Adatbányászat az informatikai biztonságban

Szücs Katalin[1], Holczer Tamás[2], Kiss Attila[3]

[1]szucs.katalin0@gmail.com
ELTE IK
[2]holczer@crysys.hu
Crysys Lab
[3]kissattiladr@gmail.com
ELTE IK

Absztrakt. Az adatbányászat nagy adattömegek érdekes mintáit, ismétlődéseit próbálja feltárni hatékonyan és lehetőleg automatikusan. Eszközeit széles skálán használják például banki és gyógyszerészeti alkalmazásokban, az autógyártásban, a mezőgazdaságban és az oktatásban. Csupán idő kérdése volt, hogy elérje az egyre kritikusabb fontosságú informatikai biztonság területét is, hiszen módszerei ideális eszközt biztosítanak a hálózati és host tevékenységek során felgyülemlő nagy mennyiségű adat hatékony elemzésére, a lehetséges támadások, behatolások detektálására. Az utóbbi időben számos eredmény született, melyek az adatbányászat ilyen célú alkalmazásait vizsgálják. Célunk ezen eredmények összefoglalása.

1. Bevezetés

Az egyre szélesebb körű internethasználat rengeteg új lehetőséget biztosít a felhasználók számára, azonban a társadalom információs rendszerektől való függése egyre több problémával is fenyeget. Ha a rendszerek alapvető működésében zavar lépne fel, az súlyos következményeket okozhatna. Gondoljunk például az információs technológiák kommunikációs vagy gazdasági területeken betöltött szerepére. A felhasználók köre egyre bővül, viszont a hálózat növekedése nem csupán előnyös változásokat hoz. A nagyobb rendszer sérülékenyebb is, hiszen szélesebb támadási felületet biztosít a rosszindulatú felhasználók számára. A legfőbb célok egy ilyen környezetben, hogy a támadásokkal szemben biztosítani tudjuk a szolgáltatások állandó megbízható működését és elérhetőségét, a rendszerben küldött üzenetek integritását és a felhasználók hitelességét. Ezen célok elérését segíti, ha képesek vagyunk a behatolásokat idejében kiszűrni és megfelelő ellenintézkedésekkel reagálni. Ezt a kutatási területet nevezzük behatolás detektálásnak. Módszerei a hálózati biztonság egyik fontos elemét képezik.

A behatolás detektáló rendszer egy olyan szoftver vagy szoftver-hardver kombináció, amely felderít és reagál a fellépő behatolásokra. Alkalmazási környezet alapján megkülönböztetünk hálózat és host alapú rendszereket. Az előbbi a hálózaton áthaladó forgalmat vizsgálja valós időben. Az utóbbi típus egy-egy adott számítógép operációs rendszerének naplóbejegyzéseit vizsgálja és igyekszik a behatolásokat felderíteni. Így, míg az előző típus egy adott hálózaton az összes végpont védelmét egyszerre szolgálja, addig ez utóbbi megoldás minden egyes végponthoz saját behatolás detektálót igényel [1]. Detektálási algoritmus szerint történő osztályozás esetén szabály – és anomália alapú megoldásokról beszélhetünk. A szabály alapú detektáló már ismert támadások mintáit igyekszik fellelni az adatforgalomban. Ilyen minta lehet például egy olyan „Ingyen képek!” tárgyú email, amelyhez az „ingyenkepek.exe” fájlt csatolták, ami egy ismert támadási forma [2]. Az ilyen megközelítés hátránya, hogy az olyan behatolások detektálására nem képes, melyek mintázata még nem szerepel a rendszer adatbázisában. Néhány ismert sza-

bály alapú detektáló rendszer a Snort, a Suricata és a Sagan [3, 4]. Az anomália alapú detektálás elve fordítottan működik. Itt a támadó mintázat keresése helyett a normális adatforgalom, például a szokásos sávszélesség használat, a jellemzően használt protokollok és portok meghatározása a cél és minden ettől eltérő esemény behatolásnak minősül. Ilyen anomáliák észlelése esetén a rendszer értesíti a felhasználót vagy a rendszergazdát. Ennek a megközelítésnek a legnagyobb előnye az, hogy a szabály alapú rendszerektől eltérően a korábban nem ismert támadások detektálására is képesek. Azonban ez a megoldás sem tökéletes, jellemzően túl gyakran ad téves riasztást, ami igencsak megnehezíti a használatukat.

A behatolás detektáló rendszerek területén érdemes adatbányászati eszközöket alkalmazni. Ennek oka, hogy mind a hálózati, mind a host tevékenységek során felgyülemlő nagy mennyiségű adat elemzésére megfelelő eszközöket biztosítanak. Az adatbányászat definíció szerint: újszerű, érvényes, nem triviális, vélhetően hasznos és magyarázható összefüggések keresése nagy adathalmazban [5]. Fő területei: előrejelzés, osztályozás, regresszió, klaszterezés, idősor elemzés, gráf mintázatok keresése és asszociációs szabályok keresése. Ezek az eszközök, vagy különböző kombinációik, sikerrel alkalmazhatók a behatolás detektálás különböző lépéseiben. Néhány speciális cél, melyek elérését lehetővé teszik:

- a riasztást generáló adatok közül a normális forgalom kiválasztása, ami lehetővé teszi az elemzők számára, hogy csak a valódi támadásokra koncentráljanak;
- a szabályostól eltérő adatforgalom kiszűrése;
- a normális adatforgalom körülhatárolása;
- a téves riasztások okának felderítése;
- az automatizálás elősegítése;
- valós idejű elemzések végzése.

