Pedagogická fakulta
Faculty of Education

Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

Jihočeská univerzita v ČB

Pedagogická fakulta

Katedra informatiky

Hesla zadávána hackery při pokusu o průnik do systému

Passwords used by hackers when trying to break into computer system

Bakalářská práce

Vypracoval: Martin Svatoš

Vedoucí práce: Mgr. Václav Šimandl, Ph.D.

České Budějovice 2022

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH

Pedagogická fakulta

Akademický rok: 2020/2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Martin SVATOŠ**
Osobní číslo:	**P190051**
Studijní program:	**B7507 Specializace v pedagogice**
Studijní obor:	**Informační technologie a e-learning**
Téma práce:	**Hesla zadávána hackery při pokusu o průnik do systému**
Zadávající katedra:	**Katedra informatiky**

Zásady pro vypracování

Cílem práce je analyzovat hesla zadávaná hackery do přihlašovacích formulářů webových služeb. Student upraví přihlašovací moduly (pro přihlášení běžného uživatele do služby i přihlášení správce do administrace) ve webové aplikaci Bobříka informatiky. Po této úpravě bude modul schopen odhalit ty pokusy o přihlášení se do systému, které jsou prováděny pomocí bootů (jde obvykle o řadu neúspěšných pokusů o přihlášení z jedné IP adresy). Tyto pokusy o přihlášení student zanalyzuje a zjistí, jakou kombinaci uživatelského jména a hesla booti (a potažmo hackeři, kteří je ovládají) používají – zda jde o sociální inženýrství, slovníkové útoky či útoky hrubou silou apod. V případě, že by množství útoků na server v monitorovaném období bylo pro analýzu nedostatečné, student bude analyzovat data poskytnutá z jiných serverů. V teoretické části práce se student zaměří na vysvětlení principů a typů útoků, dále popíše software používaný k těmto útokům. Zaměří se také na hesla – jejich bezpečnost, šifrování při přenosu, možnosti uložení na zařízení klienta i na cílovém serveru.

Rozsah pracovní zprávy:	**40**
Rozsah grafických prací:	**CD ROM**
Forma zpracování bakalářské práce:	**tištěná**

Seznam doporučené literatury:

1. KOLOUCH, Jan a Pavel BAŠTA. CyberSecurity. Praha: CZ.NIC, 2019. ISBN 978-80-88168-31-7.
2. ERICKSON, Jon. Hacking: the art of exploitation. 2nd ed. San Francisco, CA: No Starch Press, 2008. ISBN 1593271441.
3. VRÁNA, Jakub. 1001 tipů a triků pro PHP. Brno: Computer Press, 2010. ISBN 978-80-251-2940-1.
4. RAHMEL, Dan. Joomla: podrobný průvodce tvorbou a správou webů. Brno: Computer Press, 2010. ISBN 978-80-251-2714-8.
5. WELLING, Luke a Laura THOMSON. Mistrovství PHP a MySQL. Přeložil Ondřej BAŠE. Brno: Computer Press, 2017. ISBN 978-80-251-4892-1.
6. SNYDER, Chris, Tom MYER a Michael G. SOUTHWELL. Pro PHP security: from application security principles to the implementation of XSS defenses. 2nd ed. New York: Apress Media, 2010. ISBN 1430233184.
7. STUTTARD, Dafydd a Marcus PINTO. The web application hacker's handbook: finding and exploiting security flaws. 2nd ed. Chichester: Wiley, 2011. ISBN 978-1118026472.
8. HALL, Gery a Erin WATSON. Hacking: Computer Hacking, Security Testing, Penetration Testing and Basic Security. CreateSpace, 2016. ISBN 978-1541289321.

Vedoucí bakalářské práce: **Mgr. Václav Šimandl, Ph.D.**
Katedra informatiky

Datum zadání bakalářské práce: **1. dubna 2021**
Termín odevzdání bakalářské práce: **30. dubna 2022**

L.S.

doc. RNDr. Helena Koldová, Ph.D.
děkanka

doc. PaedDr. Jiří Vaníček, Ph.D.
vedoucí katedry

V Českých Budějovicích dne 1. dubna 2021

Prohlášení

Prohlašuji, že bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích dne 27. června 2022.

Martin Svatoš

.....

Abstrakt

Cílem bakalářské práce je zpracovat problematiku hackerů, kteří se snaží o průnik do systému Joomla v Bobříka informatiky. Byly vyvinuty tři různé moduly, první z nich slouží pro uživatelské prostředí, další pro administrátorské prostředí a poslední pro čištění databáze. Dále vznikla tabulka, která slouží pro záznam všech nepodařených pokusů hackerů, kde jsou uchovávaná jejich hesla a jména.

Zjištěná data poukazují na hackery využívající software, jenž mají v seznamu určitá slova (převážně v anglickém jazyce) a ty následně zadávají do přihlašovacích formulářů Bobříka informatiky. Typicky nejpoužívanější kombinací hesla a jména, je ”admin” a na druhém místě je jméno ”admin” s heslem ”123456”.

Klíčová slova

PHP, Joomla, SQL, hacker, šifrování, dešifrování, hesla, útoky hrubou silou, slovníkový útok

Abstract

The aim of the bachelor's thesis is to address the issue of hackers who are trying to break into the Joomla system in Bobřík informatics. Three different modules have been developed, the first for the user environment, the second for the administration environment and the last for database cleaning. Furthermore, a table was created, which is used to record all failed hacker attempts, where their passwords and names are stored.

The obtained data point to hackers using software, which they have in the list of certain words (mostly in English) and then enter them into the login forms of Bobřík Informatics. Typically, the most commonly used combination of password and name is ”admin”, followed by the name ”admin” with the password ”123456”.

Keywords

PHP, Joomla, SQL, hacker, encrypt, decrypt, passwords, brute-force attacks, vocabulary attack

Poděkování

Děkuji Mgr. Václavu Šimandlovi, Ph.D. za vedení mé bakalářské práce, dále děkuji za jeho nápovědy, rady, trpělivost a vstřícnost.

Obsah

1 Úvod

1.1 Cíl práce

Cílem bakalářské práce je analyzování hesel zadávaná hackery do přihlašovacích formulářů webových služeb, tedy hesla zadávaná uživatelem při pokusu o průnik do serveru, v tomhle případě se jednalo o Bobříka informatiky, kde jsem upravil přihlašovací modul (nejen pro přihlášení běžného uživatele, ale i administrátora). Po této úpravě modul je schopen odhalit ty pokusy o přihlášení se do systému. Dalším dílčím cílem je nutno určit o jaký typ útoků se jedná, třebaže jde například o obyčejné slovníkové útoky, sociální inženýrství, nebo útoky hrubou silou, do kterých se právě řadí boti (počítačový program, který vykonává určité instrukce uživatelem), apod. Další dílčí prací jsem zanalyzoval testovací server, kde byl využit rozsáhlý slovníkový útok.

V teoretické části jsem se zaměřil na vysvětlení principů a typy útoků, dále popsal software využívaný k útokům. V další dílčí části jsem ukázal, jak takový útok vzniká. Byla diskutovaná hesla, která jsou nejméně bezpečná a také nejvíce používaná, dále jak dlouhé a složité bezpečné heslo má vypadat. V poslední části teorie jsem se zaměřil na šifrování hesel při přenosu dat a na závěr uložení hesla na zařízení klienta i na cílovém serveru.

Hlavním přínosem mé bakalářské práce je, že jsme získali kompletní přehled zadávaných hesel a jmen, popřípadě jaké typy útoků se používají a jak dlouho můžou trvat. V budoucnu z toho může vzniknout ochrana. Další dílčí částí bylo rozebírání několik jednotlivých pokusů a zda takové přihlašovací údaje používat a samozřejmě máme upravené přihlašovací moduly pro Bobříka informatiky.

1.2 Metoda práce

Informace k teoretické části jsem čerpal z internetových zdrojů zaměřující se na kybernetické útoky servery.

K popisování potencionálních nebezpečných programů byly využity odborné literatury zabývající se stejnou problematikou.

K praktické části jsem vylepšil přihlašovací moduly - jak pro obyčejného uživatele tak i administrátora. Využil jsem k tomu internetové zdroje z referencí.

Co se týče samotného kódu, je to mnohem složitější, ať už z důvodu vývojového diagramu, nebo komplexnosti samotné struktury modulu. To znamená, že jsem musel vytvořit návrh funkcionality modulů včetně vhodných diagramů. Nejednalo se pouze o vylepšení "pár" příkazových řádků, ale o předělání jednotlivých modulu pro přihlášení, to zahrnuje nejen administrátora, uživatele, ale i vytvoření úplně nového modulu, který právě bude spolupracovat s moduly výše uvedené.

Nasbíraná data, která se ukládají do databáze v Bobříka informatiky, jsem zanalyzoval. Zaměřil jsem se na IP adresy, zda je poznat využití VPN, či nikoliv. Dále jména a hesla, tedy pokud se v jednotlivých pokusech opakují. Ohled jsem bral i na záměr používaných hesel, poté spojitost, tedy zda se jedná o slovníkový, nebo jiný útok. Na závěr rozsah a velikost útoků a hlavně kdy začal a skončil.

Řešení teoretické části, na rozdíl od praktické je o něco jednodušší, protože způsobem, jakým jsem to vypracoval je, že veškeré informace, které jsem si našel ohledně zadaného tématu, bylo poznamenáno a následně ověřeno, tedy zda se data shodují ve více zdrojích.

2 Redakční systém Joomla!

2.1 Model-View-Controller

Když Joomla začne zpracovávat požadavek od uživatele, jako je GET pro konkrétní stránku nebo POST obsahující data formuláře, jedna z prvních věcí, kterou Joomla dělá, je analyzování URL, aby určila, která komponenta bude zodpovědná za zpracování požadavku a předání řízení této složce [1].

Udělá to spuštěním souboru PHP komponenty vstupního bodu pro tuto komponentu. Pokud se tedy komponenta nazývá com_example, pak se Joomla spustí [1]:

```
components/com_example/example.php //front-end
administrator/components/com_example/example.php //back-end admin
```

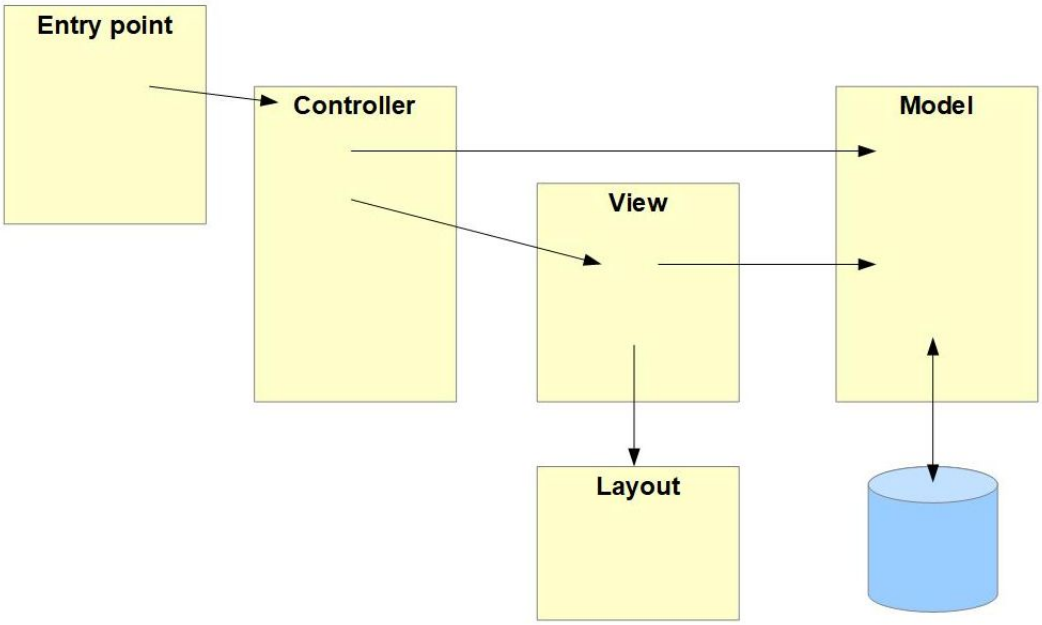
Pokud je vyvíjena komponenta, může se ve skutečnosti vložit celý kód komponenty do těchto 2 souborů example.php. Výhodou následování vzoru Joomla MVC je však to, že se může plně využít třídy Model-View-Controller knihovny Joomla, což výrazně snižuje množství kódu, který se potřebuje napsat [1], [2].

2.1.1 Entry Point code

Hlavní úlohou vstupního souboru PHP (example.php pro com_example) je určit, který driver spustit. Dělá to na základě parametru úlohy a zahrnuje určení, jaká třída controlleru by měla být načtena, kde najít kód pro danou třídu, získání instance této třídy a volání příslušné metody této třídy [1].

2.1.2 Controller

Kontrolér je odpovědný za analýzu požadavku uživatele, kontrolu, zda je uživateli povoleno provést tuto akci a za určení, jak požadavek uspokojit. Ten druhý bude zahrnovat [1]:



Obrázek 1: Model-View-Controller [1]

- určení, který model (nebo modely) bude potřeba ke splnění požadavku a vytvoření instance tohoto modelu
- volání metod modelu k provedení všech požadovaných aktualizací databáze
- určení, které zobrazení by se mělo použít k prezentaci webové stránky uživateli a vytvoření instance tohoto zobrazení, nebo
- pokud by měl uživatel místo toho obdržet přesměrování na jinou adresu URL, pak určení této adresy URL přesměrování [1].

2.1.3 Model

Model zapouzdřuje data používaná komponentou. Ve většině případů budou tato data pocházet z databáze, buď databáze Joomla, nebo nějaké externí databáze, ale je také možné, aby model získával data z jiných zdrojů, například

přes API[1] webových služeb běžící na jiném serveru. Model je také zodpovědný za aktualizaci databáze tam, kde je to vhodné. Účelem modelu je izolovat správce a zobrazit podrobnosti o tom, jak jsou data získávána nebo upravována [1].

Pokud komponenta zobrazuje formulář, který je definován v XML pomocí přístupu Joomla Form, pak model zpracovává nastavení a konfiguraci instance formuláře, připravený na rozložení do výstupních polí pomocí renderField() [1].

2.1.4 View

Zobrazení určuje, co se má na webové stránce objevit a shromažďuje všechna data nezbytná pro výstup odpovědi HTTP [1].

Poté, co driver vytvoří instanci view, zavolá metodu setModel() view a předá instanci modelu. Tímto způsobem view ví, který model má použít, a volá metody modelu, aby získal data potřebná pro návrat k uživateli [1].

2.1.5 Layout

Zobrazení nevydává HTML, ale deleguje jej na rozvržení. Rozvržení obsahuje kód PHP, který běží v kontextu metody (obvykle display()) view, což znamená, že pokud view obsahuje data odezvy, například this ->items, pak rozložení může přistupovat ke stejnému this ->items při výstupu HTML [1].

Oddělení zobrazení a rozvržení tímto způsobem umožňuje další úroveň flexibility, protože se může snadno nastavit přepsání rozvržení pro výstup dat zobrazení pomocí vlastního preferovaného HTML [1].

[1] Application Programming Interface - označuje v informatice rozhraní pro programování aplikací [3]

2.2 Tvorba modulu v CMS

2.2.1 Bezpečnost

Veškerý vstup pocházející od uživatele musí být považován za potenciálně nebezpečný a musí být před použitím vyčištěn. K načtení dat z požadavku by se mělo vždy používat třídu Joomla JInput, spíše než nezpracované proměnné $GET, $POST nebo $REQUEST, protože metody JInput standardně používají vstupní filtrování. JInput se zabývá všemi aspekty uživatelského požadavku způsobem, který je nezávislý na použité metodě požadavku. Lze jej také použít k načtení dat souborů cookie a dokonce i proměnných serveru a prostředí. Pro zajištění maximální bezpečnosti je však důležité používat správnou metodu JInput. Je velmi snadné použít JInput->get metodu s výchozími parametry a ignorování skutečnosti, že v mnoha případech je možné použít přísnější požadavek na uživatelský vstup [4].

Je velmi důležité porozumět tomu, že metody JInput neznají SQL a je zapotřebí další práce na ochranu před útoky SQL injection. Neexistuje žádná výchozí hodnota, která bude vrácena, pokud není zadána výchozí hodnota ve volání JInput->get. Pokud není zadáno žádné výchozí nastavení a argument není přítomen v proměnné požadavku, vrátí se nedefinovaný. [4]

Při zvažování uživatelského vstupu by se mělo přemýšlet o datovém typu, který očekává načtení, a použít nejpřísnější formu JInput, která je použitelná v každém případě. Zejména se vyhnout línému přístupu pomocí JInput->get, protože to vrátí pole, které může obsahovat položky, které nejsou očekávané, a přestože každá z těchto položek byla vyčištěna, často se stává, že mohlo dojít k dodatečnému filtrování a aplikovat na některé jednotlivé argumenty. Například metoda get zachází se všemi argumenty jako s řetězci, zatímco může být možné omezit některé argumenty na celá čísla [4].

