Մեքենայական լեզվով ծրագրերի պաշտպանությունը վերծանումից

Բաբկեն Վարդանյան

Ապրիլ11, 2016

Կցանկանայի խորին երախտագիտությունս հայտնել իմ ղեկավար >>>ԿՈՉՈՒՄ<<< Մարիամ Հարությունյանին (ԻԱՊԻ), ով ինձ աջակցել և խրախուսել է այս աշխատանքի ժամանակ:

Բաբկեն Վարդանյան

Բովանդակություն

1 Ներածություն

1.1 Տեխնիկական բառարան

անցնել վրայով	parse
անցնել	sweep
ապօրինի օգտագործում	piracy
բազային	base
բաժին	section
բացառություն	exception
բացել	uncompress
բեռնիչ	loader
գլխամաս	header
գծային անցում	linear sweep
դասավորում	alignment
երկակի բառ	dword
թիրախ, սպառնալիք	target
ժառանգված ծրագրեր	legacy software
իրականացում	implementation
լինկեր	linker
խառնել	shuffle
կամայական	optional
կատարվող ֆայլ	executable
կարգաբերիչ	debugger
կարգաբերում	debug
կոթ	handle
հակառակորդ	attacker
հատկություններ	characteristics
հետևում	traversal
ձևափոխել	modify
միջոց, գործողություն	technique
ներմուծում	import
չարամիտ	malicious
պատկեր	image
պրոցես	thread

սեղմող ծրագիր	packer
սկզբնական կոդ	source code
սպասարկում	maintenance
ստորագրություն	signature
վերադասավորում	permutation
վերատեղավորել	relocate
վերլուծություն	analysis
վերծանում	reverse engineering
վնասաբեր ծրագրեր	malware
տվյալների դիրեկտորիա	data directory
փոփոխություններ	tampering
քանդել	disassemble
օբֆուսկացիա	obfuscation
օրինաչափություն	pattern
ՓԿՀ	TCP
ՕԴՀ	UDP
ՆՀՀ	IDS
ԲԾԸԳ	DDoS
ընդլայնում	extension
նախապես որոշված	predefined
լռելյայն	default
ծրագրային միավորներ	module
համացանց	internet
գործընթաց	process
ծրագրային սխալ	bug
կենսափուլ	lifecycle
հակավիրուս	antivirus
ծրագրային սցենար	script
սխալ կոնֆիգուրացիա	misconfiguration
կարկատան	patch

1.2 Ինչու՞ է տեղեկատվական անվտանգությունը կարևոր

1.2.1 Ի՞նչ է տեղեկատվական անվտանգությունը

Տեղեկատվական անվտանգությունը տեղեկատվական ռեսուրսների չլիազորված օգտագործման կանխման և հայտնաբերման պրոցեսն է: [23]

Կանխումը չարամիտ չլիազորված անձանց (նաև ասում են «հակառակորդներ», «հարձակվողներ», «ներխուժողներ», «հաքերներ») կողմից ծրագրային ապահովման կամ տվյալների որոշ մասի օգտագործման դեմ ուղղված միջոցառումների համակարգն է:

Հայտնաբերումը չլիազորված մուտքի փորձի առկայության ստուգման պրոցեսն է: Եթե նման փորձ առկա է, ապա նաև՝ արդոք այն հաջողվել է, և թե կոնկրետ ինչ է տեղի ունեցել: [20]

1.2.2 Ինչու՞ հոգալ տեղեկատվական անվտանգության մասին

Այսօր համակարգիչները և էլեկտրոնային տեխնիկան օգտագործվում են կյանքի գրեթե բոլոր ոլորտներում: Բանկային համակարգի ու ներդրումների ոլորտից մինչև գնումների և հեռահաղորդակցության ոլորտ համակարգիչները դարձել են յուրաքանչյուր բիզնեսի անբաժանելի մասը: Դժվար է նշել մի ոլորտ, որը օգուտ չի քաղել տեղեկատվական տեխնոլոգիաների բուռն զարգացումից:

Չնայած ընկերությունների կողմից պահվող ոչ բոլոր տվյալները կարելի է դասակարգել որպես «հույժ գաղտնի», ցանցային ադմինիստրատորները հավանաբար չեն ուզենա որ անծանոթ անձինք հնարավորություն ունենան հետևել իրենց ընկերության ներքին հաղորդակցությանը, իրենց անձնական ինֆորմացիային, կամ փոփոխություններ կատարեն իրենց վստահված համակարգերում:

Այդ պատճառով տեղեկատվական անվտանգությունը մնում է բիզնեսի և հասարակության առտև ծառացած ամենակարևոր չհաղթահարված խնդիրներից մեկը: [21]

Սերվերի ղեկավարման հնարավորությունը հակառակորդի կողմից ռիսկի տակ է դնում ոչ միայն ընկերությանը, այլ նաև ընկերության հաճախորդներին, ինչպիսիք են օրինակ վեբ կայքի այցելուները:

1.2.3 Ինչու՞ ինչ-որ մեկը կցանկանա կոտրել որոշակի համակարգ

Հակառակորդներին հաճախ չի հուզում թե ով է օգտագործողը կամ ընկերությունը, որի վրա իրականացվում է հարձակումը:

