ივანე ჯავახიშვილის სახელობის თბილისის სახელმწიფო უნივერსიტეტი

არჩილ კალანდაძე

ანონიმურობის უზრუნველყოფა Tor-ის გამოყენებით და დეანონიმიზაციის რისკები

ინფორმაციული ტექნოლოგიები

ნაშრომი შესრულებულია ინფორმაციული ტექნოლოგიების მაგისტრის აკადემიური ხარისხის მოსაპოვებლად

ხელმძღვანელი: პროფ.ლელა მირცხულავა

თბილისი

2018

სარჩევი

ანოტაცია ----------3

1 შესავალი ----------4

2 Onion Routing ----------5

2.1 Onion Routing----------5

2.2 პირველი თაობა(1996-2004)----------6

2.3 მეორე თაობა (2004-2006)----------7

2.4 მესამე თაობა(2006-)----------7

3 Tor ქსელი ----------8

3.1 Tor ქსელი ----------8

3.2 Onion წრედის (ჯაჭვის) შექმნა----------9

3.3 შეტყობინება Onion წრედში ----------11

3.4 უჯრედი (Cells) ----------12

4 თავდასხმის საფრთხეები ----------13

4.1 დეანონიმიზაციის კატეგორიები ----------14

4.2 შეტევები ტორზე----------15

4.3 კორელაციის ტიპის შეტევები (Correlation Attacks)----------15

4.4 დაგუბების“ ტიპის შეტევა (Congestion type attack) ----------18

4.5 DOS ტიპის შეტევა----------19

4.6 დამხმარე შეტევები ----------21

5 Tor აპლიკაციის ეფექტურობის შეფასება ----------22

5.1 ტესტი ბრაუზერის Default პარამეტრებით---------- 23

5.2 ტესტი ვებ პროქსის და Tor-ის გამოყენებით----------25

6 დასკვნა----------27

7 გამოყენებული ლიტერატურა ----------28

Abstract

The main goal of encrypted networks is to create such paths in Internet, that ensures anonymity of message's sender. Tor application uses method called "Onion Routing" for encrypting messages. The name of this technique is based on several layers of encryption applied to outgoing message from a sender to a receiver. Computer that establishes connection and creates path, encrypts its message with several layers of encryption which then are peeled off by Onion Routers.

Tor intends to stay as the best low latency anonymity service among many similar applications. The first version of Tor was created in 1996. Latest, 3rd version of Tor is still under constant development and was created in 2006.

In this thesis we discuss how Tor has evolved, what kind of attacks exist against it and what sort of defences have been developed against them. We will review theoretical attacks on Tor which break anonymity and try to categorise them.

In addition, To assess effectiveness of implementation, a series of tests will be conducted with the help of an existing web application. The web application will check Tor application's behavior by conducting test in a various potential vurnelable parts. The results from the evaluation are going to be guide for the users and will help them stay anonymous while using the Internet.

ანოტაცია

კრიპტოგრაფიულად დაშიფრული ქსელების მიზანია შექმნან კომუნიკაციის ისეთი გზები ინტერნეტში, სადაც შეტყობინების გაგზავნისას შეუძლებელი იქნება მისი დაკავშირება გამგზავნთან. აპლიკაცია Tor- იყენებს ე.წ „Onion Routing" -ს შეტყობინებების დასაშიფრად. ამ მეთოდის სახელი განპირობებულია შეტყობინების რამდნენიმე დონით დაშიფვრის გამო გამგზავნიდან ადრესატამდე. კომპიუტერი, რომელიც ამყარებს კავშირს და ქმნის გზას, შეტყობინებას შიფრავს რამდენიმე ფენით , რომლებიც სათითაოდ სცილდება Onion მარშუტიზატორების დახმარებით.

Tor-ის მიზანია დარჩეს დაბალი მოლოდინის(Latency) დროით გამორჩეულ ანონუმურობის სერვისად. Tor- ის პირველი ვერსია შეიქმნა 1996 წელს, ხოლო ბოლო მესამე თაობის მოდელი ჯერ კიდევ დამუშავების პროცესშია და 2006 წელს გამოვიდა.