První tři parametry každé z metod get JInput jsou stejné. Povinný je pouze první parametr a je třeba si všimnout, že žádost zahrnuje data cookie. Obecně platí, že formát je:

```
JFactory::getApplication->
input-><data-source>->get<type>(<name>,<default>)
```

kde platí [4],

Proměnná	**Příkaz**
<type>	datový typ, který se má načíst
<name>	název proměnné, která má být načtena
<default>	výchozí hodnota
<data-source>	určuje, odkud má být proměnná načtena,

Tabulka 1: Popis syntaxe JFactory [4]

2.2.2 Tabulky a dotazy

Dotazování databáze Joomly se změnilo od zavedení nového rámce Joomla "řetězení dotazů" je nyní doporučenou metodou pro vytváření databázových dotazů (ačkoli řetězcové dotazy jsou stále podporovány) [5].

Řetězení dotazů označuje metodu propojování několika metod, jedna po druhé, přičemž každá metoda vrací objekt, který může podporovat další metodu, zlepšuje čitelnost a zjednodušuje kód [5].

K získání nové instance třídy JDatabaseQuery voláme metodu getQuery JDatabaseDriver:

```
$db = JFactory::getDbo()
$dotaz = $db->getQuery(true);
```

JDatabaseDriver ::getQuery přebírá volitelný argument "new", který může být true, nebo false (výchozí hodnota je false) [5].

K dotazu na náš zdroj dat můžeme zavolat řadu metod JDatabaseQuery; tyto metody zapouzdřují dotazovací jazyk zdroje dat (ve většině případů SQL), skrývají před vývojářem syntaxi specifickou pro dotaz a zvyšují přenositelnost zdrojového kódu vývojáře [5].

Některé z častěji používaných metod zahrnují: include; select, from, join, where a order. Pro úpravu záznamů v datovém úložišti existují také metody jako vkládání, aktualizace a mazání. Zřetězením těchto a dalších volání metod se může vytvořit téměř jakýkoli dotaz na jakékoliv úložiště dat, aniž by byla ohrožena přenositelnost kódu [5].

Třída JDatabaseQuery poskytuje řadu metod pro vytváření dotazů na vložení, z nichž nejběžnější jsou insert, columns, values [5]. Například:

```
1  $db = JFactory::getDbo();
2  $dotaz = $db->getQuery(true);
3  // Pole k aktualizaci.
4  $fields = array(
5      $db->quoteName('profile_value'). ' = '
6      .$db->quote('Aktualizace vlastni zpravy pro
7      uzivatele 1001.'),
8      $db->quoteName('objednavka') . ' = 2',
9
10 Pokud chcete ulozit hodnotu NULL, meli byste to zadat.
11     $db->quoteName('avatar') . ' = NULL',
12 );
13 // Podminky, za kterych by mely byt zaznamy aktualizovany.
14 $conditions = array(
15     $db->quoteName('user_id') . ' = 42',
16     $db->quoteName('profile_key') . ' =
17     ' . $db->quote('custom.message')
18 );
19 $query->update($db-> quoteName('#__user_profiles'))->
20 set($fields)->where($conditions);
21 $db->setQuery($dotaz);
22 $vysledek = $db->execute();
```

Výpis kódu 1: Příklad dotazu pro aktualizaci dat [5]

2.3 Umístění kódů zadané problematiky

2.3.1 Administrator

Pro přihlášení do Joomla a přístup k panelu pro správu, se musí otevřít prohlížeč a přejít na adresu http://mydomain.com/administrator. Na této stránce se dá najít přihlašovací obrazovku, kde by se mělo zadat uživatelské jméno a heslo, které se zvolilo během Joomly, instalačního procesu [1].

Soubor pro editaci přihlášení v administrátorském prostředí, lze snadno dosáhnout nalezeným souboru, jenž je v

```
/.../administrator/component/com_login
```

kde se nachází soubor controller.php. Je nutné dbát zvýšené bezpečnosti a příkazy volat pomocí JFactory [2].

2.3.2 User

Většinou soubory, pro editaci jsou mimo složky administrator a to přímo v komponentách nainstalované Joomly

```
/.../components/com_users/controllers
```

soubor user.php. Většina uživatelských přihlašovací formulářů si autoři upraví sami, to platí pro zejména vzhled a přihlašovací data, kam se mají odesílat [1].

3 Kybernetické hrozby, události, incidenty a útoky

3.1 Obecná definice

Hrozbu můžeme nejjednodušeji definovat jako něco, co je schopno narušit běžný či řádný stav věcí a zasáhnout do práv jiných subjektů. Jde o negativní působení, které může, ale nemusí být dokončeno. Pro vlastní definici je dostačující, že možnost negativního stavu hrozí a je reálná [6].

3.2 Zdroje hrozby

Hrozby způsobené člověkem. V případě, že je hrozba způsobena člověkem, je vhodné se zaměřit i na formu zavinění, jež vedlo k iniciaci dané hrozby. Z tohoto pohledu je možné rozlišovat hrozby způsobené: [6]

- Úmyslně způsobené kybernetické hrozby je možné zařadit například: úmyslné smazání dat, konfigurace systému, fyzické poškození počítačového systému či jiného prvku ICT, zcizení dat a informací, kybernetické útoky (malware, DoS, DDoS, phishing, neoprávněný odposlech). [6] [7]
- Z nedbalosti řadíme například: omylem smazaná data, fyzické poškození počítačového systému či jiného prvku ICT (např. pádem, překopnutím strukturované kabeláže), poškození dat, systémů či jiných prvků na základě neseznámení se s interními akty (právními či technickými), jiná chyba uživatele [8].

3.3 Zdroje působení

Zdroje působení se dělí na dvě části:

- vnitřní, kde se hrozby nachází v organizaci, jako když cizí, nebo místní osoba vloží flash disk do serveru.
- vnější, jenž se nachází mimo organizaci, jako například útok mimo síť, třeba že útočník se nachází na druhé straně planety [9].

3.4 Cíle hrozby

Prvním cílem je útok na triádu CIA.

- Confidentiality (důvěrnost) – např. krádeže dat, přístupových údajů a klíčů, hardware.
- Integrity (celistvost) – Snaha narušit databázi.
- Availability (dostupnost) – např. DoS a DDoS útoky, fyzické útoky na servery a strukturovanou kabeláž, výpadky proudu [6].

Dalším cílem jsou útoky na některých z prvků kybernetické bezpečnosti.

- Lidé – útoky sociálním inženýrstvím (ve světě reálném, ale i kyberprostoru), phishing, malware, krádeže.
- Technologie – veškeré hrozby uvedené v bodě 1 této klasifikace. Typicky mohou hrozby působit na hardware, databáze, síť a síťovou infrastrukturu, software, informace a data uložená v počítačových systémech [6].

3.5 Motivace

Pokud je hrozba způsobena úmyslným jednáním člověka, je vhodné se při řešení hrozby zabývat i její motivací. Na základě analýzy motivace takovéhoto jednání je v rámci procesu reakce na hrozbu možné vytvořit nápravná opatření, aby nedocházelo ke stimulu této motivace i v budoucnu [10]. Dle motivace lze sledovat:

- hrozby za účelem získání finančního prospěchu,
- hrozby za účelem získání konkurenční převahy,
- hrozby za účelem dokázání svých schopností,
- hrozby za účelem odplaty,
- hrozby z důvodu neplnění povinností [10].

3.6 Typ hrozby

Hrozby můžeme zařadit do několika bodů, těmi jsou:

- slovníkový útok,
- sociální inženýrství,
- Brute-Force attack (útoky hrubou silou)
- DoS, DDoS, DRDoS útoky,
- SQL injection. [6], [7], [11].

Podrobněji je to vysvětlené v následující kapitolách.

3.6.1 Slovníkový útok

Slovníkový útok je metoda odhalování hesel, při které útočník postupně zkouší hesla ze slovníku. Slovník je seznam slov, u kterých je pravděpodobné, že je uživatel mohl zvolil jako své heslo. Nejzákladnějším slovníkem je seznam slov uživatelova mateřského jazyka, případně jejich kombinace. Algoritmy pro slovníkové útoky počítají s tím, že je potřeba vyzkoušet různé kombinace daného slova. Běžné je zkoušet různé kombinace malých a velkých písmen, přidávat do zkoumaného slova čísla a speciální znaky. Heslo „PaSSword1.“ není tedy z tohoto pohledu o nic bezpečnější než „password“. Jeden z velmi známých slovníků, Rockyou533, je postupně vytvářen s využitím hesel, která unikla z nejrůznějších systémů. Obsahuje tak veliké množství hesel reálně používaných uživateli [12].

Vedle již existujících slovníků jsou dostupné nástroje, které umožňují vytvořit slovník na míru dle informací, jež má útočník o své oběti. Nástrojem umožňujícím vytvořit specifický slovník dle obsahu webových stránek je například program cewl. Jiným nástrojem, který lze využít pro generování slovníku na míru, je program cupp. Tento program umí vygenerovat slovník na míru konkrétní osobě [6].

3.6.2 Sociální inženýrství

Sociální inženýrství je snaha podvodem vylákat od důvěřivých uživatelů jejich osobní informace, jako jsou hesla nebo bankovní údaje, případně získat přístup k jejich počítači, za účelem instalace škodlivých programů. Zloději a podvodníci používají techniky sociální inženýrství. Je totiž snadnější podvodem vylákat uživatelské heslo, než složitě obcházet zabezpečení počítače [13].

3.6.3 Útoky hrubou silou

Brute force útok neboli útok hrubou silou je druh kyberútoku, jehož cílem je nejčastěji prolomení hesla. Útočníci používají software (prolamovač hesel), který postupně zkouší různé kombinace znaků, dokud neuhádne skutečné heslo. Tímto způsobem se mohou útočníci dostat do internetových služeb, zamčených souborů nebo do jakéhokoli digitálního prostoru, který vyžaduje uživatelské jméno a heslo [11].

Variantou brute force útoku je slovníkový útok, který nezkouší náhodné kombinace znaků, ale pracuje s databází potenciálních hesel, například zkouší nejčastěji používaná hesla. [11]

3.6.4 DoS, DDoS, DRDoS útoky

DoS neboli Denial of Service – odmítnutí služby, je způsob útoku, který způsobí, že prostředky počítače nebudou dostupné pro původní uživatele. Komunikace mezi uživateli je přetížená, neprobíhá správně a pro obnovení funkčnosti je nutné počítač restartovat [14].

Cílem se stávají nejčastěji webové servery, kdy účelem útoku je vyřadit je z provozu [14].

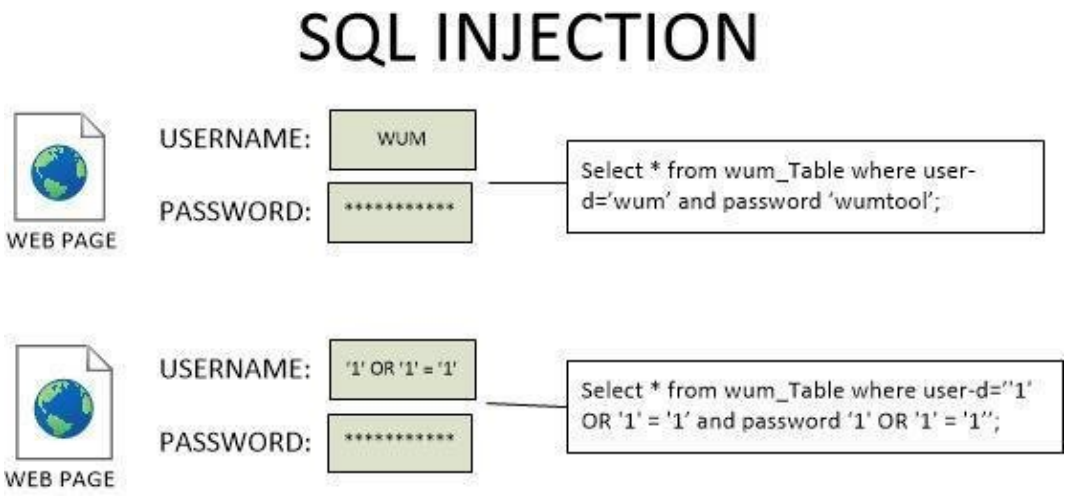
Kybernetický útok typu DDoS je, kybernetický útok typu DoS, který probíhá najednou koordinovaně z mnoha uzlů sítě. [15]

3.6.5 SQL Injection a Cross-site scripting (XSS)

SQL injection je typ útoku, který napadá databázovou vrstvu vsunutím (odtud slovo injection) kódu přes neošetřený vstup. S pomocí takto vsunutého kódu může útočník získat citlivé osobní informace, jako například číslo kreditní karty, nebo přihlašovací údaje. Také může databázi poškodit smazáním dat, dokonce může databázi upravit ve svůj prospěch [16].

Útoky Cross-Site Scripting (XSS) jsou typem injekce, při které jsou škodlivé skripty vpravovány do jinak neškodných a důvěryhodných webových stránek. K útokům XSS dochází, když útočník použije webovou aplikaci k odeslání škodlivého kódu, obvykle ve formě skriptu na straně prohlížeče, jinému koncovému uživateli. Chyby, které umožňují těmto útokům uspět, jsou poměrně rozšířené a vyskytují se všude, kde webová aplikace používá vstup od uživatele v rámci výstupu, který generuje, aniž by jej ověřovala nebo kódovala [17].

Příklad obrázku SQL Injection:



Obrázek 2: SQL Injection [18]

4 Software využíván hackery

Hackerské nástroje můžeme rozdělit do tří kategorií:

- Hardwarové nástroje - sem patří hledání a následné využití bezpečnostních děr v hardwaru. Jedná se minoritní oblast nástrojů pro hackování, pro příklad první techniky phreakerů by se daly označit za zástupce této kategorie [19] [20].
- Sociální inženýrství - jedná se o techniky zneužití lidského článku. Přestože je využito technologie, hlavním cílem je donutit klíčového člověka udělat chybu, kterou hacker následně využije. Příkladem může být phishing [19] [20].
- Softwarové (programové) nástroje - jde o převažující složku technik, které bývají pro hackerské aktivity používány. Základem je existence softwaru, který je vytvořený a uspůsobený pro určitý hackerův cíl. Spadají sem i techniky hledání bezpečnostních děr v běžných programech a softwarových systémech [19] [20].

4.1 Druhy a použití nebezpečného software hackery

4.1.1 CrackStation

CrackStation používá masivní předpočítané vyhledávací tabulky k prolomení hash hesel. Tyto tabulky ukládají mapování mezi hodnotou hash hesla a správným heslem pro tento hash. Hodnoty hashů jsou indexovány, aby bylo možné v databázi rychle vyhledat daný hash. Pokud je hash v databázi přítomen, lze heslo obnovit ve zlomku sekundy. Toto funguje pouze pro ”unsalted” hashe [21].

Vyhledávací tabulky Crackstation byly vytvořeny extrakcí každého slova z databází Wikipedie a přidáním každého seznamu hesel, který byl nalezen. Na

seznamy slov bylo použito inteligentní mandlování slov (hybrid hrubé síly), aby byly mnohem efektivnější [21].

4.1.2 Cain and Abel

Cain and Abel je nástroj pro obnovu hesla pro operační systémy Microsoft. Umožňuje snadnou obnovu různých druhů hesel prohledáváním sítě, prolomením zašifrovaných hesel pomocí útoků Dictionary, Brute-Force a Cryptanalysis, nahráváním konverzací VoIP, dekódováním zašifrovaných hesel, obnovováním klíčů bezdrátové sítě, odhalováním schránek s hesly, odhalováním hesel uložených v mezipaměti a analýzou směrování [22].

4.1.3 John The Ripper

John the Ripper je nástroj pro audit zabezpečení hesel s otevřeným zdrojovým kódem a nástroj pro obnovu hesla dostupný pro mnoho operačních systémů. John the Ripper jumbo podporuje stovky typů hash a šifer, včetně pro: uživatelská hesla variant Unix (Linux, *BSD, Solaris, AIX, QNX atd.), macOS, Windows, „webové aplikace“ (např. WordPress), groupware (např. Notes/Domino) a databázové servery (SQL, LDAP atd.); zachycení síťového provozu (ověření sítě Windows, WiFi WPA-PSK atd.); šifrované soukromé klíče (SSH, GnuPG, kryptoměnové peněženky atd.), souborové systémy a disky (soubory .dmg macOS a „sparse bundles“, Windows BitLocker atd.), archivy (ZIP, RAR, 7z) a soubory dokumentů (PDF , Microsoft Office atd.). Toto jsou jen některé příklady – je jich mnohem více [23].

4.1.4 THC - Hydra

Hydra je paralelní přihlašovací cracker [2], který podporuje četné protokoly k útoku. Je velmi rychlý a flexibilní a nové moduly lze snadno přidávat [24].

[2]Využívány více výpočetní jednotky souběžně kvůli zvýšení efektivity a rychlosti

Tento nástroj umožňuje výzkumníkům a bezpečnostním konzultantům ukázat, jak snadné by bylo získat neoprávněný přístup k systému na dálku [24].

Podporuje: Cisco AAA, Cisco auth, Cisco enable, CVS, FTP, HTTP(S)-FORM-GET, HTTP(S)-FORM-POST, HTTP(S)-GET, HTTP(S)-HEAD, HTTP-Proxy, ICQ, IMAP, IRC, LDAP, MS-SQL, MySQL, NNTP, Oracle Listener, Oracle SID, PC-Anywhere, PC-NFS, POP3, PostgreSQL, RDP, Rexec, Rlogin, Rsh, SIP, SMB(NT) , SMTP, SMTP Enum, SNMP v1+v2+v3, SOCKS5, SSH (v1 a v2), SSHKEY, Subversion, Teamspeak (TS2), Telnet, VMware-Auth, VNC a XMPP [24].