Հակառակորդի հիմնական նպատակներն են՝

- Դրամական եկամուտ - Կոտրված համակարգչից կամ սերվերից օգտագործողի կամ ընկերության բանկային հաշվի և վարկային քարտի տեղեկությունները գողանալու միջոցով
- Բիզնեսի աշխատանքի խոչնդոտում - Մի ընկերություն կարող է վարձել հակառակորդին իրենց մրցակցի համակարգչային ցանցում քաոս ստեղծելու նպատակով
- Ինֆորմացիայի գողություն - Մի ընկերություն կարող է վարձել հակառակորդին իրենց մրցակից ընկերության գաղտնիքները գողանալու և այդպիսով մրցակցային առավելություն ձեռք բերելու նպատակով
- DDoS (Բաշխված ծառայության ընդհատման գրոհ) գրոհներ իրականացնելու նպատակով այլ սերվերների վրա - ԲԾԸ գրոհի նպատակն է սերվիսը անհասանելի դարձնել՝ տարբեր աղբյուրներից չափազանց շատ հարցումներ իրականացնելու միջոցով: [32]

  Նման հարձակման դեպքում կոտրված սերվերների քանակը ուղիղ համեմատական է գրոհի հաջողությանը:
- SEO (Որոնման համակարգերի օպտիմալացում) - Կոտրված կայքը կարող է օգտագործվել այլ կայքերի SEO-ն բարձրացնելու նպատակով՝ կոտրված կայքում տեղադրելով հղումեր դեպի այդ կայքը
- Հենակետ հետագա գրոհների համար - Կոտրված սերվերը կարող է օգտագործվել որպես հենակետ՝
  - Նույն ընկերության ցանցում հետագա ավելի լայնածավալ հարձակումների համար:

  - Ավելի շատ սերվերներին տիրանալը օգնում է հակառակորդին թաքցնել իր ինքնությունը (IP հասցեն)՝ այլ ընկերությունների դեմ հետագա հարձակումների ժամանակ

- Զվարճանք - Հակառակորդը կարող է կոտրել սերվերը զուտ հետաքրքրության կամ զվարճանքի համար

1.3 Սերվերային անվտանգության ժամանակակից պրակտիկան

Սերվերների անվտանգությունը ապահովելու այսօր ընդունված ամենատարածված պրակտիկաներից են՝

1. Անջատել կամ ջնջել ոչ անհրաժեշտ սերվիսները՝
   օպերացիոն համակարգերի լռելյայն կոնֆիգուրացիան երբեմն ապահով չէ: Սովորաբար տեղադրված են բազմաթիվ չօտագործվող սերվիսներ, ինչպես օրինակ՝ պրինտ սերվերը, «Սամբա» ֆայլերի բաշխման համակարգը և այլն: Այս սերվիսները մեծացնում են հարձակման հարթությունը, բացելով ավելի շատ հնարավոր եղանակներ չարամիտ օգտագործողի համար՝ համակարգը չարաշահելու նպատակով:

   Ադմինիստրատորները պետք է անջատեն կամ մեկուսացնեն բոլոր չօգտագործվող սերվիսները, օրինակ՝ firewall-ի օգնությամբ:

2. Հեռակառավարում՝
   Չպաշտպանված, հանրային ցանցերով մուտքը սերվեր հնարավոր է դարձնում հակառակորդների կողմից տարաբնույթ հարձակումներ, ինչպիսին է man-in-the-middle և տվյալների գողություն:

   Ադմինիստրատրոը պետք է համոզվի որ բոլոր հեռակառավարման կապերը դեպի սերվեր պաշտպանված են գաղտնագրմամբ և գաղտնաբառով:

3. Թույլտվություններ և արտոնություններ՝
   Թույլտվությունների հստակ կառավարման համակարգը կարևոր դեր է խաղում սերվերային անվտանգության մեջ: Եթե չարամիտ օգտագործողը կամ պրոցեսը ունենա ավելի շատ արտոնություններ քան իրեն անհրաժեշտ է, այդ հանգամանքը կարող է նպաստել սերվերի կոտրմանը:

   Ադմինիստրատորը պետք է համոզվի, որ բոլոր օգտագործողները մուտք ունեն միայն այն ֆայլերին և ռեսուրսներին, որոնք իրենց անհրաժեշտ են աշխատանքը իրականացնելու համար, և ոչ ավելին:

4. Ժամանակին տեղադրել անվտանգության թարմացումները՝
   Կարևոր է տեղադրել օպերացիոն համակարգը և ծրագրային ապահովումը վերջին թարմացումներով և անվտանգության կարկատաններով:

   Օպերացիոն համակարգի և ծրագրային ապահովման ստեղծողները ժամանակ առ ժամանակ թողարկում են թամացումներ (կարկատաններ): Դրանք հաճախ պարունակում են անվտանգության թարմացումներ, որոնք փակում են հայտնաբերված խոցելիություններ օպերացիոն համակարգում:

   Ադմինիստրատորները պետք է համոզվեն որ թարմացումները տեղադրվում են ժամանակին:

5. Դիտարկում և լոգերի հաշվեքննություն՝
   Լոգեր ստեղծվում են բոլոր տեսակի ծրագրային ապահովման կողմից - օպերացիոն համակարգի, վեբ հավելվածների, բոլոր տեսակի սերվիսների, տվյալների բազաների, ցանցային սարքերի, երթուղավորիչների, սվիչների և այլն կողմից:

   Այս լոգերը պետք է դիտարկվեն և հաճախ ստուգվեն, քանի որ նրանք երբեմն կարող են զգուշացնել գալիք վտանգի մասին: Նույնիսկ հաջող հարձակման դեպքում սերվերների լոգերը հաճախ դատական փորձաքննություն իրականացնելու միակ միջոցն են:

6. Օգտագործողի հաշիվներ՝
   Չօգտագործվող օգտագործողի հաշիվները, ինչպիսիք են աշխատանքից ազատված աշխատակիցները, պետք է անջատվեն: Պետք է անջատվեն նաև զանազան սերվիսների կողմից ստեղծված օգտագործողների հաշիվները:

   Յուրաքանչյուր օգտագործողի հաշիվ մեծացնում է հարձակման հարթությունը: Նախկին աշխատակիցը կարող է ընկերությանը վնաս հասցնելու դրդապատճառներ ունենալ, և եթե նրա նախկին օգտագործողի հաշիվը անջատված չլինի՝ նա հնարավորություն կունենա ցանկացած գործողություն կատարել իր օգտագործողի իրավասություններով:

   Յուրաքանչյուր ադմինիստրատոր և օգտագործող ով մուտք է գործում սերվեր պետք է ունենա իր սեփական հաշիվը և գաղտնաբառը, և ճիշտ իրավասություններ: Գաղտնաբառը չպետք է բաշխվի օգտագործողներիր միջև:

7. Ջնջել չօգտագործվող մոդուլներ և ընդլայնումներ՝
   Հավելվածները ինչպես օրինակ վեբ սերվերները հաճախ կարող են պարունակել որոշակի լռելյայն ընդլայնումներ և ծրագրային միավորներ: Այս ծրագրային միավորները կարող են պարունակել խոցելիություններ, և այդպիսով մեծացնել հնարավոր հարձակման հարթությունը հակառակորդի համար:

   Ադմինիստրատորը պետք է համոզվի որ հնարավորության դեպքում միայն վեբ հավելվածների համար անհրաժեշտ միավորներն են առկա:

8. Լինել տեղեկացված՝
   Այսօր օպերացիոն համակարգերի և ծրագրային ապահովման, այդ թվում՝ դրանց անվտանգության մասին ինֆորմացիան ազատորեն հասանելի է համացանցում:

   Ադմինիստրատորները պետք է համոզվեն որ իրենք և իրենց օգտագործողները մշտապես տեղեկացված են հարձակումների և խոցելիությունների մասին վերջին լուրերին:

9. Օգտագործել սկզբնական կոդի անվտանգության սկաներներ՝
   Սկաներները ծրագրեր են, որոնք ավտոմատացնում և հեշտացնում են սերվերի և հավելվածների պաշտպանության գործընթացը:

   Ծրագրային կոդի ստատիկ և դինամիկ անալիզի գործիքները ինչպիսիք են Sonar -ը Java լեզվի համար, Valgrind-ը C լեզվի համար և այլն օգնում են գտնել ծրագրային սխալներ և խոցելիություններ ծրագրի կենսափուլի վաղ շրջանում:

10. Ընտրել գաղտնագրման և հեշավորման ապահով ալգորիթմներ՝
    Պետք է խուսափել կոտրված գաղտնագրման, հաղորդակցության և հեշավորման արձանագրությունների օգտագործումից, ինչպիսիք են՝ DES, SSL, MD5:

Այս արձանագրությունների թուլությունը հարձակման հնարավոր վեկտոր է բացում հակառակորդի համար:

Այսպիսի արձանագրությունները պետք է փոխարինվեն ժամանակակից, չկոտրված և գաղտնագրման լայն հանրության վստահությանը արժանացած արձանագրություններով:

11. Օգտագործել հակավիրուս՝
Վինդուս օպերացիոն համակարգի վրա հիմնված սերվերներում անհրաժեշտ է տեղադրել հակավիրուսային ծրագրային ապահովում: [16]

Հակավիրուսը սկանավորում է ծրագիրը հետևյալ պայմաններում՝

(a) Ամբողջական սկաներ - թողարկվում են պարբերաբար կամ օգտագործողի կողմից

(b) Աշխատանքի ժամանակ, այսինքն երբ համակարգով փոխանցվում են տվյալներ

Հակավիրուսները օգտագործում են վիրուսների հայտնաբերման հետևյալ տեխնոլոգիաները՝

(a) Ստորագրման վրա հիմնված հայտնաբերում Ֆայլը համեմատվում է հայտնի չարամիտ կոդի հետ

(b) Փորձարարության վրա հիմնված հայտնաբերում Ֆայլի վարվելաձևը համեմատվում է հայտնի չարամիտ նմուշների հետ

(c) Վարվելակերպի վրա հիմնված հայտնաբերում Սա հաճախ կատարվում է ՆՀՀ-երում

Լինուքսի վրա հիմնված համակարգերում հակավիրուս հաճախ չի օգտագործվում: [17]

Լինուքսի վրա հիմնված համակարգերում հակավիրուսի անհրաժեշտություն կարող է առաջանալ միայն այն պարագայում, երբ այն օգտագործվում է Վինդոուս համակարգերի միջև ֆայլերի փոխանակաման համար: [19]

12. Օգտագործել ցանցային սկաներներ՝
Ցանցային սկաներները օգնում են ադմինիստրատորներին համոզվել իրենց սերվերների անվտանգության մեջ: Այսպիսի գործիքները կարողանում են հայտնաբերել բաց պորտեր, խոցելի սերվիսներ, և նույնիսկ վիրուսներ: Հայտնի ցանցային սկաներներից են՝