ნაშრომში განვიხილავთ Onion Routing-ს : თუ რითი განსხვავდება სხვადასხვა თაობის Tor - ის აპლიკაცია , რა სახის შეტევების არსებობს მის წინააღმდეგ და როგორია მათგან თავდაცვის გზები. განვიხილავთ თეორიულ შეტევებს, რომლებითაც შესაძლებელია ანონიმურობის დარღვევა და დავყოფთ მათ კატეგორიებად.

ასევე, პრაქტიკულად შევამოწმებთ Tor აპლიკაციის ეფექტურობას რამდენიმე

Web-ზე დაფუძნებული ტესტის საშუალებით. ვებ აპლიკაცია შეამოწმებს Tor იმპლემენტაციის იმ ცვლადების ქცევებს, რომლებიც პოტენციურად სუსტები არიან დეანონიმიზაციის კუთხით. ტესტის შედეგები, გამოდგება ერთგვარ სახელმძღვანელოდ მომხმარებლებისთვის და დაეხმარება მათ დარჩნენ მაქსიმალურად ანონიმურები ინტერნეტით სარგებლობისას.

1. შესავალი

IP პაკეტი, რომელიც გამოიყენება ინტერნეტში ინფორმაციის გასაგზავნად, შედგება ორი ნაწილისგან : Payload (სადაც მოთავსებულია უშუალოდ ინფორმაცია) და Header იგივე Metadata (სადაც წერია ინფორმაცია გამგზავნზე, ადრესატზე, დროზე და ინფორმაციის ზომაზე). ჩვეულებრივ, Payload ნაწილი პაკეტის დაშიფრულია ხოლო Header-ი არა. ეს ქმნის ინფორმაციის უსაფრთხოების პრობლემას , როგორც ჩვეულებრივი მომხმარებლისთვის ასევე კორპორაციებისთვის. ნებისმიერი ქმედება ინტერნეტში აგზავნის შეტყობინებებს სერვერებთან რომლებიც შესაძლოა ფიზიკურად მდებარეობდეს მსოფლიოში ნებისმიერ ადგილას, რაც ნიშნავს რომ ტრაფიკის მიყურადება შედარებით იოლია, როდესაც შეტყობინება ღიად გაივლის

რამდენიმე ლოკაციას ადრესატამდე მისვლისას, შემტევისთვის საკმარისია მხოლოდ ამ გზის ნებისმიერ მონაკვეთში მოახერხოს შეტყობინების მოსმენა, რომ შეიტყოს ინფორმაცია გამგზავნ-მიმღებზე. აქედან გამომდინარე, უფრო და უფრო მეტი მომხმარებლისთვის გახდა მნიშვნელოვანი ანონიმურობა ინტერნეტით სარგებლობისას.განსაკუთრებით ისეთ ქვეყნებში სადაც არსებობს პოლიტიკური ცენზურა და შეზღუდული არის გამოხატვის თავისუფლება ჟურნალიტებისვის და ასევე ჩვეულებრივი მოქალაქეებისთვის. ანონიმურობა მნიშვნელოვანია სამართალდამცავი პირებისთვისაც, რომლებიც იძიებენ კიბერდანაშაულებს.

არსებობს რამდენიმენაირი კრიპტოგრაფიული ქსელი, რომელიც ცდილობს გაანეიტრალოს ტრაფიკის ანალიზის შეტევები და შეუძლებელი გახადოს გამგზავნ-მიმღების დადგენა. ერთერთი ასეთი ქსელი არის Tor რომელიც იყენებს ტექნოლოგიას (Onion routing) შეტყობინებების დასაშიფრად. ეს ტექნოლოგია თავდაპირველად შეიქმნა ა.შ.შ -ს არმიისთვის. დღეს მის დასახვეწად მუშაობენ მეცნიერები და პროგრამისტები მთელი მსოფლიოდან.