5 Dopady bezpečnostních incidentů

5.1 Únik citlivých a osobních informací

Útočníci mohou získat veškeré údaje z databáze, kde jsou evidovány všechny uživatelé/zákazníky. Mohou však získat i přístup k citlivým dokumentům, které nejsou dostupné ve veřejné části webu. Z kompromitovaných webů často získají i přístupové údaje a tokeny k nejrůznějším dalším službám, které na webu jsou využívány. Pokud není používáno silné hashování hesel a je zanedbána jejich síla, mohou útočníci získat hesla všech uživatelů a pokusit se je použít v jiných službách. Díky tomuto se i hackeři mohou také pokusit o cílený phishing na další zaměstnance [25].

5.2 Poškození důvěryhodnosti reputace

Pozitivní, ale i negativní recenze mohou přicházet ze stran zákazníků, velmi to ovlivní popularitu webové stránky. Kromě recenzí může však jít také o celé články, videa na YouTube, nebo i blogy a weby vedené proti firmám a osobám. Zhrzení zákazníci a zaměstnanci mohou i kontaktovat novináře, kteří o tom napíší nebo natočí reportáž [26].

5.3 Finanční ztráty

Dopady kybernetických útoků mohou mít velké přímé i nepřímé finanční dopady. Konkrétní webová služba může být mimo provoz i několik dní. Pokud není dobře vyřešené zálohování, mohla být některá data nenávratně ztracena. Malware mohl být na webu klidně i několik měsíců a během té doby mohl část návštěvníků přesměrovávat na cizí stránky [25].

6 Bezpečnost proti hackerům

6.1 Práce s hesly z hlediska vývojářů

6.1.1 Hashování

Hash je digitální otisk textu, který je výsledkem hashovací funkce. Má podobu jedinečného shluku čísel a písmen. Používá se všude tam, kde nechceme, aby třetí strana odhalila námi napsanou zprávu. V rámci internetové bezpečnosti se tento způsob šifrování uplatňuje například při zadávání hesel nebo elektronických podpisů [27].

Pokud si zvolíte heslo do e-mailu nebo na webovou stránku, projde hashovacím procesem a do cílové databáze se uloží jako pouhý otisk, který má podobu shluku písmen a čísel. Vždy, když se pomocí hesla někam přihlašujete, hodnota vámi napsaného hesla se algoritmicky porovná s hodnotou šifrovaného otisku [27].

Pokud dojde k prolomení databáze, útočník hesla v podobě hashe nepřečte. Často používaná hesla však může odhalit pomocí slovníku hashů, a proto je důležité používat silná hesla [27].

Hash je bezpečný z následujících důvodů:

- Ať už je vámi zvolené heslo složené z libovolných znaků a libovolně dlouhé, hash bude vždy stejně dlouhý (útočník tak z hashe nemůže rozpoznat délku hesla).
- Pokud v heslu změníte třeba jen jediný znak, hash se kompletně změní.
- Z hashe je nemožné rekonstruovat původní heslo.
- Je velmi nepravděpodobné, že dvěma různým heslům bude odpovídat stejný hash [27].

6.1.2 Solení hesel

Aby bylo prolomení hashů složitější, je dobrý nápad do originálního vstupu vkládat nějaký další řetězec. V ideálním případě náhodný. Tomuto procesu se říká solení hesel [28].

Bezpečnost je založena na myšlence, že útočník nebude moci použít předpočítanou tabulku hesel a hashů, protože nebude znát sůl a bude muset hesla lámat jednotlivě [28].

```
1 $password = 'tajne_heslo';
2 $salt = 'fghjgtzjjhg';
3
4 $hash = md5($password . $salt);
5
6 echo $password; // vypise puvodni heslo
7 echo $hash;     // vypise hash hesla vcetne soli
```

Výpis kódu 2: Solení hesel [28]

6.2 Práce s hesly z hlediska uživatele

Hesla stále představují nejrozšířenější způsob autentizace uživatelů, a proto je vhodné jim věnovat větší pozornost. Heslo je po zadání uživatelem porovnáno s heslem, které uživatel zadal do systému již dříve. Tím vzniká první problém, který spočívá v procesu uložení hesla do systému tak, aby se k němu nedostal útočník, pokud se mu podaří do systému proniknout. Další potencionální riziko představuje i oprávněný správce systému, který má z principu přístup do celého systému a mohl by si tak přečíst heslo uživatele a následně je zneužít [6].
Základní pravidla silného hesla:

- heslo by mělo být složené minimálně z osmi znaků
- mělo by obsahovat malá i velká písmena

- mělo by obsahovat nejméně jednu číslici
- může také obsahovat nějaký symbol např.: ? ! / [29]

Nejlepším heslem jsou na první pohled nepochopitelné shluky písmen. Například: Ko1Le2Di3Pe4Ok5. Jedná se o počáteční písmena jedné dětské písničky (Kočka Leze Dírou Pes Oknem) doplněné pouze o čísla. Toto heslo je prolomitelné jen takzvaným útokem hrubou silou (systematické kombinování možností), kdy jeho odhalení trvá několik milionů let [29].

Máme-li problém se zapamatováním tolika různých hesel k několika různým přihlašovacím účtům, můžeme používat jeden typ našeho silného hesla, ke kterému jednoduše připíšeme znaky identifikující danou službu [29]. Například:

- Ko1Le2Di3Pe4Ok5Fa - Facebook
- Ko1Le2Di3Pe4Ok5Wi - Wi-Fi
- Ko1Le2Di3Pe4Ok5Em - Email
- Ko1Le2Di3Pe4Ok5Sk - Skype

Toto řešení není sice ideální, ale je přeci jen bezpečnější, nežli používání stále téhož hesla [29].

7 Způsoby uložení hesel na počítač uživatele

7.1 Způsoby ukládaní hesel do prohlížeče

7.1.1 Cookies

Cookie může vytvořit buď server (a poslat ji do prohlížeče společně s vygenerovanou stránkou ve formě HTTP hlavičky), nebo samotný prohlížeč při interpretaci stránky pomocí jazyka JavaScript. Prohlížeč následně cookie uloží někam na disk počítače návštěvníka, obvykle do složky dočasných souborů. Cookies se pak přenášejí při každé výměně informací mezi serverem a prohlížečem[30].

Každá cookie obsahuje tyto informace:

- Název – umožňuje odlišit více cookies stejného zdroje od sebe.
- Hodnota – až 4 096 bytů dlouhý text. Kvůli předávání hodnoty cookies vytváříme.
- Datum expirace – čas, kdy cookie vyprší (zmizí z prohlížeče).
- Doména – doména, pro kterou jsou cookies dostupné. Nemusí být uvedena, pak je automaticky nastavena doména, která cookie uložila (ale ke cookie se nedostanou subdomény).
- Cesta – určuje, které stránky dané domény se ke cookie dostanou. Tj. jakou musí mít URL, aby se jim cookie poslala.
- Příznaky – například secure, HttpOnly, SameSite [31]

Atribut secure je volba, kterou může nastavit aplikační server při odesílání nového cookie uživateli v rámci odpovědi HTTP. Účelem atributu secure je zabránit tomu, aby byly soubory cookie zpozorovány neoprávněnými stranami v důsledku přenosu souboru cookie v čistém textu. K dosažení tohoto cíle budou prohlížeče, které podporují atribut secure, posílat soubory cookie s atributem secure, pouze když požadavek směřuje na stránku HTTPS. Řečeno jiným způsobem, prohlížeč neodešle cookie s nastaveným atributem secure přes nešifrovaný požadavek HTTP. Nastavením atributu secure prohlížeč zabrání přenosu souboru cookie přes nešifrovaný kanál [31].

Příznak HttpOnly, je dostupná jen serveru a nelze ji tedy přečíst pomocí JavaScriptu, což znemožňuje ukradení cookie například pomocí útoku cross-site scripting (XSS). [30].
SameSite umožňuje serverům určit, zda/kdy jsou soubory cookie odesílány s požadavky mezi weby (kde je web definován registrovatelnou doménou a schématem: http nebo https). To poskytuje určitou ochranu proti útokům na

padělání požadavků mezi lokalitami (CSRF). Má tři možné hodnoty: Strict, Lax a None [32].

7.1.2 Google Chrome

Google nabízí v operačním systému Android a webovém prohlížeči Chrome synchronizaci hesel. V rámci této funkce nyní přidal kontrolu, která analyzuje přihlašovací údaje, upozorní na nedostatečně bezpečná hesla i jejich případný únik [33].

Přihlašovací údaje jsou porovnávány s databází, obsahující informace o několika milionech kompromitovaných účtů. Google prý kvůli tomu dokonce sleduje „dark web“, kde se šíří kolekce ukradených hesel. Většina údajů v databázi však pochází z dosud známých úniků [33].

Kontrola bezpečnosti hesel tedy začíná na adrese passwords.google.com, kde bude nejprve nutné přihlásit se účtem Google. Uživatel zde může procházet své přihlašovací údaje k jednotlivým webům a službám. Jména a hesla je možné zobrazovat, kopírovat do schránky, upravovat a mazat [33].

7.1.3 Firefox

Účet Firefoxu a služba Firefox Sync umožňují ukládat si a synchronizovat své přihlašovací údaje a také informovat, jestliže je některé heslo je zranitelné. Zároveň hesla chrání pomocí šifrování, takže ani Mozilla je nezná. Firefox naproti tomu prověřuje uložené weby na základě databáze webů, u nichž došlo k úniku dat, aby informoval klienta, jestliže jsou přihlašovací údaje zranitelné [34].

Firefox si může ukládat uživatelská jména a hesla, která jsou používaná k přístupu do internetových služeb jako např. elektronickému bankovnictví nebo e-mailovému účtu. Pokud uživatel počítač sdílí s jinými uživateli, je doporučováno používat hlavní heslo [35].

Vícero zařízení/profilů: Hlavní heslo je nastaveno lokálně a mezi profily či zařízeními se nesynchronizuje. Pokud jsou používány více než jedno zařízení či

profil, každý z nich používá vlastní hlavní heslo [35].

7.2 Ukládání hesla do lokálního úložiště uživatele

7.2.1 Software - 1Password

1Password je správce, který prvně vznikl pro macOS, nyní je ale rozšířený na všechny velké systémy včetně mobilních zařízeních s iOS a Androidem. Poskytuje uživatelům místo pro ukládání různých hesel, softwarových licencí a dalších citlivých informací ve virtuálním trezoru uzamčeném hlavním heslem chráněným PBKDF2 [3] Umožní synchronizaci mezi počítači i mobilními telefony a to včetně možnosti nahrát si do služby své dokumenty, k dispozici je 1 GB prostoru [36].

7.2.2 Software - Sticky Password

Sticky Password nabízí funkce s vysokým zabezpečením, jako je 256bitové šifrování AES [4], dvoufaktorové ověřování (2FA) a možnost místní Wi-Fi synchronizace, dále nabízí odemykání pomocí otisku prstů. Uživatelsky je velmi jednoduchý, ať už díky funkcionality, nebo sdílený hesla mezi ostatními zařízení [37].

7.3 Nebezpečí automatického doplňování hesla v Google Chrome

Ačkoliv jsou hesla uložena na disku v šifrované podobě v době, kdy je uživatel do Windows přihlášen, jsou dostupná v čitelné podobě skrze "Data Protection API" (DPAPI) [38] [39].

soubor C:\Users\%Username%\AppData\Local\Google\Chrome

[3] V kryptografii jsou PBKDF1 a PBKDF2 klíčové derivační funkce s klouzavými výpočetními náklady, které se používají ke snížení zranitelnosti útoků hrubou silou [36]

[4] Pokročilý standard šifrování používaný i americkou armádou. Implementovaný pomocí PBKDF2 s vysokým počtem iterací a kryptografickou solí pro maximální zabezpečení [37].

\User Data\Default\Login Data)

To znamená, že jakýkoliv program (nebo malware) běžící pod jakýmkoliv účtem si může hesla dešifrovat a přečíst. Právě takhle fungují programy jako ChromePass a ChromePasswordDecryptor [38] [39].

Vše je tak závislé na síle hesla do uživatelského Windows účtu. A i to je samozřejmě možné z napadeného systému získat [39].

Jenže pokud do počítače pronikne hacker a stane se tak aktivní pod účtem přihlášeného uživatele, bude mít přesně stejné možnosti k získání hesel jako Chrome. Na toto je potřeba myslet a každý by měl zvážit rizika, pokud by k tomu došlo. Je potřeba brát v potaz i to, že ne každá havěť musí nutně vykrádat hesla [38].

Problém může nastat i v případě, že počítač uživatel nezamyká (kombinace kláves WIN + L), jakmile od něj odchází. Pokud někdo po uživatelovi heslech touží, stačí, aby na neodhlášeném počítači pustil aplikaci typu Chrome Password Decryptor či ChromePass. Veškerá hesla se obratem dozví a jednoduše je může z počítače vynést [38] [39].

8 Přenos dat klient-server

Klient-server (client-server) je architektura distribuované aplikace, která se skládá ze dvou základních komponent: klient a server [40].

Server je komponenta, která poskytuje určitou službu. Službou zde rozumíme například data nebo provedení nějakého výpočtu. K jednomu serveru může být připojen libovolný počet klientů. Klient je komponenta, která se k serveru připojuje, aby této služby využila [40].

Komunikaci zahajuje klient, který k serveru naváže stabilní spojení a odešle svůj první požadavek. Následně může stejné spojení využít i pro požadavky následující. Server pro každého nového klienta vytvoří nový vnitřní stav, který se označuje jako relace (session). Tento stav k němu přiřadí až do doby, než

se klient odpojí. Relace primárně slouží k omezení síťové komunikace - klient se může například přihlásit jen na začátku relace a pak už se na existující přihlášení pouze odvolávat [40].

8.1 Protokol FTP

Přenos dat a souborů je sice poměrně triviální záležitost, nicméně v některých případech umí být problematický. Třeba při komunikaci mezi dvěma systémy, které reprezentují text a data odlišným způsobem nebo mají různou adresářovou strukturu. FTP protokol data přenáší bez ohledu na jejich strukturu ve formě souvislého proudu, tzv. stream mode [41].

Spojení probíhá mezi lokálním a vzdáleným hostem. Lokální host je FTP klient, což je typ softwaru, který se nasadí na konkrétní počítač. FTP klient iniciuje spojení příkazem o připojení k FTP serveru. Na straně serveru musí být rovněž aktivní software, který zajišťuje běh FTP serveru. Ten přijímá požadavky z TCP/IP sítě a odpovídá na ně. Jakmile je spojení zahájeno, může FTP klient nahrávat, stahovat, spravovat nebo odstraňovat jednotlivé soubory z FTP serveru [41].

FTP softwarů existuje několik. Jedním z nejznámějších je FileZilla, což je open source software vhodný pro libovolnou platformu. FTP klientem pro macOS s podporou FTP a SSH je Transmit. Svůj FTP software má i Microsoft – WinSCP [41].

8.2 Protokol SFTP

SFTP (SSH File Transfer Protocol) se často spojuje s FTP, ve skutečnosti je to ale subnet protokolu SSH (Secure Shell), který běží na stejném portu jako SSH, typicky na portu 22. Protokol SSH se tradičně využívá ke vzdálenému přístupu k systému a aplikacím [41].

SFTP přenáší soubory pomocí SSH protokolu. Díky tomu je celý proces je šifrovaný a k ověřování dochází pomocí SSH klíče. Ověření jde navíc pojistit

ještě použitím šifrovaného jména a hesla. Ve spojení s VPN proto tvoří SFTP velmi účinnou ochranu přenosu souborů [41].

Praktická část

9 Základní analýza programu

Nyní k samotnému programu, musel jsem upravit dva přihlašovací moduly, jeden pro administrátorský formulář, kde při selhání přihlášení se začnou zaznamenávat data a druhý je pro uživatelské rozhraní. Needitoval jsem její vizuální stránku, ale zdrojovou.

Při přihlášení neexistuje limit na pokusy, včetně toho, že tu není akronym CAPTCHA, čili uživatel/hacker můžou bez omezení zkoušet se přihlásit na Bobříka informatiky, a to platí jak pro administrátorské rozhraní, tak i uživatelské.

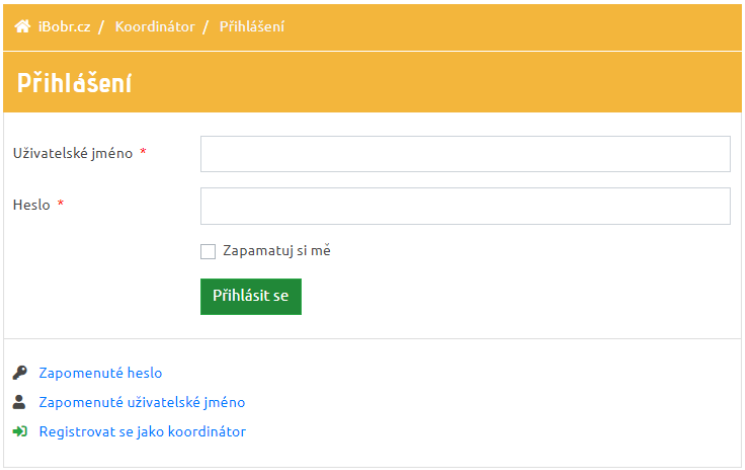
Poté jsem vyvinul třetí modul, který čistí tabulku záznamu od nevyžádaných dat v databázi Bobříka informatiky. To znamená, že jsem mezi těmi nasbíranými informacemi nechtěl, aby v záznamech zůstali obyčejný uživatelé co zapomněli svá hesla, ale pouze hackeři.