(a) Nmap

(b) Nessus

(c) Accunetix

Համակարգային ադմինիստրատորների տարածված պարտականություններից է իրենց վստահված համակարգերում պորտերի սկանավորման իրականացումը: Այսպիսի սկանավորումները օգնում են ադմինիստրատորներին գտնել խոցելիություններ իրենց համակարգերում ավելի վաղ, քան հնարավոր հակառակորդը: Այսպիսի սկանավորումներ իրականացնելու համար օգտագործվում են այնպիսի գործիքներ ինչպիսիք են՝ nmap, nessus, accunetix և այլն: Ցանցային պորտերի սկանավորման պրոցեսը հաճախ այսպիսի հաջորդականություն ունի՝

(a) Ադմինիստրատորը որոշում է հասցեների և պորտերի շրջանակը, որոնք պետք է ենթարկվեն սկանավորման:

(b) Նա տալիս է ծրագրին այդ պարամետրերը և սկսում է սկանավորումը

(c) Ծրագիրը փորձարկում է IP հասցեների և պորտերի բոլոր տրված կոմբինացիաները

(d) Եթե պարզվում է, որ պորտը բաց է, ապա աշխատեցվում է հատուկ ծրագրային սցենար, որը փորձում է գուշակել աշխատող սերվիսի մասին տվյալները՝ անունը, տարբերակը, կոնֆիգուրացիան, մատչելի օգտագործողների անունները, և այլն:

(e) Տվյալները տրվում են ադմինիստրատորին նրա նախընտրած ֆորմատով՝ XML, ելք հրամանային տողում կամ ծրագրին հատուկ ֆորմատով

2 Խնդիրը

2.1 Անհրաժեշտություն

Նախորդ բաժնի վերջին կետում մշված ցանցային սկաներների ներկայիս իրականցումը ունի որոշակի թերություններ՝

1. Ցանցում բազմաթիվ համակարգերի գոյության դեպքում յուրաքանչյուր TCP և UDP պորտի սկանավորումը պահանջում է բավականին երկար ժամանակ: Սկանավորումը արագացնելու նպատակով հնարավոր է սկանավորել միայն պորտերի սահմանափակ բազմություն, սակայն այդ դեպքում պատկերը ամբողջական չի լինի, քանի որ ոչ հայտնի պորտերի տակ նույնպես հնարավոր է աշխատի ինչ-որ սերվիս, և այն չի հայտնաբերվի նման սկանավորման ժամանակ:

2. Այն ծախսում է ցանցային ռեսուրսներ և կարող է որոշ համակարգեր անհասանելի դարձնել սկանավորման ընթացքում

3. Որոշակի սցենարների դեպքում պորտերի սկանավորումը կարող է հանգեցնել IDS-ում կեղծ ահազանգի

4. Հնարավոր են կեղծ դրական արդյունքներ և սերվիսների սխալ նույնականացումներ:

5. Չեն հայտնաբերվում բացթողումներ հետևյալ ասպարեզներում՝

   - Թույլտվություններ և արտոնություններ
   - Թարմացումների առկայություն
   - Օգտագործողի հաշիվներ
   - Գաղտնագրման և հեշավորման ապահով ալգորիթմների օգտագործում
   - Հակավիրուսի օգտագործում

Այսպիսով անհրաժեշտ է որոնել սկանավորում իրականցնելու մեկ այլ եղանակ, որը զերծ կլինի վերը նշված թերություններից:

2.2 Այլընտրանք

Այս փաստաթղթում մենք ներկայացնում ենք սերվերների խոցելիությունների հայտնաբերման այլընտրանքային եղանակ, որը սկանավորում է համակարգերը ներսից, և այդպիսով զերծ է վերը նշված թերություններից:

Յուրաքանչյուր բաց պորտի համար ծրագիրը սկանավորում է այդ սերվիսի կոնֆիգուրացիոն ֆայլը խոցելիությունների և դատարկ գաղտնաբառերի առկայության համար և հայտնում է արդյունքները օգտագործողին:

Բացի որոշելուց թե արդյոք պորտը բաց է թե ոչ, այն նաև ստուգում է թե արդյոք այն ֆիլտրված է firewall-ով:

2.3 Նախկին փորձի ուսումնասիրություն

2.3.1 Microsoft Baseline Security Analyzer

Այս ծրագրային ապահովման ճարտարապետությունը մասամբ ոգեշնչվել է MSBA ծրագրի կողմից: [24]

MSBA-ը Վինդուս համակարգերի համար նախատեսված անվտանգության սկաներ է, ստեղծված Microsoft ընկերության կողմից: Այն գնահատում է Վինդուս համակարգի և Microsoft-ի այլ ապրանքների անվտանգությունը առավել հաճախ հանդիպող սխալների առկայության համար և արդյունքները ներկայացնում է օգտագործողին:

MSBA-ը ունի որոշակի սահմանափակումներ՝

- Աշխատում է միայն Վինդուս ճարտարապետության համակարգերում
- Ստուգումներ իրականցնում է միայն Microsoft ընկերության կողմից ստեղծված ծրագրերում

2.3.2 buck-security

buck-security-ն անվտանգության սկանավորիչ է Debian և Ubuntu Linux օպերացիոն համակարգերի համար: [25]

Այս աշխատանքում ներկայացվող ծրագիրը որոշ չափով նման է buck-security-ին:

buck-security-ն ունի որոշակի սահմանափակումներ նույնպես՝

- Նախատեսված է Debian և Ubuntu համակարգերի համար միայն
- Գտնվում է Beta փուլում, և խորհուրդ չի տրվում այն օգտագործել արտադրության համակարգերում