2. Adatbányászati eszközök a behatolás detektálás területén

A szakirodalomban számos kutatással találkozhatunk, amelyek adatbányászati eszközöket alkalmaznak a behatolás detektálás során. Ebben a fejezetben ezekből az eredményekből mutatunk be néhányat.

2.1. MADAM ID

MADAM ID-t (Mining Audit Data for Automated Intrusion Detection) 1999-ben a Columbia Egyetemen fejlesztették ki, mára széles körben ismert és kutatott rendszer. Az egyik legelső olyan behatolás detektáló, amely adatbányászati eszközöket alkalmaz. Célja a hálózati behatolások szignatúra alapú észlelése. Működése során nem vizsgálja a hálózati csomagok tartalmi részét, a forgalmat csupán absztrakt adatrekordokként kezeli, melyek az üzenetek alapvető adatait tartalmazzák. Ilyen adat például a forrásgép és célgép IP címe, port száma, az üzenet hossza, küldésének ideje. MADAM ID a detektálás megkezdése előtt a rendszer modellépítéséhez címkézett tanuló adatokat igényel. Működése két lépésből áll. Első lépésben az adatrekordokat olyan elemekkel egészíti ki, amelyek relevánsak lehetnek a normális és a támadó adatforgalom elkülönítésére. Egészen pontosan a következőket hajtja végre:

1. A tanuló adatokat normális és támadó csoportokra osztja.
2. Asszociációs szabályokat és gyakori elemeket keres külön a két halmazban, majd az eredményeket összehasonlítja a normális és a támadó adatok között. Így elkészül a támadási

mintázatokat tartalmazó halmaz. Ebben kizárólak azok a szabályok és gyakori elemek találhatóak meg, amik a támadó adathalmazban szerepeltek, de a normálisban nem.

3. A támadási mintázatokhoz új attribútumokat kapcsol, amelyek fontosak lehetnek a detektálás során. Ezek az attribútumok olyan kapcsolatok számát, átlagát, arányát jelzik, amelyek néhány adatrekordban megegyeznek az aktuális kapcsolattal, amihez a támadási minta tartozik.

Ezután a második lépés a következőkből áll:

1. Egyesíti a normális és az új attribútumokkal kiegészített tanuló adatokat (a normális és támadó címkék megtartásával).
2. Betanítja az osztályozó algoritmust az így előkészített adatokon.

Fontos megjegyezni, hogy az eljárás több ponton támaszkodik a felhasználó szakértelmére. Ilyen pontok például az asszociációs szabályok és gyakori elemek keresése, a detektálás szempontjából ígéretes attribútumok kiválasztása. Továbbá a különböző támadásosztályok különböző osztályozó betanítását teszik szükségessé, ezek kiválasztása is szakértelmet igényel [6, 7].

2.2. ADAM

ADAM (Audit Data Analysis and Mining) egy széles körben használt és kutatott anomália - és hálózat alapú behatolás detektáló rendszer. A MADAM ID-hoz hasonlóan ez a rendszer sem vizsgálja az adatcsomagok tartalmi részét, kizárólag a TCP kapcsolatokra vonatkozó információkat használja. Az asszociációs szabály keresés és az osztályozás elemeit használja a behatolások detektálásához a hálózati forgalom adataiban. Az eljárás működése két lépésre bontható:

1. lépés: egy garantáltan támadásmentes adathalmazból felépíti a normális forgalom gyakori elemeinek tárát, melyben a gyakori forgalmi eseményeket asszociációs szabályok reprezentálják.
2. lépés: futási időben egy csúszó ablak használatával kigyűjti az elmúlt δ másodperc TCP kapcsolataihoz tartozó asszociációs szabályokat, majd ezeket összehasonlítja a normális fogalomról tárolt adatokkal. Így kiszűri a TCP kapcsolatok közül azokat, amelyek a normálistól eltérő mintát mutatnak. Ezt követően egy osztályozó algoritmus segítségével a következő csoportokba sorolja őket: ismert támadás, nem ismert támadás, téves detektálás.

Megfigyelhető, hogy az asszociációs szabály keresés gyakran vezet irreleváns vagy redundáns szabályok vizsgálatához. A probléma enyhítésének érdekében ADAM csak azokat az asszociációs szabályokat fogadja el, amelynek attribútumai között szerepel a forrás gép IP címe, valamint a célgép IP vagy port száma [7, 8].

2.3. COMPA

A COMPA rendszert a közösségi hálózatokkal való visszaélések detektálására hozták létre. Módszere a statisztikai modellezés és az anomália detektálás kombinációját alkalmazza [9].