Dále byla vyvinuta tabulka pro ukládaní dat, se kterou komunikují všechny tři moduly výše zmíněné.

9.1 Přihlašovací portály

9.2 User

Na obrázku můžeme vidět uživatelské rozhraní, které jsem upravoval a veškeré neúspěšné pokusy o přihlášení zaznamenával.



Obrázek 3: Přihlašování na uživatelském prostředí

9.3 Administrátor

Následné prostředí slouží pro přihlášení správce, jenž má na starost webové služby. Konkrétní rozhraní bude pravděpodobně největším terčem útoků ze strany hackerů a proto je důležité mít přísnější kritéria pro lepší sběr dat.

Obrázek 4: Přihlašování na administrátorském prostředí

10 Návrh a tvorba řešení

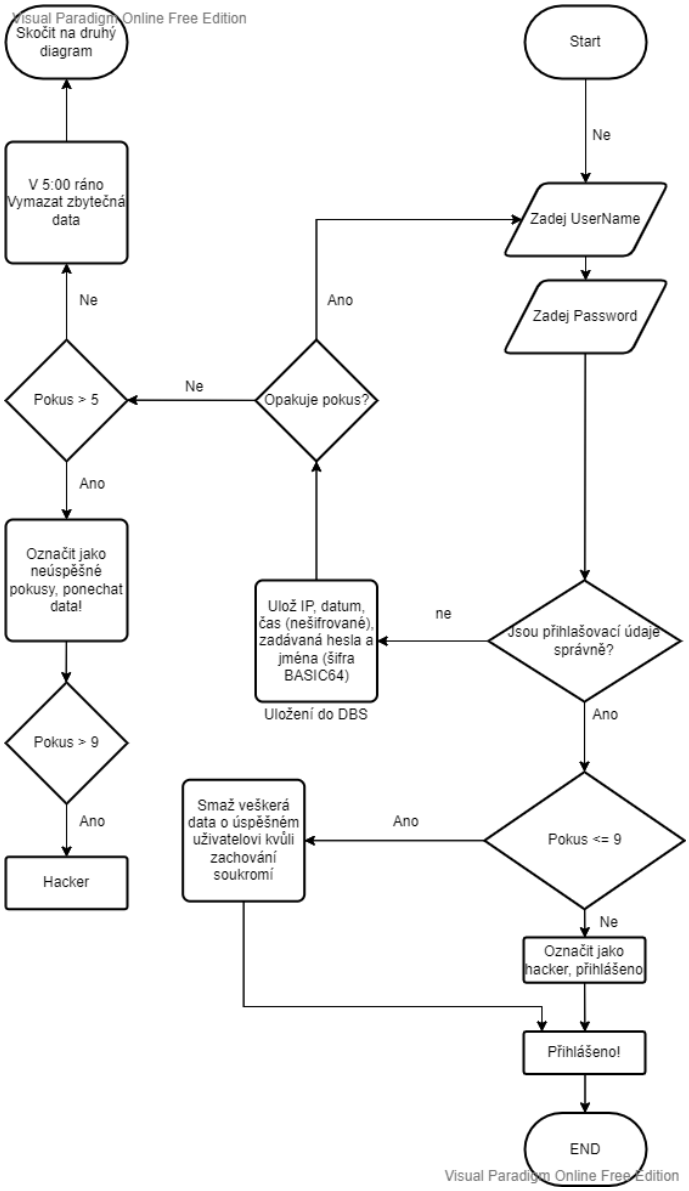
10.1 Diagram pro uživatelské prostředí

První část začíná inputem, konkrétně když uživatel zadá špatné jméno, nebo heslo do formuláře, tak v tu chvíli se modul aktivuje [5]. Začíná tedy zapisovat do tabulky v databázi Bobříka informatiky. Zjistí si jeho IP adresu, stát, čas pokusu a započítává jednotlivé série útoků, které si definujeme později. Zároveň zaznamenávají jména a hesla, jenž jsou zašifrovaná pomocí base64. To vše se děje, za předpokladu, že uživatel zadal jméno, nebo heslo špatně. Nicméně, pokud zadá heslo/jméno špatně desetkrát, data zůstanou v záznamu na trvalo. V opačném případě, tedy méně než deset pokusů se data v pět hodin ráno smažou.

Obdobně je to i v administrátorském modulu, kde podmínky jsou přísnější a to následovně: Data se ponechají, pokud pokusů je více než tři, nebo uživatelské jméno je rovno ”admin”. V opačném případě se opět smažou, jako v uživatelském modulu, aby se odstranila zbytečná data nechtěných [6] pokusů. Čili každý se může s heslem zmýlit a ne vždycky funguje přihlášení na první pokus.

[5] Pokud se to odehrává na administrátorském, aktivuje se upravený modul na administrátorovi a pokud na uživatelském, tak uživatelovi

[6] Může se jednat například o učitele, který si nemohl vzpomenout na heslo



Obrázek 5: Návrh diagramu pro přihlášení

10.2 MySQL tabulka

Při vytváření nové tabulky v Joomla, je důležité zvážit správnou předponu a název, aby nedošlo k zbytečným komplikacím, třeba že se někdo rozhodne smazat mojí tabulku. Pojmenoval jsem jej tedy jako jos_hacker a musel promyslet, co budu za data ukládat do tabulky.

10.3 Návrh MySQL tabulky

Bávrhu byl následující, je třeba znát důležité informace ohledně jednotlivých přihlašovacích pokusů. Tedy, jestli ten, kdo se přihlašuje, je obyčejný uživatel, jenž zapomněl heslo, nebo naopak je to hacker, který se bude snažit prolomit do serveru. Další důležitou částí je, co za data se budou ukládat, která z nich se budou uchovávat a která naopak se smažou. Při budoucí analýze nasbíraných dat jsem musel mít SQL tabulku vyfiltrovanou jenom na podezřelé útoky, respektive hackery. Tím je myšleno, aby se mezi nimi nenacházeli mylné pokusy o průnik, například učitel, který zapomněl heslo.

Potřebuji znát útočníkovu IP adresu. To pomáhá určit, odkud útok probíhá. Dále je třeba vědět, na jakou webovou stránku se přihlašuje, jestli uživatelskou - tedy označováno jako H (hlavní), nebo jako administrátorské rozhraní, označováno jako A. Poté co zadává za jméno ($username) a co za heslo ($password). Je třeba apelovat i na to, že data při ukládání do tabulky, musí být chráněná proti odcizení a náročnosti překladu, tedy zvolil jsem šifrování base64 kvůli rozsáhlé podpory programů, jako je Excel (pomocí visual basic). Data obsahující přihlašující údaje nikdy nesmí zůstat odhalená [7].

V další dílčí práci zjišťuji, odkud se daný jedinec přihlašuje, tedy jeho lokaci ($country), avšak tahle informace může být mylná a to z toho důvodu, že hacker může využívat služby VPN a místo ČR se připojovat z Číny. Dále tu je důležitý parametr, tím je id_ip, pomocí kterého dokážu rozdělit jednotlivé útoky na jednotlivé série pokusů o průnik. K přidělení id_ip uživateli dochází tehdy, když současný útok skončil před deseti minutami a začal znovu zkoušet po daném čase. V tu chvíli se jeho id_ip změní na novou a tím vzniká nová série útoků.

Pomocí booleovského parametru $isHacker, se dá určit, jestli se jedná o hackera, nebo ne. Hackerem se stane tehdy, když jeho pokusy překročí daný limit [8], taktéž u administrátora stačí podmínka, aby do uživatele zadal přihla-

[7] Kdyby došlo k prolomení, tak ať ta data, konkrétně jména a hesla nejsou snadná

[8] U Administrátorského rozhraní stačí aby pokusů bylo více než tři a u administrátorského

šovací jméno ”admin” a tím je záznam označen jako hacker. Dále tu je $Attempting(typu boolean), který se automaticky nastaví na ”true” pouze tehdy, pokud uživatel, či hacker se stále pokouší přihlašovat před zahájení čištění tabulky v databázi Bobříka informatiky od nevyžádaných logů a to rovno, nebo méně 15 minut.

id 1	id_ip	web	attempting	isHacker	attention	logged	IP	country	date	username	password
12187	12186	H	0	0	2	0	109.81.213.168	Czechia	2022-01-17 20:09:37	YWZlZmF3ZQ==	ZmF3ZWY=
12186	12186	H	0	0	2	0	109.81.213.168	Czechia	2022-01-17 20:09:35	YXN3ZXRhdw==	U3VtbWVyc2J5MjAxOQ==

Obrázek 6: Zobrazení tabulky

Na základě tohoto rozvržení, jsem už byl schopný ke sběru dat, které následně poslouží v pozdější analytické fázi.

devět.

11 Implementace

11.1 Modul pro uživatelské prostředí

11.1.1 Úprava modulu

Modul pro uživatelské rozhraní, se nachází v komponentách, konkrétně: /.../components/com_users/controllers, kde jsem editoval soubor user.php. Dále mi k tomu pomohl vývojový diagram, viz 5.

11.2 Import balíčků

Na vrcholu kódu, jsem importoval balíčky. PHP metody a syntaxe jsou kompatibilní, ale je třeba dbát na bezpečnost kódu, proto se doporučuje pracovat skrze JInput, viz 2.2.1.

```
use Joomla\CMS\Date\Date;
use Joomla\CMS\Factory;
```

Tyto funkce definujeme těsně pod definování JEXEC [9].

11.3 Zápis základních dat do tabulky MySQL

Kód níže, značí počátek kódu, který jsem vytvořil těsně pod neúspěšného pokusů o přihlášení uživatele v podmínce.

```
1 if (true !== $app->login($credentials, $options))
2 {
3 ...zde jsem upravoval...
4 }
```

Výpis kódu 3: Podmínka selhání přihlášení

Při selhání přihlášení uživatele, ukládám data do tabulky jos_hacker viz 6, prvním parametrem, je zjištění IP adresy uživatele, je využita konstrukce z Joomly

[9] konstanta, která je obvykle definována v indexu. php v kořenovém adresáři Joomla [1]

pomocí JFactory a to z hlediska bezpečnosti, než je PHP příkaz GET/SET. Další podmínkou je, aby záznam vždy začínal výchozí hodnotou $isHacker [10]. Dále zapisuji adresu serveru[11], datum a zašifrované jméno i heslo pomocí base64. Následní příkaz je realizování úkonů pro vkládání do tabulky.

```
1  $ip = JFactory::getApplication()->input->server->
2  get('REMOTE_ADDR','');
3  $hacker = new stdClass();
4  $hacker->isHacker=0;
5  $hacker->ip=$ip;
6  $hacker->web="H";
7  $hacker->date=date("Y-m-d H:i:s");
8  $hacker->username=base64_encode($data['username']);
9  $hacker->password=base64_encode($data['password']);
10 $result = JFactory::getDbo()->insertObject('#__hacker',
11 $hacker);
```

Výpis kódu 4: Ukládání proměnných do tabulky jos_hacker

11.4 Podmínky pro určování série útoků

Definice série jednotlivých útoků, byla následující. Hackerův útok může trvat několik pokusů za krátký, či dlouhý časový úsek. Proto jednotlivé pokusy útoků jsou rozdělené a to následujícím způsobem:

Pokud útočník se snaží nepřetržitě po dobu x sekund zkoušet se přihlásit, jeho série musí někdy skončit a tím pravidlem je tedy patnáct minut.

Pokud nebude žádná odezva po celých patnácti minut a uživatel se bude pokoušet znovu přihlásit, bude to bráno už jako jiná série útoků, v tomto případě je to označováno jako 'id_ip'.

Dále zjišťuji, zda existuje datum od posledního útoku určitého uživatele.

[10] ishacker je rovno 0

[11] A - administrátorský, H - Hlavní/uživatelský

Zmiňovaný kód je vytvořen na počátku modulu v podmínce, kde selhal pokus o přihlášení uživatele (viz 4).

```
1 $db = JFactory::getDbo();
2 $query = $db
3    ->getQuery(true)
4    ->select('date')
5    ->from($db->quoteName('#__hacker'))
6    ->where($db->quoteName('ip') . " = " . $db->quote($ip))
7    ->order($db->quoteName('date') . ' desc');
8 $db->setQuery($query);
9 $lastDateFromIP = $db->loadResult();
```

Výpis kódu 5: Existující záznam

Pokud tedy žádný datum neexistuje, čili výsledek je null, takže je nastavena podmínka, aby datum byl starší, než je aktuální kvůli budoucí kontrole. Nyní je přidělován poslední datum k určitému záznamu.

```
1 if($lastDateFromIP == 0 || $lastDateFromIP == null)
2 {
3    $lastDateFromIP = Date::getInstance(date("2000-09-11
4    5:00:00"));
5 }
```

Výpis kódu 6: Podmínka pro neexistující záznam

Definoval jsem interval patnácti minut pod kódem, kde se vytváří uživatel (viz 4). Tímto způsobem se vybírá poslední datum určitého záznamu uživatele a přičetl k tomu dalších patnáct minut.

Dalším důvodem proč vytvářet podmínky pro série útoků, je ten, že server poskytující adresu států útočníků pomocí volané IP adresy, by neměl být zahlcován informacemi každého jednotlivého pokusu o přihlášení, proto se vždy přiřazuje ke stejnému uživateli stejná lokalita. To znamená, že nedoporučuji[12]

[12]Už jednou došlo k zablokování na testovacím serveru

zatěžovat podpůrní cizí server [13], protože následně může dojít k zablokování poskytovaných služeb a proměnná, konkrétně $country by byla vždy prázdná.

Datum, který je volán pomocí konstrukce JFactory, má v sobě zabudovaný ošetření proti přesunutí času. Tedy změna letního času na zimní a naopak, neovlivní běh kódu a tudíž nehrozí selhání zápisu dat.

```
1 $currentTimeSeries =JFactory::getDate(date("Y-m-d H:i:s"));
2 $intervalMinutes = new \DateInterval('PT15M');
3 $dateWithMinutesControler = new Date($lastDateFromIP);
4 $dateWithMinutesControler->add($intervalMinutes);
```

Výpis kódu 7: Interval pro přidání času

Byla vytvořena základní kostra kódů pro ukládání záznamů. A začal jsem sepisovat podmínky. Nejprve samotná restrikce, pod kterou bude patřit celý kód.

Pokud je aktuální čas větší, než poslední záznam dané IP adresy, v tu chvíli se vytvoří parametr $country s jeho aktuální pozici hackera. Využívám k tomu webovou adresu stránky Geoplugin.

```
1 if($currentTimeSeries > $dateWithMinutesControler)
2 {
3    $addr_details = @unserialize(file_get_contents(
4    'http://www.geoplugin.net/php.gp?ip='.$ip));
5    $country = $addr_details['geoplugin_countryName'];
6    ...
7 }
```

Výpis kódu 8: Podmínka pro kontrolu série útoků

Dále bylo třeba zjistit poslední ID uživatele, čili porovnávám parametr $ip z tabulky s parametrem 'ip', který je volán z kódů výše (viz 4).

[13] Adresa podpůrného severu: http://www.geoplugin.net

```
1  ...
2  $db = JFactory::getDbo();
3  $query = $db
4     ->getQuery(true)
5     ->select('id')
6     ->from($db->quoteName('#__hacker'))
7     ->where($db->quoteName('ip') . " = " . $db->quote($ip))
8     ->order($db->quoteName('id') . ' desc');
9  $db->setQuery($query);
10 $lastid = $db->loadResult();
```

Výpis kódu 9: Zjištění poslední $ID

Pod předchozí kód byl vytvořen další dotaz, který už naopak začne přiřazovat nové hodnoty k 'id_ip' a 'country'.

```
1  ...
2  $db = JFactory::getDbo();
3  $query = $db->getQuery(true);
4  $conditions = array(
5  $db->quoteName('id_ip') . ' = ' . $db->quote('0'));
6  $fields = array(
7  $db->quoteName('id_ip') . ' = ' . $db->quote($lastid),
8  $db->quoteName('country') . ' = ' . $db->quote($country));
9  $query->update($db->quoteName('#__hacker'))->set($fields);
10 $query->where($conditions);
11 $db->setQuery($query);
12 $result = $db->execute();
```

Výpis kódu 10: Přiřazení parametrů do tabulky jos_hacker

Kód výše funguje za předpokladu, že neexistuje žádný záznam o daném útočníkovi.

