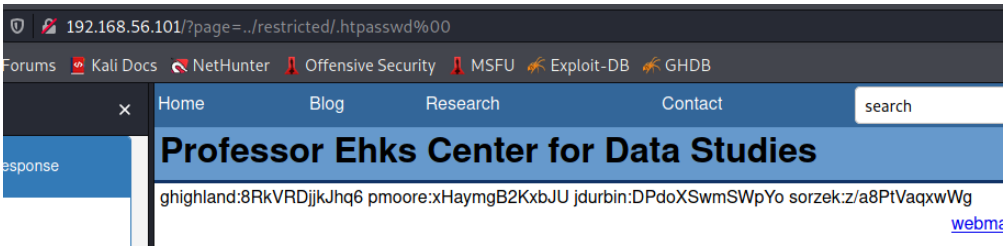
útoku na filesystem nástrojom DirBUster sme zistili že na webovom server existuje priečinok /inc ktorý obsahuje súbory ktoré majú rovnaké názvy ako všetky nám známe platné hodnoty query parametra page. Skúsme zmeniť hodnotu query parametra page na page=../inc/header. Vidíme že sa nám podarilo na stránke vyrenderovať 2x hlavičku stránky.



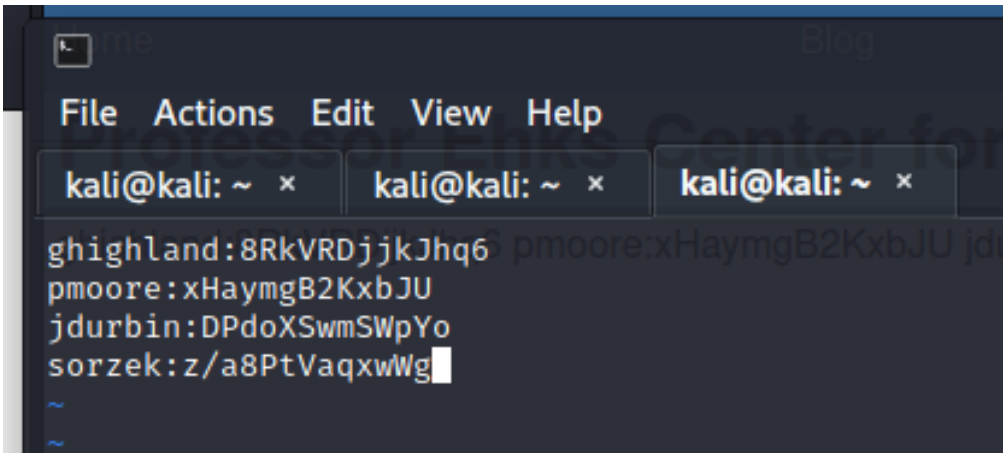
Tento poznatok vieme zneužiť neskôr.

2.6 Odhalenie uložených hesiel

Pri PHP aplikáciách bývajú uložené zahaskované heslá v súbore /resticted/.httpasword. Skúsime túto URL zdať do prehliadača. Ku tejto URL nemáme prístup no keď požijeme file inclusion útok tak sa nám to možno podarí.
Keď na koniec URL pridáme /?page=../restricted/.htpasswd%00 (%00 je teminujúci znak aby sme obišli často používane zabezpečenie v PHP aplikáciách) dostaneme obsah súboru kde sú zahashované heslá.



Tieto heslá si skopírujeme od textového súboru a oddelíme novým riadkom.
Pomocou nástroja Jack the Ripper dokážeme prelomiť heslá. Spustíme príkaz




```
john -wordlist=/usr/share/dirbuster/wordlists/directory-list-2.3-small.txt htpasswd
```

Keď program skončí tak nám poskytne heslo pre účet sorzek.

```
Using default input encoding: UTF-8
Loaded 1 password hash (descrypt, traditional crypt(3) [DES 256/256 AVX2])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
pacman           (sorzek)
1g 0:00:00:00 DONE (2021-09-27 04:53) 100.0g/s 1638Kp/s 1638Kc/s 1638KC/s
↪  translat..20060915
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Môžeme sa skúsiť prihlásiť do OS cez účet sorzek.

```
$ ssh -o KexAlgorithms=diffie-hellman-group1-sha1 sorzek@192.168.56.101
BSD SSH 4.1
sorzek@192.168.56.101's password:
Last login: Thu Dec  1 15:25:25 2016 from 192.168.56.1
[sorzek@ctf4 ~]$ whoami
sorzek
[sorzek@ctf4 ~]$
```

Vidíme že sa nám podarilo prihlásiť pod účtom sorzek do cieleného OS keďže používal rovnaké heslo na prihlásenie do PC ako na prihlásenie do webovej aplikácie.

2.7 Krádež súkromného SSH kľúča

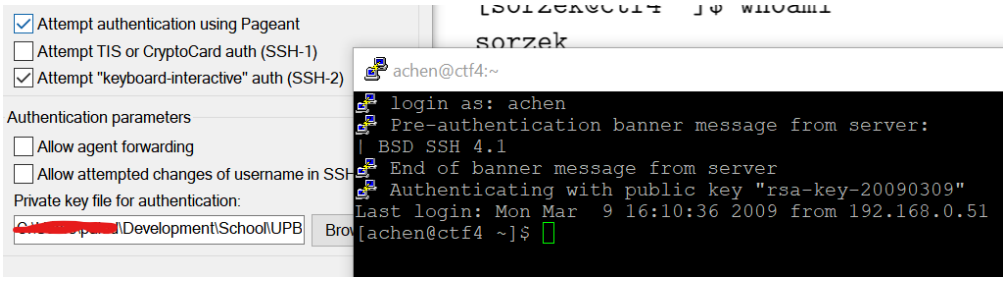
Keď zostaneme prihlásený pod účtom sorzek z predchádzajúceho kroku môžeme zistiť aký používatelia sú v skupine admin čo nám povie ktorý účet na OS má administrátorské privilégia. Tieto veci zistíme keď si dáme vypísať súbor /etc/group kde môžeme vidieť že v skupine admin sú používatelia

```
admins:x:507:dstevens,achen
```

Môžeme skúsiť preskúmať home adresár používateľa achen ktorý by nám ale nemal byť prístupný no zistili sme že admin zabudol toto obmedzenie pridať do systému takže môžeme prehliadať všetky súbory ktoré patria adminovi. Keďže súbory admionov nie sú nijak zabezpečené tak môžeme ukradnúť SSH kľúč ktorý nám dovolí prihlásiť sa do OS cez SSH bez toho aby sme museli poznať heslo používateľa pod ktorým sa prihlasujem.

SSH kľúče sú v priečinku /home/achen/.ssh. Obsah súboru /home/achen/.ssh/achen_priv.ppk skopírujeme do separátneho súboru.

Keďže kľúč je vo formáte pre aplikáciu Putty musíme si ho pomocou puttygen prekonvertovať do openSSH formátu alebo do programu putty rovno dame náš ukradnutý .ppk private key, zadáme IP adresu a sme pripojený na OS.



Keď už sme pripojený ako admin systému tak sa môžeme rozhliadať neobmedzene po OS a pozrieť si napríklad históriu príkazov ktorú používateľ spúšťal. Môžeme vidieť že jeden príkaz ktorý bol spustený adminom je zvláštny - nie je to príkaz ale root heslo omylom napísané do shellu.