ნაშრომში დაწვრილებით განვიხილავთ თუ როგორ მუშაობს onion routing. რა ხერხები გამოიყენება იმისთვის, რომ დაცული იყოს შეტყობინებები ტრაფიკის ანალიზისგან და როგორ შეიძლება ამ დაცვის გვერდის ავლა. ასევე გარკვეულწილად შევამოწმებთ Tor აპლიკაციის ეფექტურობას მომხარებლის ანონიმიზაციის კუთხით და ასევე თუ რამდენად დაცულია იგი აპლიკაციის დონის შეტევებისგან.

2.1 Onion Routing

Onion routing- ის სახელი გამომდინარეობს მონაცემთა სტრუქტურიდან რითაც ინფორმაცია გადაადგილდება ქსელში. Onion არის შექმნილი კრიპტოგრაფიულად დაშიფრული ფენებით შეტყობინების გამგზავნის მიერ. გაგზავნისას ყოველი მარშუტიზატორი (router) აცლის თითო თითო ფენას ხოლო მიღებისას კი პირიქით უმატებს დაშიფრულ ფენებს.

როგორც აღვნიშნეთ, Onion routing დაფუძვნებულია ,რამდენიმე მარშუტიზატორის გამოყენებით ,მიმღებ და გამგზავნ კომპიუტერს შორის კრიპტოგრაფიული მარშუტის აგებაზე. ამ მარშუტს ეწოდება Onion circuit. Onion circuit ში შემავალი მარშუტიზატორების არჩევა ხდება შემთხვევითად ყველა იმ სერვერებიდან რომლებიც დარეგისტრირებული არიან Onion ქსელში და ყოვლმა კვანძმა იცის მხოლოდ მისი წინა და შემდეგი მარშუტიზატორის არსებობის შესახებ. იმისთვის, რომ შეუძლებელი იყოს ყოველი კვანძისთვის შეტყობინების გახსნა გამგზავნმა კომპიუტერმა უნდა დაარიგოს დაშიფრვის გასაღებები ინდივიდუალურად ყოველი კვანძისთვის. ეს პროცესი ხდება Diffie-hellman პროტოკოლის დახმარებით.

2.2 პირველი თაობა (1996-2004)

Onion routing- ის ძირითადი მიზანია, ანონიმური კავშირის დამყარება onion ქსელიდან გარე (External) სერვერთან. ინფორმაციის უსაფრთხოების კუთხით დიდი მნიშვნელობა აქვს კვანძების რაოდენობას ჯაჭვში (ვგულისხმობთ, მარშუტიზატორების რაოდენობას, გამგზავნის და მიმღების ჩათვლით).იმისთვის რომ დაირღვეს ანონიმურობა, საჭიროა დაკავშირდეს ანონიმური შეტყობინება გამგზავნ-მიმღების წყვილთან. ანუ თეორიულმა შემტევმა უნდა გაიგოს IP მისამართი გამგზავნ-მიმღების. ზოგადად ამის გაგება ხდება პაკეტის header -ის წაკითხვით, თუმცა onion routing ის დროს ის დაშიფრულია. ყველაზე მარტივი შეტევა ასეთ შემთხვევაში მდგომარეობს შემდეგში: შემტევი არეგისტრირებს საკუთარ მარშუტიზატორს, როგორც onion კვანძს და აკვირდება გამავალ შეტყობინებებს.

თუ ჯაჭვი შედგება სამი კვანძისგან (გამგზავნი, onion მარშუტიზატორი, მიმღები) და შუა კვანძი ეკუთვნის შემტევს, მაშინ თავისუფლად შეეძლება გამგზავნ მიმღების IP წყვილის გაგება.ოთხი კვანძის შემთხვევაში (გამგზავნი, ორი onion მარშუტიზატორი, მიმღები), შემტევმა უნდა მოახერხოს კიდევ ერთი კვანძის ხელში ჩაგდება, იმისთვის რომ დაარღვიოს ანონიმურობა.