Za podmínky, že datum existuje, jsem sepsal kód, který zavolá poslední ID

a následně zjišťuji $lastCountry. Tímto minimalizuji zátěž pro podpůrný server Geoplugin, kdy už nemusím odesílat data, abych opakovaně zjišťoval stát.

```
else
{
$db = JFactory::getDbo();
$query = $db
   ->getQuery(true)
   >select('id_ip')
   ->from($db->quoteName('#__hacker'))
   ->where($db->quoteName('ip')." = ".$db->quote($ip),)
   ->order($db->quoteName('id_ip') . ' desc');
$db->setQuery($query);
$lastid = $db->loadResult();

$db = JFactory::getDbo();
$query = $db
   ->getQuery(true)
   ->select('country')
   ->from($db->quoteName('#__hacker'))
   ->where($db->quoteName('ip')." = ".$db->quote($ip),)
   ->order($db->quoteName('id_ip') . ' desc');
$db->setQuery($query);
$lastCountry = $db->loadResult();
...
}
```

Výpis kódu 11: Iniciování proměnných $lastID a $lastCountry

Na závěr vytvářím poslední dotaz, který aktualizuje data parametrů 'id_ip' k $lastID a 'country' k $lastCountry.

```
...
$db = JFactory::getDbo();
```

```
3  $query = $db->getQuery(true);
4  $conditions = array(
5  $db->quoteName('ip') . ' = ' . $db->quote($ip),
6  $db->quoteName('id_ip') . ' = ' . $db->quote('0'));
7  $fields = array(
8  $db->quoteName('id_ip') . ' = ' . $db->quote($lastid),
9  $db->quoteName('country').' = '.$db->quote($lastCountry));
10 $query->update($db->quoteName('#__hacker'))->set($fields);
11 $query->where($conditions);
12 $db->setQuery($query); $result = $db->execute();}
```

Výpis kódu 12: Aktualizace parametrů lastID a lastCountry do tabulky

11.5 Porovnávání, zda se již uživatel přihlásil

Touto proměnnou zjišťuji, zda uživatel, jenž se pokouší přihlásit, nebo nabourat do systému, se připojil. To pomáhá určit, zda se jedná o obyčejného uživatele, nebo ne. Po předchozím kódu iniciuji proměnou $logged pomocí dotazu.

```
1  ...
2  }
3  $db = JFactory::getDbo();
4  $query = $db
5     ->getQuery(true)
6     ->select('logged')
7     ->from($db->quoteName('#__hacker'))
8     ->where($db->quoteName('ip') . " = " . $db->quote($ip))
9     ->order($db->quoteName('id') . ' desc');
10 $db->setQuery($query);
11 $logged = $db->loadResult();
```

Výpis kódu 13: Načtení parametru $logged

V další dílčí částí následuje podmínka, která porovnává, zda se již uživatel přihlásil a pokud ano, ke všem jeho stejným IP adresám, se přiřadí právě hodnota $logged na 1. To bude fungovat i za předpokladu, že vznikne nový záznam o dané sérii útoků.

```
1  if($logged > 0)
2  {
3     $db = JFactory::getDbo();$query = $db->getQuery(true);
4     $conditions = array(
5     $db->quoteName('ip') . ' = ' . $db->quote($ip));
6     $fields = array(
7     $db->quoteName('logged') . ' = ' . $db->quote('1'),
8  );
9  $query->update($db->quoteName('#__hacker'))->set($fields);
10 $query->where($conditions);
11 $db->setQuery($query);
12 $result = $db->execute();
13 }
```

Výpis kódu 14: Podmínka pro přihlášení uživatele

11.6 Počítání jednotlivých pokusů

Vyvinul jsem metodu, která zajistí počítání jednotlivých pokusů a přiřazuje je právě k určitému sérii útoků 'id_ip'. Nepočítá to od počátků věků. Tato funkce označuje uživatele za hackery, poněvadž, když počet pokusů v uživatelském rozhraní je více než 9, znamená to, že je hacker a přiřadíme hodnotu proměnné $isHacker na true. Taktéž jako u logu (viz 14), platí i pro nově vytvořené záznamy.

Tímto dotazem zjišťuji celkový počet pokusů daného jedince z určitého série útoků.

```
$db = JFactory::getDbo();
$query = $db
   ->getQuery(true)
   ->select('COUNT(*)')
   ->from($db->quoteName('#__hacker'))
   ->where($db->quoteName('id_ip') . " =
   " . $db->quote($lastid));
$db->setQuery($query);
$count = $db->loadResult();
```

Výpis kódu 15: Dotaz pro počítání záznamů

Dále je tu podmínka, jestli počet pokusů není větší než 9 a pokud ano, udělíme booleovské proměnné $ishacker hodnotu true.

```
if($count > 9){$isHacker = 1;}
```

Výpis kódu 16: Podmínka určující $isHacker

V poslední části data jednotlivých záznamů aktualizuji. Iniciuji proměnnou $count, a zároveň zjišťuji, jestli se jedná o hackera. Na závěr je další dotaz, jenž má za úkol aktualizovat veškeré záznamy daného uživatele a dané série útoků $id_ip.

```
$db = JFactory::getDbo();
$query = $db->getQuery(true);
$conditions = array(
$db->quoteName('ip') . ' = ' . $db->quote($ip),
$db->quoteName('id_ip') . ' = ' . $db->quote($lastid),);
$fields = array(
$db->quoteName('attention') . ' = ' . $db->quote($count),
$db->quoteName('isHacker') . ' = ' . $db->quote($isHacker),
);
$query->update($db->quoteName('#__hacker'))->set($fields);
$query->where($conditions);
```

```
12 $db->setQuery($query);
13 $result = $db->execute();
```

Výpis kódu 17: Aktualizace $isHacker a $count v tabulce

11.7 Zamknutí záznamu před čištění tabulky v Databázi

V poslední fázi modulu user.php, kdy stále dochází k neúspěšnému přihlášení uživatele, jsem nastavil podmínky, které ochrání danou sérii útoků před smazáním z databáze. Kdyby hackerův útok začal již ve 4 hodiny ráno a jeho útoky pokračují jednou krát za deset minut, tak automatické čištění tabulky by daný záznam smazal, aniž by tentýž útok byl dokončen, protože daná série útoků by nestihla splnit podmínku pro označení za hackera, tedy za předpokladu, že se nejedná jen o obyčejného uživatele.

Staticky je definovaný čas, kdy bude databáze každý den v pět hodin ráno vyčištěna od nepotřebných záznamů.

```
$lockDate = Date::getInstance(date("5:00:00"));
```

Další je dotaz, který opět iniciuje poslední datum daného hackera, následně také interval patnácti minut, tedy k jeho datu přičtu dalších patnáct minut, který porovnávám s $lockDate.

```
1  $db = JFactory::getDbo();
2  $query = $db
3     ->getQuery(true)
4     ->select('date')
5     ->from($db->quoteName('#__hacker'))
6     ->where($db->quoteName('ip') . " = " . $db->quote($ip))
7     ->order($db->quoteName('date') . ' desc');
8  $db->setQuery($query);
9  $dateFromIP = $db->loadResult();
10
11 $intervalMinutes = new \DateInterval('PT15M');
```

```
12 $dateWithMinutes = new Date($dateFromIP);
13 $dateWithMinutes->add($intervalMinutes);
14 $currentTime = JFactory::getDate(date("Y-m-d H:i:s"));
```

Výpis kódu 18: Dotaz pro zavolání posledního času daného jedince a vytvoření intervalu

Poté je větší podmínka, jejíž úkolem je zajistit, aby připisovala proměnnou $Attempting rovno jedné, a to pouze k určeným záznamu. Tedy nesmělo se stát, že celý záznam, který obsahuje identické IP, najednou bude mít $Attempting rovno jedna. Protože v tomto případě, se počítají pouze série útoku, jenž jsem definoval jako 'id_ip'. Na závěr potřebuji vytvořit dotaz, který mi aktualizuje danou sérii útoku na hodnotu, že se útočník stále před pročištění databáze pokouší přihlašovat.

```
1  if($currentTime <= $lockDate)//pokud soucasny datum je
2  mensi, nebo rovno datu k uzamceni
3  {
4     if($lockDate  <= $dateWithMinutes)//zaroven datum
5     uzamceny je mensi nebo rovno datumu posledniho zaznamu
6     uzivatele
7     {
8        $db = JFactory::getDbo();
9        $query = $db->getQuery(true);
10       $conditions = array(
11       $db->quoteName('ip') . ' = ' . $db->quote($ip),
12       $db->quoteName('id_ip') . ' = ' .
13       $db->quote($lastid));
14       $fields = array(
15       $db->quoteName('attempting') . ' = ' .
16       $db->quote('1'),
17       );
18       $query->update($db->quoteName('#__hacker'))->
```

```
19          set($fields);
20          $query->where($conditions);
21          $db->setQuery($query);
22          $result = $db->execute();
23      }
24  }
```

Výpis kódu 19: Podmínka a dotaz pro záznam $attempting

V tuto chvíli byl vyvinut kód pro neúspěšné přihlášení, poté je řešena druhá část (úspěšné přihlášení), a to i za předpokladu, že uživatel má booleovskou proměnnou $isHacker rovno true.

11.8 Úspěšné přihlášení uživatele

V této dílčí části je řešeno úspěšné přihlášení uživatele. Úkolem modulu user.php není jenom zaznamenávat data, ale i její promazání před neadekvátními informacemi. Hlavním cílem jsou záznamy, které nejsou označené jako $isHacker rovno true.

Nejprve zjišťuji IP adresu, poté proměnnou $lastID a začínám získávat data posledního přihlášení. Je to velmi podobné, jako v předchozích kapitolách, při neúspěšném pokusu o přihlášení (viz 7).

```
1  $ip = JFactory::getApplication()->input->
2  server->get('REMOTE_ADDR','');
3
4  $db = JFactory::getDbo();
5  $query = $db
6      ->getQuery(true)
7      ->select('id_ip')
8      ->from($db->quoteName('#__hacker'))
9      ->where($db->quoteName('ip') . " = " . $db->quote($ip),)
10 ->order($db->quoteName('id_ip') . ' desc');
```

```
11 $db->setQuery($query);
12 $lastid = $db->loadResult();
13
14 $db = JFactory::getDbo();
15 $query = $db
16    ->getQuery(true)
17    ->select('COUNT(*)')
18    ->from($db->quoteName('#__hacker'))
19    ->where($db->quoteName('ip') . " = " . $db->quote($ip),
20 $db->quoteName('id_ip') . " = " . $db->quote($lastid))
21    ->order($db->quoteName('id') . ' desc');
22 $db->setQuery($query);
23 $count = $db->loadResult();
24
25 $db = JFactory::getDbo();
26 $query = $db
27    ->getQuery(true)
28    ->select('date')
29    ->from($db->quoteName('#__hacker'))
30    ->where($db->quoteName('id_ip') . " = " .
31    $db->quote($lastid))
32    ->order($db->quoteName('date') . ' DESC');
33 $db->setQuery($query);
34 $dateFromIP = $db->loadResult();
35
36 $intervalHours = new \DateInterval('PT30M');
37 $thirtyMinutes = new Date($dateFromIP);
38 $thirtyMinutes->add($intervalHours);
39 $currentTime = JFactory::getDate(date("Y-m-d H:i:s"));
```

Výpis kódu 20: Dotaz pro zjištění času a interval 30-ti minut

Při definování a zjišťování základních hodnot z daného záznamu, byly sepsány

podmínky. Jejich úkolem je kontrolovat, zda počet pokusů je menší, nebo rovno devíti a zároveň daný čas jedince je menší, nebo rovno intervalu třiceti minut.

Pokud je podmínka splněna, dojde k promazání záznamu, kteří nejsou podezřelí z pokusu o průnik do systému. V opačném případě, jsou označený booleovskou proměnnou $logged rovno true.

```
1  if($count <= 9 && $currentTime <= $thirtyMinutes)
2  {
3  $db = JFactory::getDbo();
4  $query = $db->getQuery(true);
5  $conditions = array(
6     $db->quoteName('isHacker') . ' = 0',
7     $db->quoteName('ip') . ' = ' . $db->quote($ip),
8     $db->quoteName('id_ip') . ' = ' . $db->quote($lastid),);
9  $query->delete($db->quoteName('#__hacker'));
10 $query->where($conditions);
11 $db->setQuery($query);
12 $result = $db->execute();
13 }
14 else
15 {
16 $db = JFactory::getDbo();
17 $query = $db->getQuery(true);
18 $conditions = array(
19 $db->quoteName('ip') . ' = ' . $db->quote($ip));
20 $fields = array(
21 $db->quoteName('logged') . ' = ' . $db->quote('1'),
22 );
23 $query->update($db->quoteName('#__hacker'))->set($fields);
24 $query->where($conditions);
25 $db->setQuery($query);
26 $result = $db->execute();}
```

Výpis kódu 21: Kontrolní podmínka pro předčasnému čištění databáze

Sestrojený kód je nyní plně funkční v uživatelském prostředí.

11.9 Modul pro administrátorské prostředí

11.9.1 Změny oproti uživatelskému prostředí

Jak jsem již zmínil, tento modul se od uživatelské prostředí neliší, vyjma na pár změn. První věcí je, že se jinak získávají atributy z přihlašovacího formuláře, konkrétněji přihlašovací jméno a heslo pomocí funkci $credentials, namísto $data.

```
//z user.php
$hacker->username=base64_encode($data['username']);
$hacker->password=base64_encode($data['password']);
vs.
//z controller.php
$hacker->username=base64_encode($credentials['username']);
$hacker->password=base64_encode($credentials['password']);
```

Je to způsobeno tím, že administrátorské rozhraní, se nachází v jiné komponentě, konkrétně ve složce administrator. Dále se změnila hodnota 'web', z 'H', na 'A'. Adresa souboru se nachází:

```
\...\administrator\components\com_login\controler.php
```

Stejně jako v předchozím modulu, byly importovány balíčky na začátku kódu viz 11.2. Dále se to liší strukturou celkové kostry, to znamená, že v user.php jsme měly typicky selhání přihlášení hned výše na začátku programu, zatím co zde je to naopak, kde máme neúspěšné přihlášení téměř níže v kódu.

Zpřísnil jsem čištění záznamu z databáze. Důvodem je, že se jedná o administrátorské prostředí, kde jsou hackeři většinou nejvíce aktivní a snaží se

prolomit do tohoto segmentu. Taktéž stačí pouze tři neúspěšné pokusy, aby byla změněná booleovská proměnná $isHacker na true.

11.10 Čištění databáze pomocí Crone

11.10.1 Použití crontab

Pro použití příkazu crone, který bude spouštět vybraný skript, tedy modul pro čištění tabulky v databázi, jsem využíval crontab, je to hlavně doporučeno [42]. Syntaxe je tedy následující:

1 2 3 4 5 /path/to/command arg1 arg2

kde:

- 1: Minuty (0-59)
- 2: Hodiny (0-23)
- 3: Dny (0-31)
- 4: Měsíce (0-12 [12 == Prosinec])
- 5: Dny víkendu(0-7 [7 nebo 0 == neděle])
- /path/to/command – Skript, nebo příkaz
- *: tento operátor specifikuje všechny možné hodnoty pro pole. Hvězdička v poli v hodinovém času, odpovídá každé hodině, nebo hvězdička v poli měsíci, odpovídá každému měsíci [42].

Příklad:

0 5 * * * /root/hacker-cleaner.php

Každý den, v pět hodin ráno , se spustí soubor hacker-cleaner.php.

11.10.2 Vytvoření třetího modulu pro čištění tabulky v databázi

Tento modul byl vyvinut pouze z jednoho důvodu. To znamená, že když pokaždé, co je zavolán vnějším vlivem, chcete li Linux cronem, byl schopen pročistit tabulku hacker od nevyžádaných dat. Crone každý den v pět hodin ráno bude právě pracovat s tímto modulem.

Název modulu jsem pojmenoval jako hacker_cleaner.php a musel definovat strukturu modulu. Stejně jako v modulu user.php, importujeme stejné balíčky.

11.10.3 Kód pro čištění tabulky

Při vytváření dotazu, jsem musel nejprve vytvořit zašifrované proměnné, abych dokázal roztřídit ta správná data, konkrétně "ibobr" a "admin", čili na základě těchto parametrů třídím záznamy neúspěšných pokusů.

```
1 $ibobr = base64_encode('ibobr');
2 $admin = base64_encode('admin');
```

Výpis kódu 22: Zašifrování jména a hesla

Další dílčí částí vytvářím dotaz pro vyhledávání v tabulce, kdy všechny záznamy obsahující "admin", nebo "ibobr", jsou automaticky přiděleny booleovskou hodnotou 'isHacker' na true.

```
1
2 //update condition, FIND all HACKERS IBOBR
3 $db = JFactory::getDbo();
4 $query = $db->getQuery(true);
5 $conditions = array(
6    $db->quoteName('attempting') . ' = ' . $db->quote('0'),
7    $db->quoteName('username') . ' = ' . $db->quote($ibobr),
8    $db->quoteName('country').' != '.$db->quote('Czechia'),
9 );
10 $fields = array(
11    $db->quoteName('isHacker') . ' = ' . $db->quote('1'),
```

```
12 );
13 $query->update($db->quoteName('#__hacker'))->set($fields);
14 $query->where($conditions);
15 $db->setQuery($query);
16 $result = $db->execute();
17
18 //update condition, FIND all HACKERS ADMIN
19 $db = JFactory::getDbo();
20 $query = $db->getQuery(true);
21 $conditions = array(
22    $db->quoteName('attempting') . ' = ' . $db->quote('0'),
23    $db->quoteName('username') . ' = ' . $db->quote($admin),
24    $db->quoteName('country').' != '.$db->quote('Czechia'),
25 );
26 $fields = array(
27    $db->quoteName('isHacker') . ' = ' . $db->quote('1'),
28 );
29    $query->update($db->quoteName('#__hacker'))->
30    set($fields);
31    $query->where($conditions);
32    $db->setQuery($query);
33    $result = $db->execute();
```

Výpis kódu 23: Nalezení a aktualizování všech záznamu obsahující ibobr a admin

Poté je řešeno mazání všech záznamů, které nejsou označený jako 'isHacker' rovno true, a zároveň mají proměnnou 'attempting' rovno false [14]. Vztahuje se to jak na administrátorský web 'A', tak i uživatelský web 'H'.