A közösségi portálok felhasználói fiókjaival szemben általában kétféleképpen lehet visszaélni. Vannak kifejezetten a támadás céljából létrehozott, úgynevezett Sybil fiókok. Ezek viselkedése az átlagostól feltűnően eltérő, ezért viszonylag könnyű őket detektálni. Az olyan támadások viszont, amikor a behatoló egy már létező felhasználó fiókját kompromittálja, sokkal hatékonyabbak, hiszen a támadó kihasználhatja az áldozat meglévő bizalmi rendszerét. Az okozott kár helyreállítása is sokkal nehezebb, hiszen a rendszergazdáknak nem elég a fiókot egyszerűen törölni. COMPA-t kifejezetten az ilyen típusú behatolások detektálására fejlesztették ki. Bár korábban is készültek a problémát célzó megoldások, azok tipikusan vagy csupán az URL-t tartalmazó rosszindulatú üzenetek kiszűrésére, vagy a Sybil fiókok detektálására voltak csak alkalmasak.

COMPA megoldása több ponton eltér az eddigi eredményektől. Először is nem kizárólag a küldött üzenetekre, hanem a felhasználók viselkedésére koncentrál. Másodszor pedig különbséget tesz a támadó által létrehozott és a kompromittált fiókok között. Csak az utóbbiak detektálása a célja. Működése dióhéjban két lépésből áll:

1. Hasonló üzenetek csoportosítása.
2. A felhasználók viselkedésében fellépő számottevő változás észlelése.

A két lépés tetszőleges sorrendben végrehajtható. Bár a támadók sok hasonló szöveggel vagy URL-el rendelkező üzenetet küldenek, a hasonlóság vizsgálata önmagában nem elég. Gondoljunk például a „Boldog születésnapot!" üzenetekre. Másrészt, mivel viselkedésbeli változások a normális felhasználók körében is megfigyelhetőek, így önmagában ennek vizsgálata sem elegendő.

A hasonló üzenetek csoportosítására két hasonlósági mértéket használnak. A tartalmi hasonlóság észleléséhez 4-szó-gram alapú vizsgálatot alkalmaznak. Az URL alapú megközelítés szerint pedig két üzenet akkor számít hasonlónak, ha tartalmaznak közös URL-t.

A viselkedési profilok elkészítéséhez minden felhasználóhoz kigyűjtik az általuk közzétett üzeneteket kronológiai sorrendben. Ha valamely lista tíznél kevesebb üzenetet tartalmaz, azt elvetik. Ezután minden üzenetből kivonják a következő hét tulajdonsághoz tartozó információt:

- Idő: a nap azon órája, amelyben az üzenetet küldték.
- Üzenet forrása: az alkalmazás neve, amin keresztül a küldés történt.
- Üzenet nyelve
- Linkek: az üzenetben szereplő URL címek domain része.
- Közvetlen interakció más felhasználókkal: azon felhasználók listája, akikkel az üzenet küldője valaha közvetlen kapcsolatot teremtett (például üzenetet tett közzé a falán, vagy megemlített egy hozzászólásban).
- Üzenet témája: #-el megjelölt téma címkék.
- Szomszédosság: néhány közösségi portál a földrajzi elhelyezkedésnek megfelelő hálózatokba sorolja a felhasználóit. Ez a tulajdonság a hálózaton belül és a hálózaton kívül küldött üzenetek arányát tartja számon.

Ezek a tulajdonságok statisztikai modellek felépítéséhez szükséges. Tulajdonságonként egy modell épít a rendszer. Minden modellt egy M halmaz reprezentál, amelynek elemei az <fv, c> rendezett párok, ahol fv a tulajdonság értéke, c pedig az fv érték előfordulásainak száma az eddig vizsgált üzenetekben. Továbbá N jelölje az eddig vizsgált üzenetek számát. A modelleket két osztályba sorolhatjuk. Vannak kötelező modellek, ahol bármely üzenetben a modellhez tartozó tulajdonság pontosan egy értéket vesz fel. Például: idő, forrás, szomszédság, nyelv. Az opcionális modellekben viszont előfordulhat, hogy a tulajdonsághoz egynél több érték tartozik egy üzenetben, vagy nem tartozik hozzá érték. Ilyen például: link, közvetlen interakció, téma. Az opcionális modellekhez tartozik egy speciális null elem. Ez jelöli, hogy az eddig vizsgált üzenetek közül hányban nem tartozik érték az adott tulajdonsághoz. Az idő érték meghatározása egy kicsit eltér a többitől a modellalkotás során. Először itt is elkészülnek az <fv, c> párok. Ezután M minden <i, ci> elemében ci-t ci'-re cseréljük, ahol ci' a következő három szám átlaga: az i. órában küldött üzenetek száma, az (i-1). órában küldött üzenetek száma, az (i+1). órában küldött üzenetek száma.