დეველოპერებმა არ ჩათვალეს ეს მეთოდი საკმარისად უსაფრთხოდ და კვანძების მინიმალური რაოდენობა ჯაჭვში დაწესდა ხუთად (გამგზავნი, სამი onion მარშუტიზატორი, მიმღები).

ძირითადი განსხვავება პირველ და შემდეგ თაობებს შორის არის ის რომ, პირველი თაობის მომხმარებლის პროგრამული უზრუნველყოფა გაერთიანებული იყო Onion მარშუტიზატორთან. რაც ნიშნავდა რომ თუ მომხარებელს სურდა ანონიმურობის სერვისით სარგებლობა აუცილებლად უნდა ჩაერთო საკუთარი მარშუტიზატორი Onion ქსელში. ეს მიდგომა შეიცვალა შემდეგ თაობაში.

2.3 მეორე თაობა (2004 – 2006)

მეორე თაობის ვერსიაში მოხდა რამდენიმე მეთოდის დამატება იმისთვის რომ გაუმჯობესებულიყო ინფორმაციის უსაფრთხოების ხარისხი.ერთერთი ასეთი მეთოდი იყო ტრაფიკისთვის ისეთი შეტყობინებების დამატება რომელიც შეიცავდა შემთხვევით ინფორმაციას (Padding), რაც ართულებს ტრაფიკის ანალიზის ტიპის შეტევის განხორციელებას. მეორე ასეთი ტექნიკა იყენებდა გამტარობის (bandwidth) შეზღუდვას კონსტანტურ სიჩქარეზე, რათა შეუძლებელი ყოფილიყო შემტევისთვის ტრაფიკის დინებაში ცვლილებების გაანალიზების დახმარებით დაერღვია ანონიმურობა.თუმცა ეს ორივე მეთოდი აღმოჩნდა რომ მოითხოვდა დიდ გამოთვლით ხარჯს და არასაკმარისად აუმჯობესებდა ინფორმაციის უსაფრთხოების ხარისხს.ამ ფაქტის გათვალისწინებით ეს ორი მეთოდი აღარ არსებობს მესამე თაობაში. ასევე ერთერთი ყველაზე მნიშვნელოვანი ცვლილება იყო ისეთი ჯაჭვების შექმნა სადაც კვანძების რაოდენობა აღემატებოდა ხუთს. შესაძლებელი იყო თერთმეტკვანძიანი ჯაჭვის შექმნა, რაც ინფორმაციის უსაფრთხოების და ანონიმურობის დაცვის კუთხით უმჯობესი იყო , თუმცა დიდი მოლოდინის დრომ ეს ტექნიკა არაპრაქტიკული გახადა.შესაბამისად ეს მეთოდიც შემდეგ ვერსიაში აღარ გვხვდება და ჯაჭვში კვანძების რაოდენობა ისევ მაქსიმუმ ხუთია.

2.4 მესამე თაობა (2006 –)

მესამე თაობა არის უახლესი ვერსია Onion routing-ის. მისი დამუშავება ამ პერიოდშიც გრძელდება. ჩვენ განვიხლიავთ იმ ფუნქციებს რომლებიც დაიბეჭდა 2006 წელს. სხვაობა მესამე თაობას და წინა თაობებს შორის შედარებით მცირეა ვიდრე პირველ და მეორეს შორის.

ყველაზე დიდი სხვაობა წინა თაობებთან შედარებით, გასაღებების დარიგების მეთოდშია. იგი იყენებს Diffie-Hellman პროტოკოლს რაც განაპირობებს ე.წ Forward secrecy. ეს ნიშნავს, რომ იმ შემთხვევაშიც კი თუ შემტევი მოახერხებს ერთი კვანძისთვის ერთი ფენის გასაღების გაშიფრვას , მას ვერ გამოიყენებს სხვა კვანძის ფენის გასაშიფრად.