[14] Čili se nesnaží před vyčištění tabulky přihlásit, v tedy 15 minut.

```
$db = JFactory::getDbo();
$query = $db->getQuery(true);
$conditions = array(
   $db->quoteName('web') . ' = ' . $db->quote('A'),
   $db->quoteName('isHacker') . ' = 0',
   $db->quoteName('attempting') . ' = 0');
$query->delete($db->quoteName('#__hacker'));
$query->where($conditions);
$db->setQuery($query);
$result = $db->execute();

$db = JFactory::getDbo();
$query = $db->getQuery(true);
$conditions = array(
   $db->quoteName('web') . ' = ' . $db->quote('H'),
   $db->quoteName('isHacker') . ' = 0',
   $db->quoteName('attempting') . ' = 0');
$query->delete($db->quoteName('#__hacker'));
$query->where($conditions);
$db->setQuery($query);
$result = $db->execute();
```

Výpis kódu 24: Čištění záznamů

V poslední částí tohoto kódu je, aby všechny pokusy, které mají booleovskou proměnou $attempting označenou jako true, se změnily na false. Důvod je ten, že kdyby se nejednalo o hackera, tak aby ten záznam nezůstal imunní proti vyčištění tabulky.

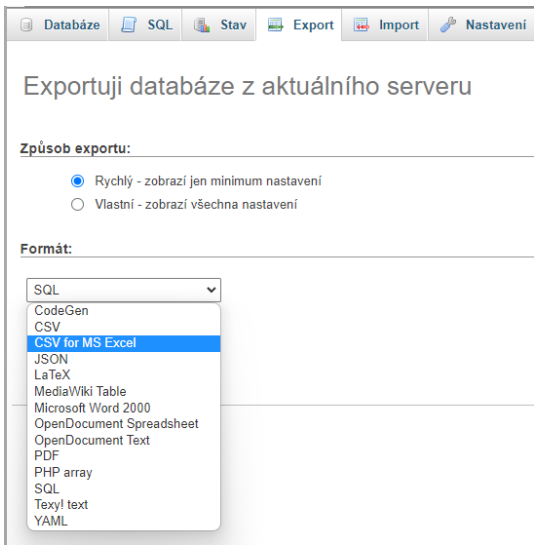
```
$db = JFactory::getDbo();
$query = $db->getQuery(true);
$conditions = array(
   $db->quoteName('isHacker') . ' = 0',
```

```
5     $db->quoteName('attempting') . ' = 1');
6 $fields = array(
7     $db->quoteName('attempting') . ' = ' . $db->quote('0'),
8 );
9 $query->update($db->quoteName('#__hacker'))->set($fields);
10 $query->where($conditions);
11 $db->setQuery($query);
12 $result = $db->execute();
```

Výpis kódu 25: Anulování $attempting

11.11 Stahování dat z MySQL

Tak jako každá databáze, i MySQL na adrese www.adresa/phpmyadmin, můžeme data nejen importovat, ale i exportovat. Výhodou je, že když chceme data stáhnout, můžeme jednoduše zvolit její formát, do kterého se nasbíraná data konvertují. V mém případě to je Excel, tedy formát .CSV from MS Excel, protože později s nimi lze manipulovat (viz 7).



Obrázek 7: Export databáze do MS Excel

11.12 Excel a dešifrování dat

K dešifrování hesel a jmen z tabulky jos_hacker byly využity služby Microsoft Excel, kde byl aktivovaný vývojářský režim na němž jsem si vytvořil makro, které mi konvertuje z base64 na UTF-8.

```
1
2  Function DecodeBase64(b64$)
3      Dim b
4      With CreateObject("Microsoft.XMLDOM").createElement(
5      "b64")
6          .DataType = "bin.base64": .text = b64
7          b = .nodeTypedValue
8          With CreateObject("ADODB.Stream")
9              .Open: .Type = 1: .Write b:
10              .Position = 0: .Type = 2: .Charset = "utf-8"
11             DecodeBase64 = .ReadText
12             .Close
13         End With
14     End With
15 End Function
```

Výpis kódu 26: Makro - Dešifrování Basic64 na UTF-8

Díky tomuto makru, jsem byl schopný dešifrovat veškerá jména i hesla zadávaná hackery ve staženém záznamu.

12 Metoda analýzy dat

Při analýze dat jsem se řídil několika zásadními podmínkami, které mi pomohly roztřídit jednotlivé útoky a rozeznat, jakým způsobem útočník se snaží prolomit do systému bobříka informatiky.

Co tedy analyzovat za jednotlivá data, jsem musel rozdělit do sedmi částí a to následovně:

1. IP adresa, zda je poznat využití VPN, či nikoliv.
2. Jméno a heslo, jestli se u všech konkrétních záznamů opakují, nebo jsou něčím odlišné, zda li se jedná o SQL injekci.
3. Záměr používaných hesel a jmen.
4. Spojitost hesel, jestli je to slovníkový útok jejíž hesla abecedně na sebe navazují, nebo čerpá z libovolného seznamu odcizeních záznamů přihlášeních uživatelů (například: z jiných serverů)
5. Rozsah (velikost) útoků - pomocí parametru $attempting, který souvisí s datem určitého pokusu.
6. Datum i čas, který definuje, zda jsou útoky pravidelné, nebo náhodné.
7. Celkový počet používaných jednotlivých hesel, přihlašovacích jmen.

Na základě předchozích definovaných bodů jsem mohl začít analyzovat data.

13 Nasbíraná data

Podařilo se mi nasbírat již mnoho zajímavých výsledků, ať už z produkčního serveru Bobříka informatiky, nebo testovací serveru, kde se převážně odehrává jeden typ útoku agresivně, tím je slovníkový útok.

Vzhledem k vysokému množství záznamů, ukážu několik zajímavějších záznamů, které postupně rozeberu.

13.1 První záznam - hacker zadávající stejná přihlašovací hesla

Prvním záznamem je takzvaný "Beblerox", který neustále pokouší stejnou kombinaci hesel.

id	id_ip	web	empt	Hack	tenti	ggg	ip	country	date	Decrypted username	Decrypted password
591	591	A	0	1	6	0	5.41.182.6	Saudi Arabia	14.2.22 9:41	itester	Beeblebr0x
592	591	A	0	1	6	0	5.41.182.6	Saudi Arabia	14.2.22 9:41	itester@ibobr.cz	Beeblebr0x
593	591	A	0	1	6	0	5.41.182.6	Saudi Arabia	14.2.22 9:41	admin	Beeblebr0x
594	591	A	0	1	6	0	5.41.182.6	Saudi Arabia	14.2.22 9:41	administrator	Beeblebr0x
595	591	A	0	1	6	0	5.41.182.6	Saudi Arabia	14.2.22 9:41	ibobr	Beeblebr0x
596	591	A	0	1	6	0	5.41.182.6	Saudi Arabia	14.2.22 9:41	Beeblebr0x	Beeblebr0x
599	599	A	0	1	1	0	185.220.101.82	Germany	14.2.22 9:42	admin	Beeblebr0x
601	601	A	0	1	1	0	91.219.237.21	Hungary	14.2.22 9:42	ibobr	Beeblebr0x
606	606	A	0	1	6	0	182.52.56.3	Thailand	14.2.22 11:06	itester	Beeblebr0x
607	606	A	0	1	6	0	182.52.56.3	Thailand	14.2.22 11:06	itester@ibobr.cz	Beeblebr0x
608	606	A	0	1	6	0	182.52.56.3	Thailand	14.2.22 11:06	admin	Beeblebr0x
609	606	A	0	1	6	0	182.52.56.3	Thailand	14.2.22 11:06	administrator	Beeblebr0x
610	606	A	0	1	6	0	182.52.56.3	Thailand	14.2.22 11:06	ibobr	Beeblebr0x
611	606	A	0	1	6	0	182.52.56.3	Thailand	14.2.22 11:06	Beeblebr0x	Beeblebr0x
613	613	A	0	1	1	0	109.70.100.28	Austria	14.2.22 11:07	admin	Beeblebr0x
615	615	A	0	1	1	0	91.219.236.197	Hungary	14.2.22 11:07	ibobr	Beeblebr0x
618	618	A	0	1	6	0	77.137.70.102	Israel	14.2.22 21:07	itester	Beeblebr0x
619	618	A	0	1	6	0	77.137.70.102	Israel	14.2.22 21:07	itester@ibobr.cz	Beeblebr0x
620	618	A	0	1	6	0	77.137.70.102	Israel	14.2.22 21:07	admin	Beeblebr0x
621	618	A	0	1	6	0	77.137.70.102	Israel	14.2.22 21:07	administrator	Beeblebr0x
622	618	A	0	1	6	0	77.137.70.102	Israel	14.2.22 21:07	ibobr	Beeblebr0x
623	618	A	0	1	6	0	77.137.70.102	Israel	14.2.22 21:07	Beeblebr0x	Beeblebr0x
626	626	A	0	1	1	0	195.176.3.20	Switzerland	14.2.22 21:08	admin	Beeblebr0x
628	628	A	0	1	1	0	185.14.97.145	Norway	14.2.22 21:08	ibobr	Beeblebr0x
631	631	A	0	1	5	0	154.160.17.21	Ghana	15.2.22 2:45	itester	Beeblebr0x
632	631	A	0	1	5	0	154.160.17.21	Ghana	15.2.22 2:45	itester@ibobr.cz	Beeblebr0x
633	631	A	0	1	5	0	154.160.17.21	Ghana	15.2.22 2:45	admin	Beeblebr0x
634	631	A	0	1	5	0	154.160.17.21	Ghana	15.2.22 2:45	administrator	Beeblebr0x
635	631	A	0	1	5	0	154.160.17.21	Ghana	15.2.22 2:45	Beeblebr0x	Beeblebr0x
638	638	A	0	1	1	0	51.15.244.188	France	15.2.22 2:45	admin	Beeblebr0x
639	639	A	0	1	1	0	51.83.131.42	Poland	15.2.22 2:46	ibobr	Beeblebr0x
641	641	A	0	1	6	0	114.4.212.80	Indonesia	15.2.22 3:21	itester	Beeblebr0x
642	641	A	0	1	6	0	114.4.212.80	Indonesia	15.2.22 3:21	itester@ibobr.cz	Beeblebr0x
643	641	A	0	1	6	0	114.4.212.80	Indonesia	15.2.22 3:21	admin	Beeblebr0x
644	641	A	0	1	6	0	114.4.212.80	Indonesia	15.2.22 3:21	administrator	Beeblebr0x
645	641	A	0	1	6	0	114.4.212.80	Indonesia	15.2.22 3:21	ibobr	Beeblebr0x

Obrázek 8: První záznam - Beblerox

Jeho útoky jsou mířený na administrátorské rozhraní, datum je nepravidelný, ale co je nejzajímavější, že se jedná o stejného útočníka, i navzdory tomu, že se jeho IP adresa neustále během několika sekund mění. To znamená, že využívá služby VPN. Taktéž, zadává pouze stejná hesla a zkouší různou kombinaci jmen. Pravděpodobně zkouší štěstí, zda existuje nějaký fanoušek filmu "Stopařův průvodce po galaxii" [15], který by si právě dal jako heslo tamního prezidenta. Nicméně, jeho pokusy byli marné, protože se mu nepodařilo proniknout do systému. Kdyby ano, jeho hodnota 'logged' by se rovnala jedné (parametr vlevo hned vedle IP). Jeho kombinace různých jmen, by se dala přirovnat k útoku hrubému silou, ale spíše se jedná o druh software, kde si definoval jen pár základních základních přihlašovacích hodnot, byť těch nejčastějších a pak už jenom spustil sekvenci, se kterou program se pokoušel přihlásit na server.

Celkový počet pokusů je 38, přičemž zaměnil svojí IP adresu 14-krát.

13.2 Druhý záznam - Útok s přihlašovacím jménem domény Bobříka informatiky

Dalším záznam se týká hackera, který nijak nemění svojí IP adresu, časy jsou nepravidelné, interval útoku nespojitý, to znamená, že nemá stejnou sekvenci útoků. Zajímavější je fakt, že do přihlašovacího jména zadává webovou adresu, která se týká Bobříka informatiky.

Počet celkových pokusu tedy měl 58, přičemž jeho země je údajně ze Švédska, nicméně nemůžu tento údaj brát v potaz, kvůli opět možnému krytí pomocí služby VPN.

[15] https://cs.wikipedia.org/wiki/Zafod_B%C3%ADblbrox

id	id_ip	web	empt	Hack	tenti	ogge	ip	country	date	Decrypted username	Decrypted password
955	955	A	0	1	8	0	185.159.156.20	Sweden	3.3.22 2:32	www.ibobr.cz	pass123
958	955	A	0	1	8	0	185.159.156.20	Sweden	3.3.22 4:01	www.ibobr.cz	password123
959	955	A	0	1	8	0	185.159.156.20	Sweden	3.3.22 4:01	www.ibobr.cz	Password
960	955	A	0	1	8	0	185.159.156.20	Sweden	3.3.22 4:01	www.ibobr.cz	pass12345
961	955	A	0	1	8	0	185.159.156.20	Sweden	3.3.22 4:01	www.ibobr.cz	password
962	955	A	0	1	8	0	185.159.156.20	Sweden	3.3.22 4:01	www.ibobr.cz	password1
963	955	A	0	1	8	0	185.159.156.20	Sweden	3.3.22 4:01	www.ibobr.cz	Password1
964	955	A	0	1	8	0	185.159.156.20	Sweden	3.3.22 4:01	www.ibobr.cz	password12345
1029	1029	A	0	1	23	0	185.159.156.20	Sweden	4.3.22 22:30	www.ibobr.cz	111111
1061	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 0:36	www.ibobr.cz	1234567
1062	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 0:36	www.ibobr.cz	12345
1063	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 0:36	www.ibobr.cz	123123
1064	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 0:36	www.ibobr.cz	1234567890
1066	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	0
1067	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	112233
1068	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	123123123
1069	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	121212
1070	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	159753
1071	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	12345678910
1072	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	102030
1073	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	20100728
1074	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	666666
1075	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	123456789
1076	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	654321
1077	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	5201314
1078	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	123
1079	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	11111111
1080	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	1234
1081	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	123321
1082	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	123654
1083	1029	A	0	1	23	0	185.159.156.20	Sweden	5.3.22 2:54	www.ibobr.cz	147258369
1120	1120	A	0	1	22	0	185.159.156.20	Sweden	6.3.22 22:40	www.ibobr.cz	88888888
1121	1120	A	0	1	22	0	185.159.156.20	Sweden	7.3.22 0:52	www.ibobr.cz	6655321
1122	1120	A	0	1	22	0	185.159.156.20	Sweden	7.3.22 0:52	www.ibobr.cz	987654321
1123	1120	A	0	1	22	0	185.159.156.20	Sweden	7.3.22 0:52	www.ibobr.cz	789456123
1124	1120	A	0	1	22	0	185.159.156.20	Sweden	7.3.22 0:52	www.ibobr.cz	686584

Obrázek 9: Druhý záznam - www.ibobr.cz

13.3 Třetí záznam - Nejpoužívanější hesla

Dalším záznamem, je tu údajně z Izraele, útoky nejsou nijak pravidelné, jednoduše neexistuje mezi nimi žádná souvislost, ale čím je tento případ výjimečný, je kombinace různých hesel, že zadává "ibobr2019". Zvládl udělat přes padesát pokusů během 24 hodin a každý útok trval méně než minutu.