A modellek megalkotása után minden új üzenet kiértékelésre kerül, melynek eredménye egy anomália pontszám meghatározása 0 és 1 között. A pontszám megmutatja, hogy az új üzenet mennyire illeszkedik a felhasználó viselkedési profiljába. Minél nagyobb ez a pontszám, az üzenet annál kevésbé illeszkedik a profilba. Az anomália pontszám meghatározása a kötelező modellek esetén három lépésből áll:

1. Meghatározzuk az új üzenetb fv értékét. Ha az M halmazban még nem szerepel olyan elem, aminek az első értéke fv, a folyamat itt megáll és az anomália pontszám értéke 1.
2. Ha M-ben szerepel az <fv, c> pár, akkor legyen M' M-ben a második koordináták összege osztva M elemeinek számával. Ha c >= M', az új üzenet illeszkedik a viselkedési profilba, az anomália pontszám értéke 0.
3. Ha c < M', legyen f = c/N. Az anomális pontszám pedig 1 –f.

Az opcionális modellek esetén az eljárás a következő:

1. Az új üzenetből meghatározzuk fv értékét. Ha M már tartalmaz olyan párt, amelynek első értéke fv, az anomália pontszám 0.
2. Ha még nincs ilyen pár, az anomália pontszám p, ahol p annak a valószínűsége, hogy a felhasználó üzenetéből hiányzik a modellhez tartozó tulajdonság. Legyen a tulajdonsághoz tartozó nullelem c_null. Ekkor az anomália pontszám értéke p = c_null / N. Ha M-ben nincs null értéket tartalmazó pár, akkor legyen c_null = 0.

A különböző modellek kiértékelése után egy végső anomália pontszám kerül meghatározásra. A végső pontszám a modellek pontszámainak súlyozott összegeként számítódik ki. Egy üzenet akkor sérti meg a felhasználó viselkedési profilját, ha a végső anomália pontszám meghalad egy bizonyos határértéket.

Összegezve, a kompromittált fiókok detektálási folyamata a következő módon zajlik. A megfigyelési intervallum során érkező üzenetek először csoportosításra kerülnek a hasonlósági mértékek alapján. Minden csoportban a rendszer felépíti a felhasználók viselkedési profilját és megvizsgálja, van-e olyan üzenet, ami megsérti a viselkedési profilt. A gyanús csoportok azok a csoportok, amelyekben a viselkedési profilt sértő üzenetek aránya meghalad egy adott th korlátot, ami a csoport elemszámától függően számítódik ki. Ha a csoport elemeinek száma n, akkor ez a korlát:

$$th(n) = \max(0.1, kn + d), \quad \textbf{(1)}$$

ahol k= 0.005, d= 0.82. A gyanús csoportokhoz tartozó minden felhasználói fiókot kompromittáltként jelezi a rendszer.

Néhány népszerű alkalmazás sok hasonló üzenetet küld a felhasználóinak, ami hamis detektálást eredményezhet. COMPA külön kezeli ezeket az alkalmazásokat, így a népszerű alkalmazások üzenetei nem sértik a viselkedési profilokat.

A rendszert Facebook és Twitter adatokon tesztelték. A Twitter adatokat 2011. május 13. és 2011. augusztus 12. között gyűjtötték be, a nyilvános adatok 10%-át tartalmazza (kb. 15 millió tweet naponta). A Facebook adatok 2007. szeptember és 2009 július közöttiek, összesen 106 millió poszt. A teszt eredményeit az 1-3. táblázat tartalmazza (a hiányzó adatok az eredeti cikkben sem voltak megtalálhatóak).

Értékelés Twitter adatokon

	Csoportok száma	Fiókok száma
Összesen	374 920	-
Kompromittált	9 362	343 729
Fals pozitív	4% (377)	3,6% (12 382)

1. táblázat: COMPA teljesítménye a Twitter adatokon a szöveg hasonlósági mérték alapján.

Értékelés Twitter adatokon

	Csoportok száma	Fiókok száma
Összesen	14 548	-
Kompromittált	1 236	54 907
Fals pozitív	5,8% (72)	3,8% (2 141)

2. táblázat: COMPA teljesítménye a Twitter adatokon az URL hasonlósági mérték alapján.

Értékelés Facebook adatokon

	Csoportok száma	Fiókok száma
Összesen	14 548	-
Kompromittált	1 236	54 907
Fals pozitív	5,8% (72)	3,8% (2 141)

3. táblázat: COMPA teljesítménye a Twitter adatokon az URL hasonlósági mérték alapján.

3. Saját tapasztalati eredmények

Munkánk célja egy olyan anomália alapú behatolás detektáló rendszer tervezése és implementálása volt, amely adatbányászati eszközök felhasználásával jelzi az esetleges támadásokat a hálózati forgalomban. A tervezés során a két legfontosabb szempont a még nem ismert támadások detektálása és a fals pozitív riasztások minimalizálása volt.

3.1. Koncepció

Alkalmazási környezetként a folyamatirányítási rendszerek területét választottuk. Az ötlet lényege, hogy míg nyílt hálózati környezetben (például egy kávézó vezeték nélküli internet hálózata) a támadásmentes adatforgalom tetszőlegesen változatos lehet, addig más, speciális környezetekben a forgalmi adatok szigorúbb, előre könnyebben megjósolható mintát követnek. Jellemzően ilyen típusúak a folyamatirányítási rendszerek (például SCADA [10]), amelyek feladata más eszközök vagy rendszerek működésének összehangolása, felügyelete, irányítása, szenzor adatok gyűjtése, kiértékelése. Sok olyan létezik köztük, melyek működésének megfelelő biztosítása kritikus, hiszen téves működés esetén helyreállíthatatlan sérülést okozhatnak az irányítás alatt álló fizikai rendszerben, esetleg az azoktól függő embereknek. Ilyen rendszerek megtalálhatóan az

orvoslásban, egészségügyi eszközök működésének részeként, de alkalmazzák a közüzemi művek, a közlekedésirányítás területén, gyárakban, kommunikációs rendszerekben is.