	Year published	Mixing added	Length of chain	Client and server	Key distribution protocol
1.Generation	1996 [28]	No	5	Integrated	Using onions
2.Generation	2004 [22]	Yes	1 --11	Separate	Using onions
3.Generation	2006 [22]	No	5	Separate	Diffie-Hellman protocol

Table1. The most distinctive features between different generations

3.1 Tor ქსელი

ზოგადად Tor არის Onion routing-ის იმპლემენტაცია და მისი ქსელი მოიცავს მომხარებლებს, მარშუტიზატორებს და სერვერებს. ქსელის დეცენტრალიზებურ ბუნებას განაპირობებს ის ფაქტი რომ ნებისმიერ მომხმარებელს სურვილისამებრ შეუძლია იყოს ასევე მარშუტიზატორი(კვანძი) ქსელში.2016 წლის მონაცემებით Tor

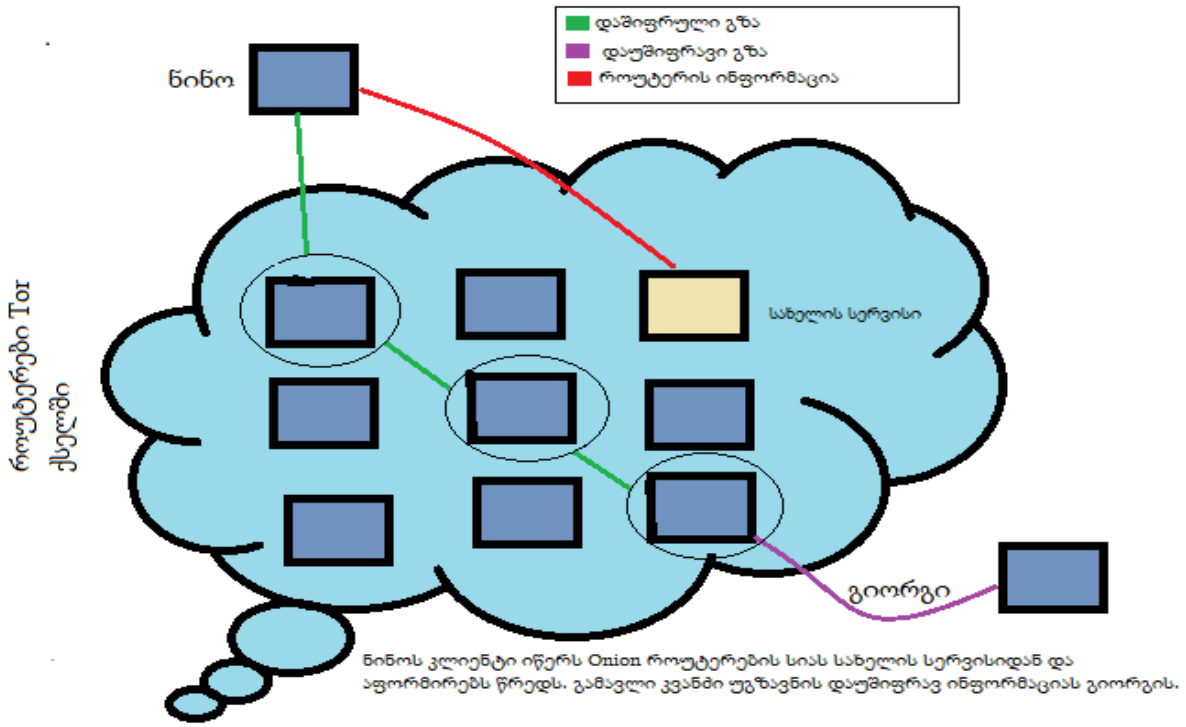
ქსელში გაერთიანებულია 2 000 000 მომხარებელი და 7000 მარშუტიზატორი(კვანძი) ამ უკანასკენლის მნიშვნელოვანი ნაწილი მოხალისე მომხმარებლების პირადი ინფრასტრუქტურაა.Tor ის მესამე თაობაში დამატებულია ე.წ შემავალი მცველები (Entry Guard) კვანძები. მათი მიზანია შეამცირონ იმის შანსი რომ პირველი არჩეული კვანძი ჯაჭვში იყოს შემტევის. თავდაპირველ იმპლემენტაციაში პირველი კვანძიც შემთხვევითობის პრინციპით ირჩეოდა, რაც ნიშნავდა რომ საკმარისი მცდელობის შემდეგ შემტევის მარშუტიზატორი გახდებოდა პირველი კვანძი ჯაჭვში.2006 წლის შემდეგ შემავალი(entry) სამი კვანძი ირჩევა მცირე რაოდენობის სანდო მარშუტიზა-ტორებისგან. ეს სამი მცველი კვანძი გამოიყენება 30 დან 60 დღემდე ყველა წრედისთვის რომელსაც მომხარებელი შექმნის შემავალ კვანძად.ეს მეთოდი ეფექტურია ისეთი შეტევების გასანეიტრალებლად, სადაც შემტევი ცდილობს თავისი მარშუტიზატორი მოახვედროს Onion წრედში (ჯაჭვში). ასევე უფრო მეტი უსაფრთხოებისვის არსებობენ ხიდები (bridges). Tor ის ქსელის მარშუტიზატორების ძირითადი ნაწილის მისამართი არის ღიად გამოცხადებული (public) რაც საშუალებას აძლევს ცენზურის დამწესებელს firewall-ის დახმარებით დაბლოკოს კონკრეტული მისამართები. ისეთი ავტორიტარული ქვეყნები როგორიცაა ჩინეთი და ირანი იყენებენ მსგავს შეტევებს Tor ის წინააღმდეგ. მომხმარებელს რომელსაც არ აქვს წვდომა Tor-ზე ასეთი firewall-ის გამო, შეუძლია გამოიყენოს ხიდები მის გვერდის ასავლელად, რადგაც ხიდის მისამართი არ არის ღიად გამოცხადებული.