1347	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	superman
1348	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	access
1349	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	batman
1350	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	football
1351	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	qazwsx
1352	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	asdasd
1353	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	1qaz2wsx
1354	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	dragon
1355	1320	A	0	1	20	0	185.185.134.116	Israel	14.3.22 16:56	manager	asdfghjkl
1406	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	0
1407	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	12345
1408	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	0
1409	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	Password1
1410	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	1234567890
1411	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	123123
1412	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	11111
1413	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	55555
1414	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	password12345
1415	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	1234567
1416	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	Password
1417	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	12345678
1418	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	27653
1419	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	1234
1420	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	111111
1421	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	987654321
1422	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	Password1
1423	1406	A	0	1	18	0	185.185.134.116	Israel	17.3.22 3:50	manager	11111
1466	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 17:15	anna	0
1469	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 20:02	anna	123321
1470	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 20:02	anna	qwerty
1471	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 20:03	anna	qwe123
1472	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 21:16	anna	pass
1473	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 21:16	anna	12345678
1474	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 21:16	anna	asdasd
1475	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 21:16	anna	102030
1476	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 21:16	anna	password
1477	1466	A	0	1	12	0	185.185.134.116	Israel	18.3.22 21:16	anna	iloveyou

Obrázek 10: Třetí záznam - Útok nejpoužívanějších hesel

13.4 Čtvrtý záznam - Útoků z Ruska

Tento útok je koncentrovaný pravděpodobně z Ruska, trvá přibližně měsíc, kdy opět zkouší sadu hesel a jmen. Jméno ”admin” s heslem ”123456” se zde vyskytuje 3-krát, pravděpodobně generuje hesla z nějakého slovníku a tudíž dochází k jejímu opakování. Nicméně čím je zajímavější, tak začal zadávat do hesla název domény, konkrétně ”ibobr” s kombinací hesel. Jednak už jsem tuto podobu viděl, s tím, že místo domény si zadali název základní školy (v Radomyšli) a k tomu číslo, konkrétnější datum.

id	id_ip	web	empt	Hack	tenti	ogge	ip	country	date	Decrypted username	Decrypted password
1814	1814	A	0	1	1	0	5.188.62.140	Russia	9.4.22 21:45	admin	admin
1815	1815	A	0	1	1	0	5.188.62.140	Russia	9.4.22 23:41	admin	secret
1816	1816	A	0	1	1	0	5.188.62.140	Russia	10.4.22 1:35	admin	123456
1817	1817	A	0	1	1	0	5.188.62.140	Russia	10.4.22 3:32	ibobr	ibobr
1821	1821	A	0	1	1	0	5.188.62.140	Russia	10.4.22 9:28	admin	demo
1823	1823	A	0	1	1	0	5.188.62.140	Russia	10.4.22 11:28	admin	ibobr
1824	1824	A	0	1	1	0	5.188.62.140	Russia	10.4.22 13:32	admin	12345
1825	1825	A	0	1	1	0	5.188.62.140	Russia	10.4.22 15:37	admin	pass
1826	1826	A	0	1	1	0	5.188.62.140	Russia	10.4.22 17:44	admin	admin123
1827	1827	A	0	1	1	0	5.188.62.140	Russia	10.4.22 19:54	admin	1234
1831	1831	A	0	1	1	0	5.188.62.140	Russia	11.4.22 1:06	ibobr	ibobr123
1832	1832	A	0	1	1	0	5.188.62.140	Russia	11.4.22 3:30	admin	12345678
1836	1836	A	0	1	1	0	5.188.62.140	Russia	11.4.22 10:58	ibobr	ibobr@123
1839	1839	A	0	1	1	0	5.188.62.140	Russia	11.4.22 19:57	ibobr	ibobr2020
1842	1842	A	0	1	1	0	5.188.62.140	Russia	12.4.22 5:28	admin	ibobr2020
1853	1853	A	0	1	1	0	5.188.62.140	Russia	12.4.22 9:04	ibobr	ibobr2019
1871	1871	A	0	1	1	0	5.188.62.140	Russia	12.4.22 15:04	ibobr	ibobr@2020
1873	1873	A	0	1	1	0	5.188.62.140	Russia	12.4.22 23:35	admin	admin2020
1938	1938	A	0	1	1	0	5.188.62.140	Russia	26.4.22 14:26	admin	admin
1945	1945	A	0	1	1	0	5.188.62.140	Russia	26.4.22 18:32	admin	secret
1949	1949	A	0	1	1	0	5.188.62.140	Russia	26.4.22 22:47	admin	123456
1950	1950	A	0	1	1	0	5.188.62.140	Russia	27.4.22 3:08	ibobr	ibobr
1956	1956	A	0	1	1	0	5.188.62.140	Russia	27.4.22 16:18	admin	demo
1957	1957	A	0	1	1	0	5.188.62.140	Russia	27.4.22 20:39	admin	ibobr
1959	1959	A	0	1	1	0	5.188.62.140	Russia	28.4.22 1:11	admin	12345
1960	1960	A	0	1	1	0	5.188.62.140	Russia	28.4.22 5:49	admin	pass
1963	1963	A	0	1	1	0	5.188.62.140	Russia	28.4.22 10:25	admin	admin123
1967	1967	A	0	1	1	0	5.188.62.140	Russia	28.4.22 15:09	admin	1234
1969	1969	A	0	1	1	0	5.188.62.140	Russia	29.4.22 3:16	ibobr	ibobr123
1970	1970	A	0	1	1	0	5.188.62.140	Russia	29.4.22 8:57	admin	12345678
1975	1975	A	0	1	1	0	5.188.62.140	Russia	30.4.22 4:53	ibobr	ibobr@123
2040	2040	A	0	1	1	0	5.188.62.140	Russia	7.5.22 4:41	admin	admin
2043	2043	A	0	1	1	0	5.188.62.140	Russia	7.5.22 13:00	admin	secret
2044	2044	A	0	1	1	0	5.188.62.140	Russia	7.5.22 16:58	admin	123456
2070	2070	A	0	1	1	0	5.188.62.140	Russia	11.5.22 11:58	admin	admin
2076	2076	A	0	1	1	0	5.188.62.140	Russia	11.5.22 22:15	admin	123456

Obrázek 11: Čtvrtý záznam - Útok z Ruska

13.5 Bonusový záznam - Slovníkový útok na testovací server

Celkový počet útoku, byť to nebylo součástí plánu, se mi podařilo zaznamenat přes 300 tisíc různých pokusů. Jedná se o stejného člověka, který pravděpodobně využívá bota, poněvadž jeho útoky jsou pravidelné a to v intervalu přibližně šesti hodin. Počet sérií útoků je vždycky 250. Typ útoků jsem definoval jako slovníkový, protože začíná jeho pokus abecedně a to vzestupném pořadí. V poslední části už zkouší jenom kombinaci čísel. Pokus o průnik se odehrává již od 14.1.2022.

id	id_ip	web	atte	isHa	atten	log	IP	country	date	decode u	decode password
5419	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	welshandproud
5420	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	weloveus
5421	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	weloveme
5422	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	weloveit
5423	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	weloveboys
5424	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	wellys
5425	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	wellyboots
5426	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	wellness1
5427	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	wellhithere
5428	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	wellens
5429	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	wellah
5430	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	welkom12
5431	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	welford
5432	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	welcum
5433	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	welcometo
5434	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	welcomee
5435	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	welcomeback
5436	5419	A	0	1	250	0	193.106.31.130	Ukraine	14.01.2022 19:07	admin	welcome6

Obrázek 12: Testovací server - Slovníkový útok část 1.

232752	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	188attkh
232753	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	18894
232754	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	18892
232755	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	1888cfc
232756	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	18855
232757	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	188418
232758	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	1883
232759	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	188111
232760	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:22	admin	18801880
232761	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	188000
232762	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	187rideordie
232763	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	187crip
232764	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	187626
232765	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	187619
232766	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	187600
232767	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	187400
232768	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	1873rfc
232769	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18721872
232770	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	1871994
232771	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	187123
232772	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18693
232773	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	186921
232774	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18681868
232775	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18661866
232776	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18641864
232777	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18639
232778	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18631863
232779	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18621862
232780	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	1861987
232781	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18611861
232782	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	18591
232783	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	185872
232784	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	185421
232785	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	185400
232786	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	1851995
232787	232538	A	0	1	250	0	193.106.31.130	Ukraine	27.04.2022 18:23	admin	1851988

Obrázek 13: Testovací server - Slovníkový útok část 2.

14 Zjištění

14.1 Rozbor hesel jednotlivých záznamů

Zajímavostí prvního záznamu (viz 13.1) je, že "Admin", "administrator", "ibobr", "itester", jsou základní jména, která překvapivě používá řada z nás, leč ibobr je doménové. Kdyby se stránka jmenovala "ikocka", určitě by nepoužíval "ibobr", ale "ikocka", protože zkouší název přímo webové stránky. Na základě těchto pokusů, není bezpečné používat jako přihlášení doménové jméno tvořeného serveru. Dále je zvláštní, že některá jeho jména se opakují a přitom mění VPN adresu, znamená to tedy, že počítá i s tím, že se lze připojit pouze z určitých států, nebo je příliš opatrný.

Dalším záznamem, který je specifický v kombinaci hesel (viz 13.2). Když se podíváte na obrázek 14, tak nejprve začíná typicky heslem jako "password" s různými kombinacemi, ale poté už zkouší jenom různá kombinace číslic, podle toho co zadává, například 1234567, 1023456789, atd., se dá vyvodit, že opět jsou to hesla, která jsou nejvíce používaná - hlavně mezi uživateli a opakují se v mnoha záznamech o pokusů o průnik.

Nejzajímavější hodnotou, která se opakuje mezi záznamy, je nula v hesle. Vzhledem k tomu, že do přihlašování nelze zadávat prázdnou hodnotu, navíc hodnota "0" byla použita celkově 23-krát.

V třetím záznamu (viz 13.3) je impozantní, že, proč zrovna "ibobr2020", "ibobr2019", když se většinou administrátoři webových služeb takového hesla snaží aktualizovat datum na něco aktuálnějšího, nicméně je možné, že někdo si nechává staré datum, byť už je roky starý.

Další záznam (viz. 13.4) je specifický v nejpoužívanějších hesel na světě. Navzdory tomu, že mu došli kombinace hesel ve svém slovníku [16], začíná opět používat nejpoužívanější přihlašovací jméno a tím je "anna", kde opět zadává stejná hesla, akorát s jiným přihlašovacím jménem.

[16]Sada slov, se kterým pracuje jeho program a zadává je přímo do přihlašovacího formuláře

Když se podrobně podíváte na záznam 10 všechno to jsou hesla, nejvíce v dnešní době využívaná, na základě internetových zdrojů [43],[44],[45],[46]. Tím jsou myšlena hesla jako ”dragon”, ”qwe”, ”12345”, ”asdfghjkl”, ”iloveyou”. Dále si můžeme všimnout na obrázku 10, že využívá opět hesla jako ”0”, a proto i takové heslo, je nebezpečné. Nicméně většina webových služeb má při registraci podmínku, že musí zadat minimální počet znaků různých velikostí a čísel.

Posledním záznamem (viz 13.5) je specifický tím, že se jedná o rozsáhlý slovníkový útok na testovací server, kdy začal od písmena Z, nicméně se mi podařilo zaznamenat až od W do číslic, protože ze začátku jsem nevěřil tomu, že by se někdo pokoušel utočit na testovací server a data smazal, nicméně podle referencí [17] má již obsáhlý záznam o pokusu o průnik po celém světě. To znamená, že Bobřík informatiky není jediný server, na který utočí, ale útočí paralelně. Musí k tomu využívat podpůrný software s virtualizací [18], kde naprogramovaný bot, za něj dělá pokusy o přihlášení. Na základě těchto dat, jsem zjistil, že slovníkový útok je pomalý, ale rozsáhlý. Zkouší veškeré různé kombinace hesel a rozhodně je třeba trpělivost při její používání.

[17] https://www.abuseipdb.com/check/193.106.31.130

[18] aby mohl utočit současně na více zdrojů

14.2 Hackeři v záznamech utočí i na cizí servery

Jak jsem již napsal, útok na testovací server Bobříka informatiky, je zaznamenán v několika zemích, nicméně to není jediný hacker, který to zkouší.

Velkým příkladem je útok z Ruska [19] jehož záznamu je přes 3 tisíce a jeho útok se odehrávají po celém světě, taktéž jeho pokusy o proniknutí jsou označovány podle tamních reportů jako útoky hrubou silou, útok na aplikaci webu a spam. Většinou se snaží proniknout do registrovaných portů (od 1024 do 49151), ale samozřejmě i do dynamických. Je vidět, že hacker ví co dělá a na co má utočit, protože si dokáže zjistit jakoukoliv adresu, včetně portu a tvrdě na něj zaútočit.

Druhý záznam, který se vztahuje k nejpoužívanějším hesel (viz. 13.2, kde podle webové stránky [20] má přesně 61 nahlášení za pokus o průnik.

Z toho jde závěru, že výše zmíněný hackeři neutočí jen na Bobříka informatiky, ale i na jiné webové stránky.

14.3 Produkční server

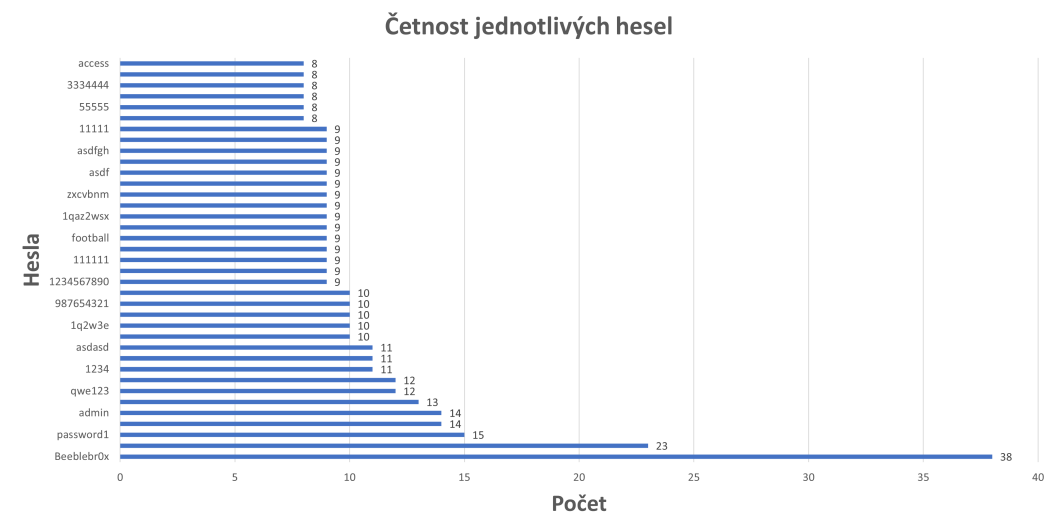
Celkový počet útoků na webové služby bobříka informatiky za období 7.2.2022 do 26.5.2022 bylo zaznamenáno až 817 pokusů o průnik, což by odpovídalo, že se účastnilo 103 hackerů. Nicméně, jak jsme viděli v tabulkách, někteří jsou schopný změnit IP adresu až 15-krát v daném časovém úseku, viz 8.

14.4 Nejpoužívanější hesla

Když vynecháme "Beeblebr0x", poněvadž to výsledek zkresluje, tak nejznámější kombinace jsou v grafu opravdu vyobrazené 14, například: password, admin, qwe123, atd. Z toho lze vyvodit, že se nevyplatí dávat jednoduchá hesla, nejvýhodnější je si vybrat něco originálnějšího a pokud možno i použít českou jazykovou sadu písmen, pokud to aplikace dovolí.

[19] https://www.abuseipdb.com/check/5.188.62.140

[20] https://www.abuseipdb.com/check/185.159.156.20



Obrázek 14: Nejpoužívanější hesla

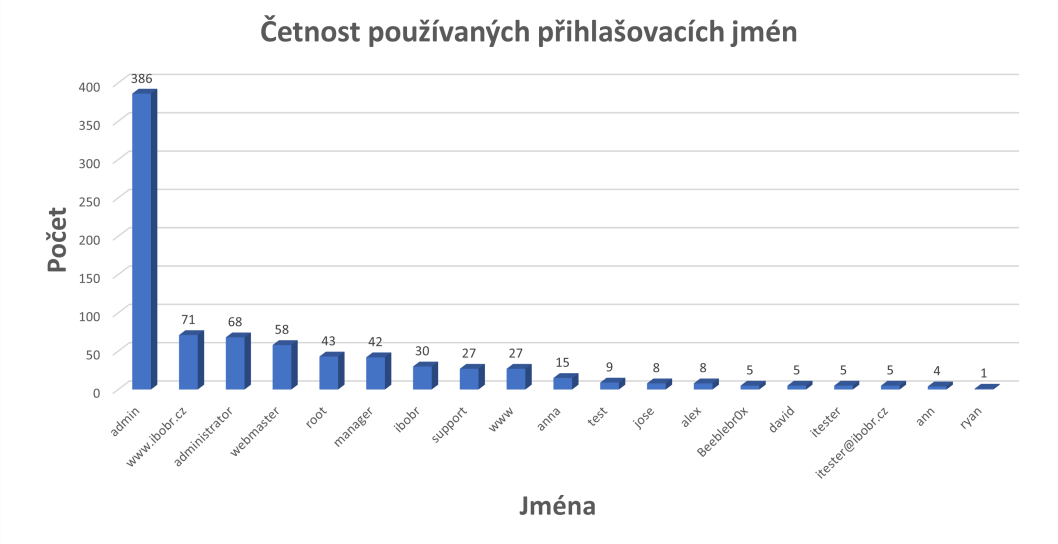
14.5 Nejpoužívanější přihlašovací jména

Podobně jako u nejpoužívanějších hesel 14, bychom si měli dávat pozor na to, aby pro přihlášení na administrátora, nebyl pouze ”admin”, nebo doménový název, jako je v tomto případě ”ibobr”, protože je to první věc, kterou útočník vyzkouší. Zajímavý je fakt, že na druhém místě nejpoužívanějších hesel, je přímo adresa webového serveru. Dále si můžeme na grafu všimnout 15, že je ohromný rozdíl mezi prvním místem a druhým. Takže opravdu doporučuji změnit přihlašovací jméno na něco jiné.