Behatolás detektálás szempontjából a legnagyobb előnyük az, hogy előre meghatározott, mely eszközök milyen időközönként kommunikálhatnak egymással, valamint a küldött üzenetek értékei is ismert határokon belül mozoghatnak. Ezek a körülmények lehetővé teszik az anomália alapú detektálás alkalmazását. Így munkánk első lépésében megfogalmazhattuk azt a feltevést, hogy egy ilyen környezetben a támadó forgalom a normális adatforgalomtól minőségben különbözik, így attól elkülöníthető. Továbbá feltettük, hogy a normális adatforgalom mennyisége lényegesen több, mint a támadó.

Megoldásunk koncepciója vázlatosan a következő:

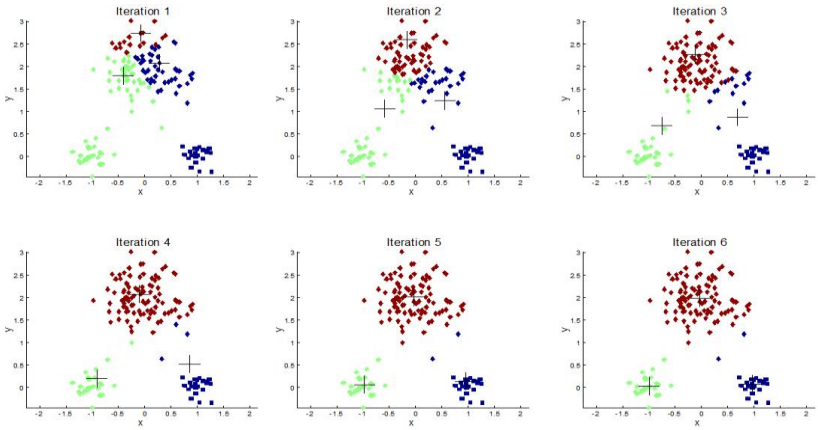
- Első lépésben a forgalmi adatokból vektorteret építünk. A forgalmi elemekhez tartozó vektorok koordinátái a normális adatforgalmat jellemző tulajdonságokat reprezentálják.
- Ezt követően egy klaszterező algoritmus segítségével csoportosítjuk a vektortér elemeit.
- A klasztereket méretük alapján támadó és normális osztályokba soroljuk. A klaszterekben szereplő forgalmi elemekről a klaszter besorolása alapján döntjük el, hogy a forgalmi minta normális vagy támadó jellegű.

3.2. A klaszterező algoritmus

A vektortér elemeinek csoportosítására az Y – közép algoritmust alkalmaztuk [11]. Az eljárás alapja a széles körben használt K – közép algoritmus [13], melynek célja n darab megfigyelés k klaszterbe való sorolása olyan módon, hogy minden megfigyelés a hozzá legközelebb eső klaszterközéppontú klaszterben kapjon helyet. A lépései, melyeket az 1. ábra[1] példaadatsoron mutat be, a következők:

1. lépés: kiválasztunk k darab megfigyelést véletlenszerűen. Ezek alkotják kezdetben a klaszterközéppontokat.
2. lépés: minden megfigyelést a hozzá legközelebb eső klaszterközépponthoz rendelünk.
3. lépés: helyettesítsünk minden klaszterközéppontot a klaszterhez tartozó elemek átlagával.
4. lépés: ismételjük a 2-es és a 3-as lépést. Ha a lépések elvégzése után nincs több változás, az algoritmus megáll.

1 Kép forrása: https://apandre.wordpress.com/visible-data/cluster-analysis/



1. ábra: K - közép algoritmus működése példa adatsoron.

Az algoritmust különböző területeken évtizedekig sikeresen alkalmazták. Futási ideje hatékony, O(nmt), ahol n a megfigyelések, m a klaszterek és t az iterációk számát jelöli. Mindezek ellenére behatolás detektálás szempontjából két hátránnyal rendelkezik. Az egyik, hogy előre meg kell határoznunk a klaszterek számát, a másik, hogy előfordulhat, hogy az algoritmus üres klasztereket is készít. Behatolás detektálás során a kialakítandó klaszterek számának előre meghatározása problémát jelent, hiszem ez a szám az aktuális hálózati forgalomtól függően változik. Az üres klaszterek kezelése is okozhat fennakadást a normális és támadó klasztercsoportok kialakítása közben.

Az Y – közép algoritmus erre a két problémára szolgáltat hatékony megoldást. Bár itt is meg kell adnunk az algoritmus számára a kezdő klaszterek számát, a végső klaszterek mennyiségét az eljárás automatikusan határozza meg. Ezt az egymáshoz közeli klaszterek összeolvasztásával és a külső pontokat tartalmazó klaszterek kettéválasztásával éri el.