3.2 Onion წრედის (ჯაჭვის) შექმნა.

როცა მომხმარებელს სურს დაუკავშირდეს სერვერს Tor ქსელის მიღმა, მომხარებლის პრორამული უზრუნველყოფა პირველ რიგში გადმოწერს რეგისტრირებული აქტიური მარშუტიზატორების სიას. შემავალი კვანძი (Entry node) აირჩევა იმ მარშუტიზატორებისგან, რომელებიც სახელის სერვისებიგან გამოცხადებულია როგორც სანდო და სწრაფი. როცა შემავალი კვანძი აირჩევა

დანარჩენი წრედიც ყალიბდება. კვანძებს რომელთაც გამოცხადებული აქვთ დიდი გამტარობა (bandwidth) და ჩართულობა (uptime) ენიჭებათ უპირატესობა რომ ჩაერთონ წრედში.შემდეგი ნაბიჯი ჯაჭვის შექმნის პროცესში დაშიფრვის გასაღების დარიგებაა Diffie-Hellman ის პროტოკოლით. ინკრემენტულად ხდება ყოველი ახალი წყვილი გასაღების შეთანხმება ყოველ კვანძთან. ამ ნაბიჯის შემდეგ ყოველ მარშუტიზატორს აქვს გასაღები შიფრის ფენის შესაქმნელად და მოსახსნელად, აგრეთვე ინფორმაცია თუ რომელ კვანძთან გადააგზავნონ შეტყობინებები.

ამ პროცესის შემდეგ, შემავალმა კვანძმა იცის რომ მომხმარებელი არის დაკავშირებული, თუმცა ვერ ხედავს საბოლოო მიმართულებას ის მხოლოდ უმისამართებს შეტყობინებას ჯაჭვში მომდევნო კვანძს.მესამე კვანძმა, იცის საბოლოო დანიშნულების მისამართი, თუმცა არ იცის ვინ არის გამომგზავნი და რა ინფორმაცია დევს შეტყობინებაში. ამ კვანძს გამავალი კვანძი(exit node) ეწოდება. საფინალო კვანძი, ანუ მიმღები(არ მდებარეობს Tor ქსელში), იღებს შეტყობინებას რომელიც დაუშიფრავია.