2.3.3 Lynis

Lynis-ը անվտանգության աուդիտի և կարծրացման գործիք է UNIX համակարգերի համար: Այն օգնում է ադմինիստրատորներին արագ հայտնաբերել և լուծել անվտանգության սխալները: [29]

Օպերացիոն համակարգեր: Unix ընտանիք Լիցենզիա: Հանրային տարբերակը՝ GPL3, կա նաև վճարովի առևտրային տարբերակ: [30]

2.3.4 Tiger

Tiger-ը անվտանգությունը գնահատող ծրագիր է UNIX համակարգերի համար:

Ցավոք, այն ներկայումս ակտիվորեն չի մշակվում: Վերջին կայուն տարբերակը թողարկվել է 2010 թվականին:

Օպերացիոն համակարգեր: Unix ընտանիք Լիցենզիա: GPL3 [31]

3 Պահանջներ

Այս աշխատանքի նպատակն է ստեղծել ծրագրային հավելված, որը Լինուքսի վրա հիմնված սերվերային համակարգի վրա տեղադրման պարագայում աշխատեցնելիս կգնահատի համակարգերի անվտանգությունը և կհայտնի արդյունքները օգտագործողին:

Ծրագրային հավելվածի առաջնային նպատակն է օգտագործողին ներկայացնել համակարգի անվտանգության ընդհանուր պատկերը:

Ծրագիրը պետք է աշխատի բոլոր ժամանակակից Linux համակարգերի տակ: Հնարավորության դեպքում՝ նաև UNIX ընտանիքի այլ համակարգերում:

Ծրագրի տեղադրումը պետք է լինի հնարավորինս պարզ:

Ծրագրի ստուգումներից յուրաքանչյուրը պետք է հնարավոր լինի անջատել՝ մյուսներից անկախ:

Եթե ծրագրի մի մոդուլը իրականացնում է բազմատեսակ ստուգումներ, ապա դրանցից յուրաքանչյուրը պետք է հնարավոր լինի անջատել՝ մյուսներից անկախ:

Ստորև ներկայացվում է ծրագրի կողմից կատարվող ստուգումների ցանկը:

3.1 Ծրագրային թարմացումներ

Ծրագիրը պետք է ստուգի թե վերջին անգամ երբ է թարմացվել օպերացիոն համակարգը: Եթե դա կատարվել է բավականաչափ ուշ, ապա օգտագործողը պետք է զգուշացվի, հայտնելով վերջին թարմացման ժամանակը:

Այդ ժամանակային միջակայքը պետք է հնարավոր լինի կարգաբերել ծրագրի կոնֆիգուրացիայով:

Ստորև ներկայացված են Լինուքսի յուրաքանչյուր տարբերակին առանձնահատուկ վերջին թարմացման ժամանակի ստուգումները՝

3.1.1 Debian/APT-ի վրա հիմնված համակարգեր

Ծրագիրը պետք է որոշի վերջին թարմացման ժամանակը /var/lib/apt/periodic/update-success-stamp ֆայլի ստեղծման ժամանակով: [8]

Դա Ubuntu ընտանիքին հատուկ ֆայլ է, որի ստեղծման ժամանակը համընկնում է apt-get update հրահանգի վերջին կատարման ժամանակի հետ: Դա պայմանավորված է նրանով, որ նշված հրահանգը կատարելիս աշխատեցվում է /etc/apt/apt.conf.d/15update-stamp ծրագիրը, որը և թարմացնում է նշված դրոշմ-ֆայլը՝ վերագրելով նրա ստեղծման ժամանակը ներկայիս պահին: [34]

3.1.2 Red Hat/YUM-ի վրա հիմնված համակարգեր

Ծրագիրը պետք է որոշի վերջին թարմացման ժամանակը `yum history` հրահանգի ելքը վերլուծելով: [9]

yum history հրահանգը արտածում է նմանօրինակ ելք՝

```
# yum  history
Loaded plugins: fastestmirror, refresh-packagekit
ID     | Login user               | Date and time    | Action(s)      | Altered
-------------------------------------------------------------------------------
    41 | root <root>              | 2012-04-27 20:17 | Install        |   19
    40 | root <root>              | 2011-11-20 10:09 | Install        |   10
    39 | root <root>              | 2011-11-20 08:14 | Install        |    1 E<
    38 | root <root>              | 2011-11-19 15:46 | Update         |    1
```

[35]

Օպերացիոն համակարգի վերջին թարմացման ժամանակը հնարավոր է որոշել այս ելքը վերջից փնտրելով 'Update' բառը, այնուհետև առաջին համընկնող տողում վերլուծելով 'Date and time' սյան տեքստը:

3.1.3 Arch Linux/Pacman-ի վրա հիմնված համակարգեր

Pacman փաթեթների մենեջերի գործողությունների գրանցամատյանը գտնվում է `/var/log/pacman.log` ֆայլում: [10] Ծրագիրը պետք է ստուգի վերջին թարմացման ժամանակը վերլուծելով վերոնշյալ ֆայլի պարունակությունը: Այն ունի նմանօրինակ պարունակություն՝

```
[2016-04-05 10:22] [ALPM] transaction started
[2016-04-05 10:23] [ALPM] installed pycharm-community (2016.1-1)
[2016-04-05 10:23] [ALPM] transaction completed
[2016-04-05 10:24] [PACMAN] Running 'pacman -S -y -y -u'
[2016-04-05 10:24] [PACMAN] synchronizing package lists
[2016-04-05 10:24] [PACMAN] starting full system upgrade
[2016-04-05 10:24] [PACMAN] Running 'pacman -S -y -y -u'
[2016-04-05 10:24] [PACMAN] synchronizing package lists
[2016-04-05 10:24] [PACMAN] starting full system upgrade
[2016-04-05 10:28] [ALPM] transaction started
[2016-04-05 10:28] [ALPM] upgraded tzdata (2016b-1 -> 2016c-1)
[2016-04-05 10:28] [ALPM] upgraded alsa-utils (1.1.0-1 -> 1.1.0-2)
[2016-04-05 10:28] [ALPM] upgraded graphite (1:1.3.6-1 -> 1:1.3.8-1)
[2016-04-05 10:28] [ALPM] upgraded harfbuzz (1.2.3-1 -> 1.2.4-1)
[2016-04-05 10:28] [ALPM] upgraded fontconfig (2.11.1-2 -> 2.11.94-1)
[2016-04-05 10:28] [ALPM-SCRIPTLET] updating font cache... done.
[2016-04-05 10:28] [ALPM] installed tslib (1.1-1)
[2016-04-05 10:28] [ALPM] upgraded libxkbcommon (0.5.0-1 -> 0.6.0-1)
```

Օպերացիոն համակարգի վերջին թարմացման ժամանակը հնարավոր է որոշել այս ելքը վերջից փնտրելով 'starting full system update' նախադասությունը, այնուհետև առաջին համընկնող տողում վերլուծելով ժամանակը, որը գտնվում է առաջին քառակուսի փակագծերի մեջ:

3.2 Ֆայլերի և դիրեկտորիաների թույլտվություններ

Բոլոր ֆայլերը և դիրեկտորիաները պետք է ունենան ճիշտ թույլտվություններ: Հակառակ դեպքում համակարգը խոցելի է: Բոլոր օգտագործողների կողմից գրման հնարավո-

րություն ունեցող (worldwritable) ֆայլերը և դիրեկտորիաները կարող են օգտագործվել հակառակորդի կողմից՝ կամեցած ֆայլի կամ դիրեկտորիայի մեջ ցանկացած բան փոփոխելու կամ ջնջելու համար: [26][25]

Բացառություն են կազմում այն worldwritable դիրեկտորիաները, որոնք ունեն sticky բիթ, ինչպես նաև այն բոլոր ֆայլերը որոնք չեն սկսվում կետով և չեն պատկանում համակարգային օգտագործողին:

Linux-ի ֆայլային համակարգերում Stiky բիթը դիրեկտորիայի հատուկ ատրիբուտ է: Եթե այն առկա է, ապա նշանակում է որ նրանում պարունակվող ֆայլերի տերը միայն (owner) և համակարգային օգտագործողը (root user) իրավունք ունեն ջնջել կամ վերանվանել վերոնշյալ ֆայլը:

Այսպիսով, այս ստուգման ժամանակ նման ֆայլերի և դիրեկտորիաներ որոնման ժամանակ պետք է կիրառել հետևյալ ֆիլտրը՝

- worldwritable դիրեկտորիաները պետք է ունենան sticky բիթ
- worldwritable ֆայլերը չպետք է սկսվեն կետով (".")
- worldwritable ֆայլերը չպետք է պատկանեն համակարգային օգտագործողին (root user)

Նման ֆայլերի հայտնաբերման դեպքում ծրագիրը պետք է զգուշացնի, և թվարկի այդ ֆայլերը:

3.3 Բաց TCP և UDP պորտեր

Յուրաքանչյուր բաց պորտ մեծացնում է հարձակման հարթությունը: [37]

Այդ պատճառով անհրաժեշտ է սահմանափակել բաց պորտերի քանակը, կամ դրանք ծածկել firewall-ով:

Ծրագիրը պետք է օգտագործողին ներկայացնի բոլոր բաց TCP և UDP պորտերի ցանկը, և դրանց տակ աշխատող սերվիսների անունները, հնարավորության դեպքում՝ նաև այն կատարվող ֆայլի անունը, որը գործարկվել է սերվիսը աշխատեցնելիս:

3.4 Օգտագործողների UMASK ստուգում

UMASK-ը օգտագործողի ատրիբուտներ է, որը որոշում է թե նոր ստեղծված ֆայլերը ինչ թույլտվություններ պետք է ունենան:

Եթե օգտագործողի UMASK-ը այնպիսին է, որ ստեղծում է worldwritable ֆայլեր, ապա սա ունի նույն թերությունները ինչ նախորդ բաժնում նկարագրված թերությունը: [28]

3.5 Ֆայլերի և դիրեկտորիաների SETUID և SETGID ստուգում

Եթե SETUID բիթը դրվում է կատարվող ֆայլի վրա, ապա այն պրոցեսը որը ստեղծվում է այդ ֆայլը աշխատեցնելիս, աշխատում է այդ ֆայլի օգտագործողի թույլտվություններով:

Նույնը կատարվում է SETGID բիթի տեղադրման ժամանակ՝ ֆայլի խմբի համար: [27]

3.6 Մուտք որպես համակարգային օգտագործող

Ծրագիրը պետք է ստուգի՝ արդյոք ներկայիս օգտագործողի սեսսիան լոգին սեսսիա է թե ոչ, և արդյոք նա ունի ադմինիստրատորի իրավասություններ:

Այսինքն, եթե օգտագործողը մուտք է գործել համակարգ որպես համակարգային օգտագործող (root user), ապա պետք է զգուշացնել: [12]