Lépései a következők:

1. lépés: n darab elem esetén először véletlenszerűen kiválasztunk k darab klaszterközéppontot a megfigyelések közül, ahol k egy 1 és n közötti egész szám.
2. lépés: minden elemet a hozzá legközelebb eső középponthoz rendelünk. A klaszterközéppontokat az így kapott klaszterek elemeinek átlagával helyettesítjük.
3. lépés: az üres klasztereket új klaszterekkel helyettesítjük.
4. lépés: ha valamely D klaszter bármely elemére igaz, hogy a klaszterközépponttól való távolsága nagyobb, mint $t_D = c*d$, ahol c paraméter, d pedig a klaszter elemeinek szórása, az elemet eltávolítjuk a klaszterből és új középpontnak jelöljük. Újra elvégezzük a 2. lépést.
5. lépés: ha létezik C, D két olyan klaszter, melyek középpontjainak távolsága kisebb, mint $t_C + t_D$, vagyis a klaszterekhez tartozó határparaméterek összege, a két klasztert összeolvasztjuk. Az új középpont a két klaszter középpontjának átlaga. Újra elvégezzük a 2. lépést.
6. lépés: ismételjük a 3-5. lépéseket addig, amíg van változás.

3.3. Teszt adatok

A koncepció elkészítése után megoldásunkat valós adatokon teszteltük. A tesztelést az iCTF 2013 verseny [14] két szerverének adatforgalmán végeztük, ami összesen 476 TCP kapcsolatot tartalmaz. A verseny során különböző hacker csapatok támadják egymás szervereit, amelyeket kifejezetten a verseny céljából hozták létre. Első ránézésre minden szolgáltatás szokványosnak tűnik, de bővebb elemzés után kiderül, hogy mind rejtett sérülékeny pontokat tartalmaz. A versenyzők célja saját szolgáltatásuk fenntartása és a sérülékeny pontok kiaknázása a többi csapat szerverén.

A szervereken futó szolgáltatások egyszerűségéből adódott, hogy az általuk generált forgalmi minták a normális működés során kevés változatosságot mutattak. Így megfelelő környezetet biztosítottak az anomália alapú behatolás detektálás kiépítéséhez.

3.3.1. Példa szerver

Összesen tíz darab szolgáltatást használtak a játék lebonyolítása alatt. Ezek közül az egyik DexWare, melynek feladata titkos vegyi formulák tárolása és kezelése volt. A versenyzők feladata az, hogy megszerezzék ezeket a jelszóval védett formulákat a jelszó ismerete nélkül. A kapcsolat létrejöttével a szerver a 2. ábrán látható hat darab lehetőséget ajánlja fel a látogató számára. A többi szolgáltatás ehhez hasonlóan működött.

```
Welcome to the uber-secret formula management system.
Select an option:
1) List formulas
2) Add formula
3) Get formula
4) Upgrade
5) About
6) Quit
```

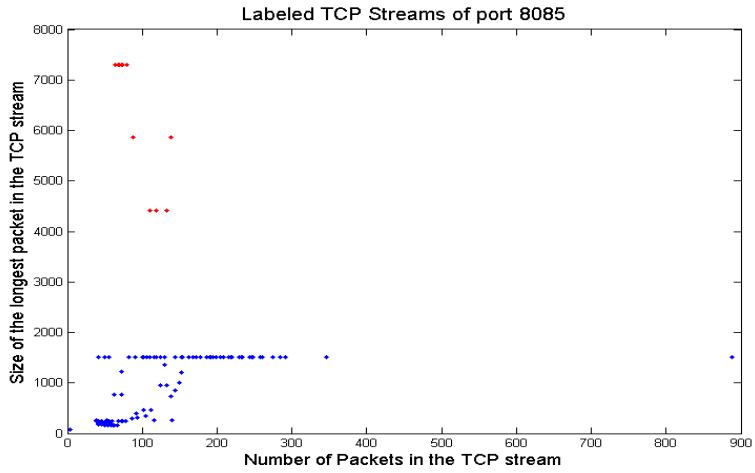
2. ábra: a DexWare adatbázisában végrehajtható műveletek

A szerver sérülékeny pontjára akkor derült fény, ha a versenyzők a hat lehetséges opció helyett a 42. opciót választották. Ekkor képesek voltak bármilyen fájlt feltölteni a szerverre, majd végrehajtani a fájlban található bármilyen kódot. Ilyen támadás esetén a csomag mérete, amely a feltöltendő fájlt tartalmazza jelentősen nagyobb, mint a normális forgalomé.