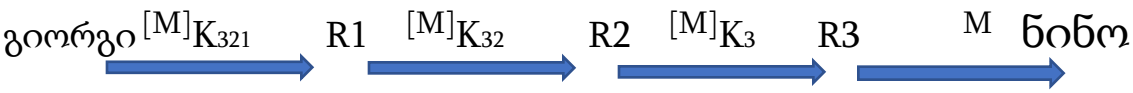


3.3 შეტყობინება Onion წრედში.

მომხმარებელი ქმნის n-ზომის ჯაჭვს, რის შედეგადაც ჯაჭვი შედგება მარშუტიზატორებისგან R_1, R_2, … R_n, რომელთათვისაც მომხმარებელს დარიგებული აქვს სიმეტრიული გასაღები K_i Diffie-Hellman პროტოკოლის გამოყენებით. შეტყობინების გაგზავნამდე მომხმარებელი შიფრავს მას გასაღებით K_n , შემდეგ K_{n-1} და ასე K_1 დონემდე. განვიხილოთ მაგალითი :

გიორგი ანონიმურად უგზავნის შეტყობინებას ნინოს Tor-ის დახმარებით. მან შექმნა onion წრედი სამი კვანძით, R_1 ->R_2->R_3. გამოსახულება $[M]_{Ki}$ ნიშნავს რომ შეტყობინება M დაშიფრულია სიმეტრული გასაღებით K_i. ხოლო $[M]_{Kijk}$ ნიშნავს რომ შეტყობინება M თავდაპირველად დაშიფრულია გასაღებით K_i ,შემდეგ K_j და ბოლოს გასაღებით K_k. ჩვენს მაგალითში გიორგი შეტყობინებას დაშიფრავს პირველად გასაღებით K_3 , შემდეგ K_2, და ბოლოს K_1 -ით. ანუ $[M]K_{321}$.

ყოველი როუტერი ხსნის შიფრის თავის ფენას საკუთარი K_i - გასაღებით და უგზავნის ჯაჭვში შემდეგ კვანძს. შედეგად R_3 მარშუტიზატორი ნინოს გადაუგზავნის M დაუშიფრავ შეტყობინებას.

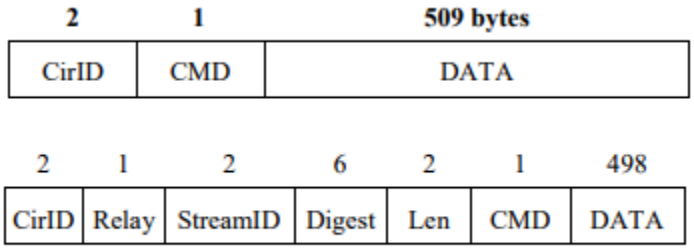


როდესაც ნინო პასუხობს შეტყობინებით M. ანალოგიურად ჯერ გადაუგზავნის R_3 რომელიც დაშიფრავს გასაღებით K_3 და ასე გაგრძელდება გიორგიმდე. რადგან მხოლოდ გიორგიმ იცის სამივე გასაღები მხოლოდ მას შეეძლება შეტყობინების წაკითხვა.

3.4 უჯრედი (Cells).

შეტყობინებებს, რომლებიც კვანძებს შორის იცვლება უჯრედები ეწოდებათ. მონაცემების გადაცემის გარდა არსებობს უჯრედები რომლებიც წრედის შესაქმნელად (CREATE cells), წრედის დასანგრევად (DESTROY cells) და კავშირის შესანარჩუნებლად (PADDING cells) გამოიყენება. ტრაფიკის ანალიზის შეტევის გასანეიტრალებლად უჯრედებს გააჩნიათ ფიქსირებული ზომა 512 ბაიტი. ზემოთ აღნიშნულ უჯრედებს კონტროლ უჯრედებს უწოდებენ მათ Header-ში ჩაწერილია წრედის საიდენტიფიკაციო კოდი circID და დანიშნულება უჯრედის control cell command ანუ CMD.