14.6 Nejčastější kombinace hesel a jmen

Pomocí kontingenční tabulky, jsem dále zjistil, že nejčastější používaná jména a hesla jsou:

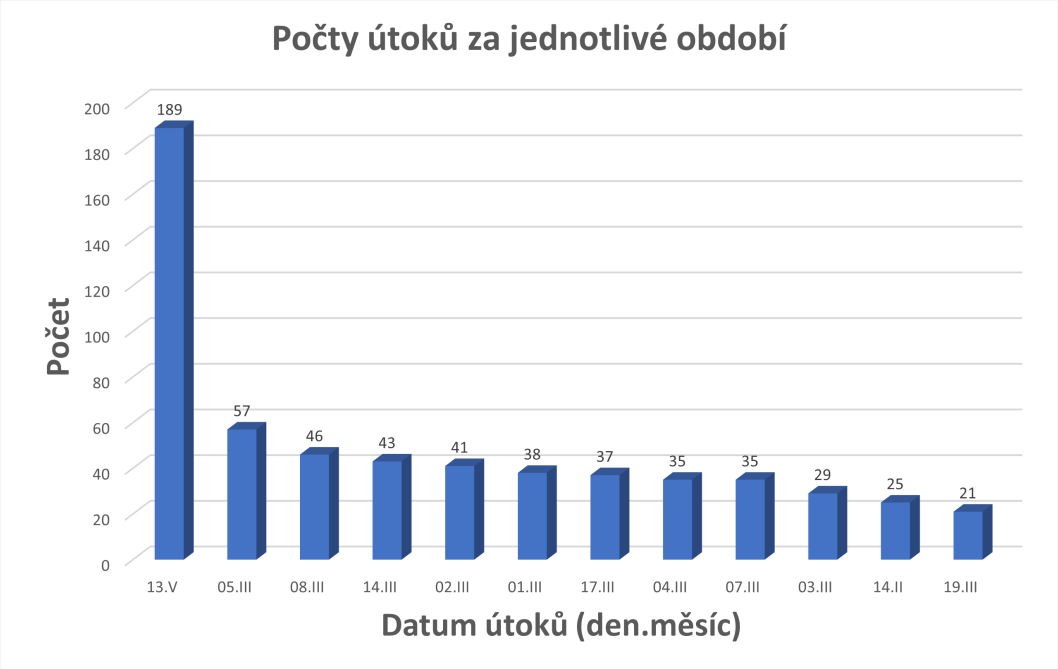
1. ”admin”; heslo ”admin” -> výskyt 14-krát.
2. ”admin”; heslo ”12345” -> výskyt 10-krát.
3. ”admin”; heslo ”0” -> výskyt 9-krát



Obrázek 15: Nejpoužívanější jména

14.7 Nejaktivnější období hackerů

Dále jsem srovnal data, kdy jsem zjistil, ve kterém období hackeři byli nejaktivnější. Na prvním místě je tu pátek třináctého v Květnu. Nicméně jak vidíme na grafu, počet pokusu za ten den byl 189, což je velký rozdíl oproti druhému místu, který má 57 pokusů.



Obrázek 16: Nejaktivnější období pro hackery

14.8 Shrnutí

Všechny hackerské útoky, které se podařilo zaznamenat, míří na administrátorské rozhraní, nikoliv uživatelské. Jejich záměr je pravděpodobně uškodit celému serveru, nikoli pouze jednomu jedinci. Faktem je, že hackeři neutočí jen na jeden server, ale mají jich za cíl více.

Útoky hrubou silou jsou vždy krátký, netrvají déle než tři dny, zatím co slovníkové útoky trvají měsíce.

Všechna hesla (ne)jsou zranitelná, to znamená, že se to dá rozčlenit do čtyřech následující bodů:

- Zda hesla nejsou příliš krátká, obsahují číslice, speciální znaky, nebo velká či malá písmena
- Zda hesla využívá široká veřejnost, například: ”12345”, ”anna”.
- Zda se nacházíte v USA, nebo jinde, protože v České Republice využíváme znakovou sadu UTF-8 a pravděpodobně hackeři z východu nebudou

mít šanci prolomit české heslo.

- Každé slovo, byť sada znaku, které se použilo jako heslo a bylo následně ukradeno z libovolného internetového zdroje, může být součástí sadu slov, jenž vlastní hacker v textovém editoru.

Dalším hlavním faktem je to, že hackeři nepoužívají jména ani hesla v českém jazyce, to znamená, že kombinaci "základníŠkolaŘepa", nepoužijí, i navzdory tomu, že mohli využít nějakou databázi ukradených českých hesel a použít jej. Tedy nesnaží se zadávaná hesla/jména přizpůsobit z hlediska jazykové specificity. Dále z toho jde vyvodit, že většina z nich, útočila ze zahraničí, samozřejmě i se službou VPN.

15 Závěr

Bakalářská práce se zabývala hesly zadávaná hackery při pokusu o průnik do systému, v tomto případě se jednalo o server bobříka informatiky. Tento koncept mě velmi zaujal, nejvíce však výsledky. Nicméně pár věcí mě nepotěšilo, první je tou, že jsem očekával, že těch útoků bude značně více a budou nějakým způsobem agresivnější, z toho důvodu třeba když jsem využil THC-Hydru, jsem zvládl udělat 300 pokusů za sekundu pomocí útoku hrubou silou, protože mi to přihlašovací formulář dovolil, nicméně jsem neudělal o tom žádné záznamy. Další věcí je, že ty útočníci, mi přišli jako neorganizovaný a jejich hesla neměla žádný význam, mimo jiné, že si vyhledali seznam nejpoužívanějších hesel a přihlašovacích jmen.

V Teoretické části jsem se zaměřil nad problematikou, jak taková struktura Joomly vypadá, podrobně jí popsal, aby na základě těchto údajů, bylo možné vytvořit vlastní moduly. Poté jsem popsal druhy útoků, pár těch záznamů bylo podobné útoku hrubou silou, ale jenom jeden z nich zkoušel slovníkový. Samozřejmě obecné terminologie jsem pro připomenutí zmínil též a zabýval se i jak bezpečné heslo má vypadat a sepsal některé způsoby uložení hesel na počítač uživatele, popřípadě do internetových prohlížečů.

V praktické části, jsem upravil celkově dva moduly a třetí vytvořil, jedná se o podrobný výpis všeho, co dané soubory user.php a controller.php obsahují, včetně toho vyvinutého modulu hacker-cleaner.php. Další částí bylo, že jsem rozebral jednotlivé záznamy pokusu o průnik, pomocí kontingenčních tabulek. Navzdory tomu, že ty útoky nebyly agresivnější, jak jsem očekával, se přesto pár zajímavějších našlo.

Na závěr, sepsané kódy by měly být kompatibilní s jakoukoliv webovou platformou podporující redakční systém Joomla. Dále doporučuji omezit počet pokusů na přihlášení, nejlépe tři pokusy každý časový úsek (třeba patnáct minut), nicméně to neplatí pro servery, kde se sbírají data ohledně hackerů. Dle mého názoru, za několik měsíců, se nasbírají mnohem zajímavější data a

věřím, že se s nimi bude moci dát vyvodit lepší ochrana proti hackerům.

Reference

[1] EDDIE, Andrew, Brian TEEMAN, Johan JANSSENS, Marie SIMONET a spol. Joomla.org. Joomla: Joomla! Documentation. Joomla [online]. 2005 [cit. 2022-02-11]. Dostupné z: https://www.joomla.org/

[2] RAHMEL, Dan. Joomla: podrobný průvodce tvorbou a správou webů. Brno: Computer Press, 2010. ISBN 978-80-251-2714-8.

[3] KOUDOUŠKOVÁ, Barbora. Rascasone.com. Co je to API a jaké jsou možnosti jeho využití? [online]. Prosecká 527/24, Libeň, 180 00 Praha, 2021 [cit. 2022-06-27]. Dostupné z: https://www.rascasone.com/cs/blog/co-je-api

[4] EDDIE, Andrew, Brian TEEMAN, Johan JANSSENS, Marie SIMONET a spol. Joomla.org. Retrieving request data using JInput [online]. 2005 [cit. 2022-02-11]. Dostupné z: https://docs.joomla.org/Retrieving_request_data_using_JInput

[5] EDDIE, Andrew, Brian TEEMAN, Johan JANSSENS, Marie SIMONET a spol. Joomla.org. Selecting data using JDatabase [online]. 2005 [cit. 2022-02-11]. Dostupné z: https://docs.joomla.org/Selecting_data_using_JDatabase

[6] KOLOUCH, Jan a Pavel BAŠTA. CyberSecurity. Praha: CZ.NIC, z.s.p.o., 2019. CZ.NIC. ISBN 978-80-88168-31-7.

[7] Kybez. Kybez.cz. Hrozby [online]. GORDIC spol. s r. o, 2021 [cit. 2022-04-09]. Dostupné z: https://www.kybez.cz/

[8] VRANÝ, Boleslav. Socialninauka.cz. Kyberbezpečnost 101: Velmi stručný přehled rizik, útoků i obrany v kyberprostoru [online]. 2020 [cit. 2022-04-09]. Dostupné z: https://www.socialninauka.cz/files/files/Kyberbezpecnost-101.pdf

[9] POŽÁR, Josef. Cybersecurity.cz. Vybrané hrozby informační bezpečnosti organizace [online]. Fakulta bezpečnostního managementu PA ČR v Praze Katedra managementu a informatiky [cit. 2022-04-09]. Dostupné z: https://www.cybersecurity.cz/data/pozar2.pdf

[10] Kybez. Kybez.cz. Před čím chránit? – Bezpečnostní hrozby, události, incidenty. [online]. [cit. 2022-04-09]. Dostupné z: https://www.kybez.cz/bezpecnost/pred-cim-chranit

[11] HALL, Gary a Erin WATSON. Hacking: Computer Hacking, Security Testing, Penetration Testing, and Basic Security. Open Spirit Publishing, LLC (P)2017 Open Spirit Publishing. ISBN 9781541289321.

[12] MINAŘ, Pavel. Jaknait.cz Slovníkový útok. Jaknait [online]. 2020 [cit. 2022-04-09]. Dostupné z: https://www.jaknait.cz/co-je/slovnikovy-utok/

[13] Avast. Avast.com. Sociální inženýrství [online]. 2022 [cit. 2022-04-09]. Dostupné z: https://www.avast.com/cs-cz/c-social-engineering

[14] ESET. Eset.com. DoS útoky [online]. 2022 [cit. 2022-04-09]. Dostupné z: https://help.eset.com/glossary/cs-CZ/dos_attacks.html

[15] Núkib. Nukib.cz. DoS / DDoS útoky [online]. Nároční úřad pro kybernetickou a informační bezpečnost, 2022 [cit. 2022-04-09]. Dostupné z: https://nukib.cz/download/publikace/doporuceni/Doporuceni_DDoS.pdf

[16] HRANICKÝ, Jan. Itnetwork.cz. Technika útoku SQL injection [online]. 2022 [cit. 2022-04-09]. Dostupné z: https://www.itnetwork.cz/php/bezpecnost/technika-utoku-sql-injection

[17] SNYDER, Chris, Tom MYER a Michael G. SOUTHWELL. Pro PHP security: from application security principles to the implementation of XSS defenses. 2nd ed. 2010 ISBN 1430233184.

[18] IQBAL, Muhammad. Researchgate.net. SQL Injection [online]. 2017 [cit. 2022-05-30]. Dostupné z: https://www.researchgate.net/figure/A-SQL-injection-attack_fig3_322250414

[19] Software Testing Help. Softwaretestinghelp.com. 11 Password Cracker Tools (Password Hacking Software 2022) [online]. 2022 [cit. 2022-04-10]. Dostupné z: https://www.softwaretestinghelp.com/password-cracker-tools/

[20] STROUHAL, Lukáš. Hackerské programové nástroje [online]. 2012 [cit. 2022-05-30]. Dostupné z: https://wiki.knihovna.cz/

[21] How CrackStation Works [online]. 2019 [cit. 2022-04-10]. Dostupné z: https://crackstation.net/

[22] Cain And Abel [online]. 2017 [cit. 2022-04-10]. Dostupné z: https://www.darknet.org.uk/

[23] Openwall. openwall.com. John the Ripper password cracker [online]. 2022 [cit. 2022-04-10]. Dostupné z: https://www.openwall.com/john/

[24] Kali. Kali.org. Hydra [online]. 2022 [cit. 2022-04-10]. Dostupné z: https://www.kali.org/tools/hydra/

[25] Lynt. Lynt.cz. Bezpečnost webů [online]. 2022 [cit. 2022-04-10]. Dostupné z: https://lynt.cz/bezpecnost/

[26] LANGEROVÁ, Jana. Podnikatel.cz. Budujte si na internetu pozitivní pověst a chraňte se tak před online útoky [online]. 2019 [cit. 2022-04-10]. Dostupné z: https://www.podnikatel.cz/clanky/budujte-si-na-internetu-pozitivni-povest-a-chrante-se-tak-pred-online-utoky/

[27] LUJKA, Miloslav a Pavel ŘEZNÍČEK. Digitalnipevnost.cz. Hash [online]. 2018 [cit. 2022-05-30]. Dostupné z: https://www.digitalnipevnost.cz/viki/hash

[28] BARÁŠEK, Jan. Php.baraja.cz. Hashování řetězců a hesel [online]. 2019 [cit. 2022-05-30]. Dostupné z: https://php.baraja.cz/hashovani#soleni-hesel

[29] cz.nic. Nebojteseinternetu.cz. Bezpečná hesla [online]. 2022 [cit. 2022-04-10]. Dostupné z: https://www.nebojteseinternetu.cz/page/3448/bezpecna-hesla/

[30] ŠTRÁFELDA, Jan. Strafelda.cz. Cookies [online]. 2022 [cit. 2022-06-26]. Dostupné z: https://www.strafelda.cz/cookies

[31] COATES, Michael a spol. Owasp.org. Secure Cookie Attribute [online]. 2022 [cit. 2022-06-26]. Dostupné z: https://owasp.org/www-community/controls/SecureCookieAttribute

[32] Mozzila. developer.mozilla.org. Using HTTP cookies [online]. 2022 [cit. 2022-06-26]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies

[33] KILIÁN, Karel. Zive.cz. Používáte bezpečná hesla? Google vám je zkontroluje a upozorní na jejich případný únik [online]. 2019 [cit. 2022-06-26]. Dostupné z: https://www.zive.cz/clanky/pouzivate-bezpecna-hesla-google-vam-je-zkontroluje-a-upozorni-na-jejich-pripadny-unik/sc-3-a-200584/default.aspx

[34] Soucet. Support.mozilla.org. Jak Firefox bezpečně ukládá hesla [online]. 2022 [cit. 2022-06-26]. Dostupné z: https://support.mozilla.org/cs/kb/jak-firefox-bezpecne-uklada-hesla

[35] Soucet. Support.mozilla.org. Ochrana uložených přihlašovacích údajů pomocí hlavního hesla [online]. 2022 [cit. 2022-06-26]. Dostupné z: https://support.mozilla.org/cs/kb/ochrana-ulozenych-prihlasovacich-udaju-pomoci-hlav

[36] KLUSKA, Vladislav. Zive.cz. 5 nejlepších služeb pro správu hesel a citlivých údajů [online]. 2018 [cit. 2022-06-26]. Dostupné z: https://www.zive.cz/clanky/5-nejlepsich-sluzeb-pro-spravu-hesel-a-citlivych-udaju/sc-3-a-191643/default.aspx#part=2

[37] Sticky a spol. Stickypassword.com. Neprůstřelná vesta pro vaše hesla [online]. 2022 [cit. 2022-06-26]. Dostupné z: https://www.stickypassword.com/cs/bezpecnost

[38] Igi. Viry.cz. Kam s heslama? [online]. 2020 [cit. 2022-06-26]. Dostupné z: https://viry.cz/kam-s-heslama/

[39] MATĚJÍČEK, Pavel. Spajk.cz. Neukládejte si hesla do prohlížeče [online]. 2020 [cit. 2022-06-26]. Dostupné z: https://spajk.cz/neukladejte-si-hesla-do-prohlizece/

[40] HORDĚJČUK, Vojtěch. Voho.eu. Architektura Klient-Server [online]. 2022 [cit. 2022-06-26]. Dostupné z: http://voho.eu/wiki/klient-server/

[41] JAKUBOVÁ, Veronika. Master.cz. FTP, SFTP, SMB a další protokoly pro přenos souborů: který vybrat? [online]. 2022 [cit. 2022-06-26]. Dostupné z: https://www.master.cz/blog/ftp-sftp-smb-protokoly-pro-prenos-souboru-ktery-vybrat/

[42] BOTH, David. Opensource.com. How I use cron in Linux [online]. 2020 [cit. 2022-06-08]. Dostupné z: https://opensource.com/article/17/11/how-use-cron-linux

[43] MARINO, Michael. Safetydetectives.com. The 20 Most Hacked Passwords in the World: Is Yours Here? [online]. 2022 [cit. 2022-06-10]. Dostupné z: https://www.safetydetectives.com/blog/the-most-hacked-passwords-in-the-world/

[44] MABIRIA, Douglas. Privacysavvy.com. The most hacked passwords in the world (an extensive report) [online]. 2022 [cit. 2022-06-10].

Dostupné z: https://privacysavvy.com/password/guides/most-hacked-passwords-worldwide/

[45] STEEL, Amber. Blog.lastpass.com. The Most Hacked Passwords Around the World [online]. 2022 [cit. 2022-06-10]. Dostupné z: https://blog.lastpass.com/2022/05/the-most-hacked-passwords-around-the-world/

[46] HSU, Jeremy. Spectrum.ieee.org. Ow Language Shapes Password Security Differences between Chinese- and English-language passwords have big security implications for popular Web services [online]. 2019 [cit. 2022-06-10]. Dostupné z: https://spectrum.ieee.org/how-language-shapes-chinese-and-english-password-security

Seznam obrázků

Výpisy kódu

A Příloha

Součástí bakalářské práci je DVD s nasbíranými daty.