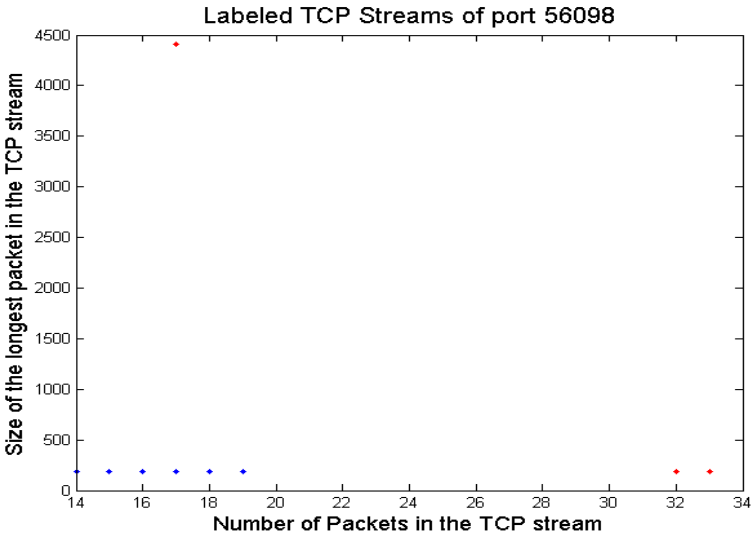
3.3. Eredmények

Hogy alkalmazhassuk a klaszterező algoritmust, a forgalmi adatokból egy két-dimenziós vektorteret építettünk. Minden TCP kapcsolatnak egy vektort feleltettünk meg, amelynek koordinátái a kapcsolatban szereplő legnagyobb csomag hossza és a csomagok száma. A 3. ábra mutatja a DexWare szerver adataihoz tartozó vektorokat a térben, míg a 4. ábra a Temperature szerverhez tartozó adatokat tartalmazza.

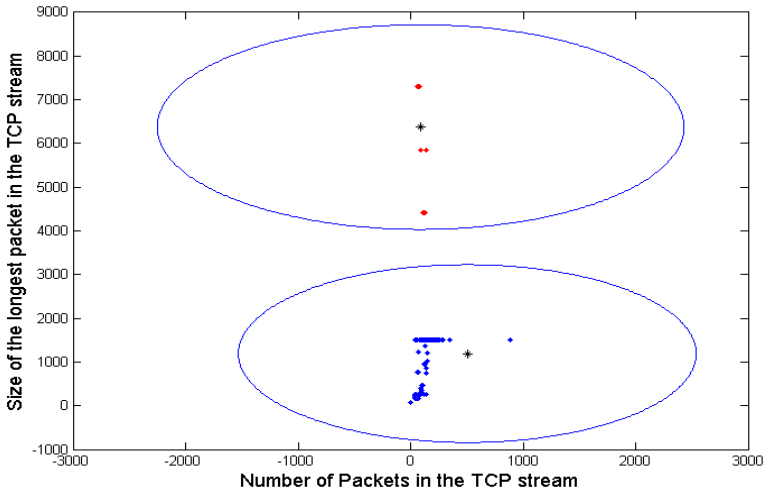
A vektortér elkészítése után az Y – közép algoritmust Matlab függvényként implementáltuk, mely bemeneti paraméterként várja az induló klaszterek számát és a klaszterek határparamétereinek konstansát. Az algoritmust a szerverekhez tartozó normált vektorokon futtattuk. A 5. ábra a DexWare, míg a 6. ábra a Temperature szerver adatainak klaszterezését mutatja. Jól látható, hogy mindkét esetben a támadó és a normális adatforgalom külön klaszterekbe kerültek.



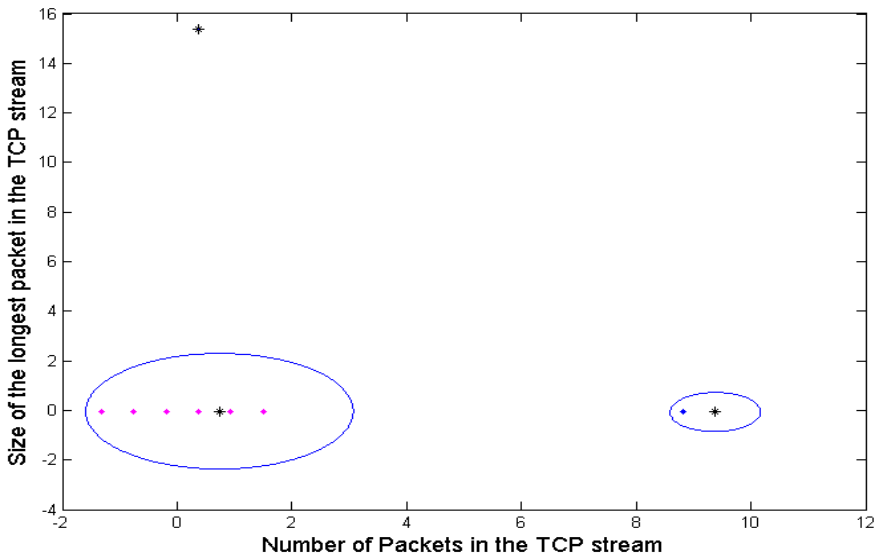
3. ábra: A DaxWare szerverhez tartozó TCP kapcsolatok a vektortérben. A normális forgalom (kék) az ábra bal alsó részén egy téglalapnyi területen tömörül össze, míg a támadó forgalom (piros) ezektől jól láthatóan elkülönül.