Բացառություն է կազմում այն դեպքը, եթե ծրագիրը իրականցվում է `sudo՝ հրահանգով: Այս դեպքում զգուշացնել պետք չէ:

3.7 Դատարկ կամ թույլ գաղտնաբառեր

Եթե սերվերը արտաքինից հասանելի է սերվիսների միջոցով, որոնք իսկության ստուգման համար օգտագործում են լոկալ լինուքսի օգտագործողների հաշիվները, ապա ծրագիրը ստուգում է թե արդյոք այդ օգտագործողները ունեն դատարկ գաղտնաբառեր: [13]

Դա կատարվում է՝

3.8 Համոզվել որ ոչ համակարգային օգտագործողների UID-ն 0 է

[15]

3.9 Ամենատարածված սերվիսներում ոչ անվտանգ կոնֆիգուրացիաների առկայության ստուգումներ

3.9.1 SSHd

SSH սերվերի կոնֆիգուրացիոն ֆայլն է՝

/etc/ssh/ssh_config

Այսօր խոցելի համարվող SSH v1 արձանագրությունը չպետք է միացված լինի: Այս համակարգի խոցելիությունը խոցվել է վայրի միջավայրում WOOT նախագծի կողմից: [7]

Ծրագիրը նաև ստուգում է՝ արդյոք SSH-ի գաղնտաբառով մուտքի հնարավորությունը թույլատրվում է: Եթե այո, ապա օգտագործողը զգուշացվում է: [11]

cat PasswordAuthentication no

3.9.2 MySQL

MySQL-ը այսօր ամենատարածված ռելացիոն տվյալների բազաներից է աշխարհում: [4] Ծրագիրը սկանավորում է հետևյալ կոնֆիգուրացիոն ֆայլերը սխալների համարէ

/etc/my.cnf /etc/mysql/my.cnf /.my.cnf

3.9.3 Telnet

Եթե telnet-ի աշխատող սերվիս է հայտնաբերվում, ապա օգտագործողը զգուշացվում է: [13]

3.9.4 FTP

Բացառությամբ այն դեպքի, որ աշխատող FTP սերվիսը միայն կարդացվող և հանրորեն հասանելի է, օգտագործողը զգուշացվում է FTP-i օգտագործման դեմ: FTP արձանագրությունը ապահով չէ, քանի որ օգտագործողի անունը և գաղտնաբառը փոխանցվում են բացիեբաց: [14]

4 Իրականացում

4.1 Ծրագրավորման լեզվի ընտրություն

Ծրագրի իրականացման համար ընտրվել է Python 3 լեզուն: Այն ունի մի շարք առավելություններ նշված խնդրի իրականացման համար՝ [36]

1. Python-ի շարահյուսությունը չափազանց հեշտ է և՛ սովորել և՛ հասկանալ
2. Python-ը անվճար է և ունի ազատական լիցենզիա
3. Python-ը աշխատում է բոլոր հիմնական օպերացիոն համակարգերում՝ Windows, Linux, OS X
4. Python-ը ունի ներդրված և հասանելի գրադարանների առատ բազմություն

4.2 Ծրագրային պահանջներ

Ծրագրի աշխատանքի համար անհրաժեշտ է՝

1. Unix-ի վրա հիմնված օպերացիոն համակարգ
2. Python-ի նոր տարբերակ: Ծրագիրը փորձարկվել է Python 3.5.1-ով
3. `psutil՝ (python process and system utilites) Python գրադարանը

4.3 Մոդուլների իրականացումը

4.3.1 Ծրագրային թարմացումներ՝ update.py

Նախ ծրագիրը որոշում է թե ինչ օպերացիոն համակարգի միջավայրում է այն աշխատում: Այս խնդրի լուծման համար օգտագործվել է platform.linux_distribution() կանչը: Կախված այդ կանչի արդյունքից աշխատեցվում է օպերացիոն համակարգին հատուկ ծրագիրը, որը որոշում է համակարգի վերջին թարմացման ժամանակը՝

- Եթե ծրագիրը աշխատում է Arch Linux միջավայրում, ապա ծրագիրը տող-առ-տող կարդում է /var/log/pacman.log ֆայլի պարունակությունը: Եթե հերթական տողում առկա է 'starting full system upgrade' բառակապակցությունը, ապա ծրագիրը դադարեցնում է կարդալ տողերը և վերջին թարմացման ժամանակը համարում է նշված տողի առաջին քառակուսի փակագծերի միջև գտնվող ժամանկային գրառումը:

  Եթե 'starting full system upgrade' բառակապակցությունը չի հայտնաբերվել, ապա ծրագիրը հայտնում է ստուգման անհնարինության մասին:

- Եթե ծրագիրը աշխատում է Debian կամ Ubuntu Linux միջավայրում, ապա ծրագիրը որոշում է /var/lib/apt/periodic/update-success-stamp ֆայլի վերջին փոփոխման ժամանակը:

  Եթե ֆայլը չի հայտնաբերվել, ապա ծրագիրը հայտնում է ստուգման անհնարինության մասին:

- Եթե ծրագիրը աշխատում է Red Hat/CentOS միջավայրում, ապա ծրագիրը տող-առ-տող կարդում է yum history հրահանգի ելքը: Եթե հերթական տողում առկա է 'Update' բառը, ապա ծրագիրը դադարեցնում է կարդալ տողերը և վերջին թարմացման ժամանակը համարում է նշված տողի Date and time սյունակում գտնվող ժամանկային գրառումը:

  Եթե 'Update' բառը չի հայտնաբերվել, ապա ծրագիրը հայտնում է ստուգման անհնարինության մասին:

Ստացված ամսաթիվը համեմատվում է ծրագրի կոնֆիգուրացիայի update.warn_last_update_interval_days գրառման հետ: Եթե վերջին թարմացման ամսաթիվը ավելի նոր է, ապա ստուգումը համարվում է հաջող, հակառակ դեպքում՝ անհաջող:

4.3.2 Ֆայլերի և դիրեկտորիաների թույլտվություններ՝ worldwritable.py

Նախ ծրագիրը փնտրում է հետևյալ ֆիլտրին համապատասխանող բոլոր ֆայլերը և դիրեկտորիաները՝

1. Ֆայլեր, որոնք worldwritable են և սկսվում են կետով
2. Դիրեկտորիաներ, որոնք worldwritable են և չունեն sticky բիթ
3. Ֆայլեր, որոնք worldwritable են և պատկանում են համակարգային օգտագործողին (root)

Ֆայլերի և դիրեկտորիաների փնտրումը կատարվում է os.walk() կանչի օգնությամբ, իսկ ատրիբուտների ստուգումը՝ os.stat() կանչով:

Եթե վերոնշյալ պայմաններին համապատասխանող ֆայլեր կամ դիրեկտորիաներ հայտնաբերվել են, ապա ծրագիրը արտածում է թե որ պայմանն է խախտվել, և այդ ֆայլերի կամ դիրեկտորիաների ցուցակը:

Եթե մի քանի պայմաններ են խախտվել, ապա յուրաքանչյուր պայմանի համար կատարվում է առանձին ելք:

5 Եզրակացություն

Այս ծրագրային ապահովման իրականացման ընթացքում պարզվեց, որ՝

- Անվտանգության ամենակարևոր գործոնը մարդն է:
- Նմանօրինակ ծրագրային հավելվածը օգտակար կարող է լինել միայն բազմակողմանի աջակցության և երկարատև զարգացման դեպքում:
- Սկանավորում իրականացնող գործիքները սահմանափակ են իրենց կարողություններում՝ անվտանգությանը նպաստելու տեսանկյունից:

Գրականության ցանկ

1. 1
   http://sectools.org/
2. 2
   https://docs.python.org/2/library/socket.html
3. 3
   https://pythonhosted.org/psutil/
4. 4
   http://db-engines.com/en/ranking
5. 5
   http://www.yolinux.com/TUTORIALS/LinuxTutorialInternetSecurity.
   html
6. 6
   http://www.yolinux.com/TUTORIALS/LinuxTutorial-woot-project.html
7. 7
   http://www.iss.net/threats/advise100.html
8. 8
   http://serverfault.com/questions/20747/find-last-time-update-was-perf
9. 9
   http://serverfault.com/questions/389650/how-to-check-when-yum-update-
10. 10
    https://bbs.archlinux.org/viewtopic.php?id=150428
11. 11
    https://www.digitalocean.com/community/tutorials/
    7-security-measures-to-protect-your-servers
12. 12
    http://askubuntu.com/questions/16178/why-is-it-bad-to-login-as-root
13. 13
    http://www.tecmint.com/linux-server-hardening-security-tips/
14. 14
    https://www.digitalocean.com/community/tutorials/
    an-introduction-to-securing-your-linux-vps
15. 15
    http://www.cyberciti.biz/tips/linux-security.html
16. 16
    http://serverfault.com/questions/632/do-you-run-antivirus-on-your-win

17. 17
http://www.howtogeek.com/135392/htg-explains-why-you-dont-need-an-ant
?PageSpeed=noscript
18. 18
https://antivirus.comodo.com/how-antivirus-software-works.php
19. 19
http://security.stackexchange.com/a/53462/37546
20. 20
http://cybercellmumbai.gov.in/html/general-tips/what_is_computer_
security.html
21. 21
http://www.acunetix.com/websitesecurity/webserver-security/
22. 22
https://www.onehoursitefix.com/why-would-hackers-hack-my-website/
23. 23
http://searchsecurity.techtarget.com/definition/
information-security-infosec
24. 24
https://msdn.microsoft.com/en-us/library/ff647642.aspx
25. 25
http://www.buck-security.net/buck-security.html
26. 26
http://www.softpanorama.org/Access_control/Permissions/world_
writable_files_problem.shtml
27. 27
http://shop.oreilly.com/product/9780596527631.do
28. 28
http://www.cyberciti.biz/tips/understanding-linux-unix-umask-value-us
html
29. 29
https://github.com/CISOfy/lynis/
30. 30
https://cisofy.com/pricing/
31. 31
http://git.savannah.gnu.org/cgit/tiger.git/
32. 32
http://www.digitalattackmap.com/understanding-ddos/

33. 33
    https://security.illinois.edu/content/updates-and-patches
34. 34
    http://serverfault.com/questions/20747/find-last-time-update-was-perf
35. 35
    http://serverfault.com/questions/389650/how-to-check-when-yum-update-
36. 36
    https://en.wikiversity.org/wiki/Python/Why_learn_Python
37. 37
    http://superuser.com/questions/82488/why-is-it-bad-to-have-open-ports

TODO: iso 27001 TODO: http://lazy2hack.blogspot.am/2010/03/collection-of-security-checks-for-linux.html

ՀԱՎԵԼՎԱԾ

Հավելված

Սկզբնական կոդ

main.py

Իրականացման աշխատանքը