4. ábra: A Temperature szerver TCP kapcsolataihoz tartozó vektorok. Bár a képen 238 vektort ábrázoltunk, a normális adatforgalomhoz (kék) tartozó vektorok csupán hat ponton koncentrálódnak. A támadó forgalom (piros) ezektől jól láthatóan elkülönül



5.ábra: Y – közép algoritmus alkalmazása a DexWare szerverhez tartozó normált vektorokon. A kék ellipszisek a klaszterek határait, míg a fekete csillagok a klaszterközéppontokat jelzik. A nagyobb klaszter kizárólag a normális forgalomhoz tartozó vektorokat tartalmazza, míg a kisebben csak támadások találhatóak.



6. ábra: A Teperature szerverhez tartozó vektorok normálás és klaszterezés után. A kék ellipszisek a klaszterek határait, míg a fekete csillagok a klaszterközéppontokat jelzik. A nagyobb klaszter kizárólag a normális forgalomhoz tartozó vektorokat tartalmazza, míg a két kisebben csak támadások találhatóak.

4. Összefoglalás

Munkánk során az adatbányászati módszerek behatolás detektáló rendszerekkel való integrálhatóságát vizsgáltuk. Először néhány példán keresztül tekintettük át az adatbányászat ilyen irányú alkalmazásait, majd összefoglaltuk saját tapasztalati eredményeinket. Az adatbányászat eszközeinek alkalmazása az informatikai biztonságban továbbra is aktív kutatási terület.

Irodalom

1. http://www.synergon.hu/term_szolg/bizt_mo/ved_int/bahatolas.html
2. Karen Scarfone and Peter Mell: *Guide to Intrusion Detection and Prevention Systems*, NIST Special Publication, 2007
3. http://www.aldeid.com/wiki/Suricata-vs-snort
4. Joshua S. White, Thomas T. Fitzsimmons, Jeanna N. Matthews Quantitative Analysis of Intrusion Detection Systems: Snort and Suricata : Publication,Wallace H. Coulter School of Engineering Department of Computer Science, Clarkson University, Potsdam, NY USA, 2013
5. http://hu.wikipedia.org/wiki/Adatb%C3%A1ny%C3%A1szat
6. Aleksandar Lazarević, Jaideep Srivastava, Vipin Kumar: *Data Mining for Intrusion Detection*, Army High Performance Computing Research Center Department of Computer Science University of Minneso, 2003
7. Daniel Barbará, Sushil Jajodia: *Applications of Data Mining in Computer Security.* Kluwer Academic Publishers, 2002
8. ADAM: http://cs.gmu.edu/~dbarbara/adam.html
9. Manuel Egele, GianlucaStringhini, ChristopherKruegel, and Giovanni Vigna: *COMPA: Detecting Compromised Accounts on Social Networks, NDSS, 2013*
10. *Wei Gao, Thomas Morris, Bradley Reaves, and Drew Richey: On SCADA Control System Command and Response Injection and Intrusion Detection, Mississippi State University, 2010*
11. Guan Y., Ghorbani Ali-Akbar, Belacel Nabil: *Y-Means: A Clustering Method for Intrusion Detection*, published in the Canadian Conference on Electrical and Computer Engineering, 2003
12. Leonid Portnoy, Eleazar Eskin and SalStolf: *Intrusion Detection with Unlabeled Data Using Clustering*, Columbia University, 2001
13. http://en.wikipedia.org/wiki/K-means_clustering
14. http://ictf.cs.ucsb.edu/#/
15. Guan Y., Ghorbani, Ali-Akbar, Belacel, Nabil: *An Unsupervised Clustering Algorithm for Intrusion Detection*, published in Advances in Artifcial Intelligence, The 16th Conference of the Canadian Society for Computational Studies of Intelligence, 2003
16. Roger Storl kkenl: *Labelling clusters in an anomaly based IDS by means of clustering quality indexes*, Gj vik University College, 2007
17. Alvaro A. C ardenas, Saurabh Amin and Shankar Sastry: *Research Challenges for the Security of Control Systems*, University of California, Berkeley, 2008
18. Jose F. Nieves and Dr. Yu (Cathy) Jiao: *Data Clustering for Anomaly Detection in Network Intrusion Detection*, Oak Ridge National Laboratory, 2009
19. Mrutyunjaya Panda and Manas Ranjan Patra: *A Novel Classication via Clustering Method for Anomaly Based Intrusion Detection System*, ACEEE International Journal on Network Security, 2010

20. Patel, Ahmed, Qais Qassim, and Christopher Wills.: *A survey of intrusion detection and prevention systems*. Information Management & Computer Security 18.4 (2010): 277-290.
21. Keith Stou
er, Joe Falco and Karen Scarfone: *Guide to Industrial Control Systems (ICS) Security* : NIST Special Publication, 2011
22. Peter Scherer, Martin Vicher, Pavla Drazdilova, Jan Martinovic, Jir Dvorsky and Vaclav Snasel: *Using SVM and Clustering Algorithms in IDS Systems*, Technical University of Ostrava, 2011
23. Harley Kozushko Intrusion Detection: Host-Based and Network-Based Intrusion Detection Systems: Independent Study, September 2003