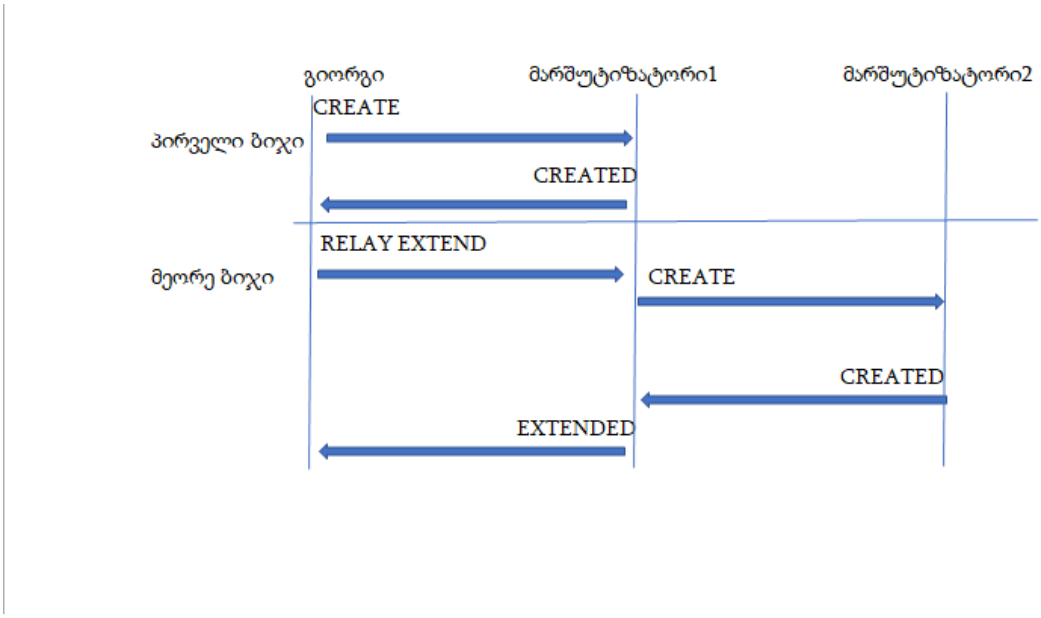
უჯრედებს რომელებიც ატარებენ მონაცემებს ეწოდებათ relay cells. მათ Header-ში უფრო მეტი ინფორმაციაა ჩაწერილი : დამატებით circID და CMD- სა ჩაწერილია ნაკადის მაიდენტიფიცირებელი, ჩეკსამი და მონაცემების ზომა. უშუალოდ მონაცემის სეგმენტი 498 ბაიტის ზომისაა და მას ასევე შეუძლია გადაიტანოს კონტროლ უჯრედები ინდივიდუალურ კვანძებამდე წრედში, მაგალითად Diffie-Hellman handshake ის ფაზაში.



კონტროლ უჯრედის header(ზედა) და relay უჯრედის header(ქვედა)

არსებობს რამდენიმე ტიპის relay უჯრედი. RELAY DATA უჯრედები გამოიყენება უშუალოდ მონაცემების გასაგზავნად და მისაღებად წრედში. RELAY BEGIN უჯრედები ნაკადის გასახსნელად, ხოლო RELAY END დასახურად. ასევე ამ ტიპის

უჯრედები გამოიყენება წრედის შესაქმნელად. RELAY EXTEND უჯრედი ატყობინებს მარშუტიზატორს რომ გადააგზავნოს „handshake” ჯაჭვში მომდევნო მარშუტიზატორთან. პასუხად ბრუნდება CREATED უჯრედი თუ handshake წარმატებულია.



წრედის შექმნის პირველი ორი ნაბიჯი.

4. თავდასხმის საფრთხეები

Tor ის მთავარი მიზანია თავიდან აირიდოს მომხმარებლის იდენტიფიკაცია. შესაბამისად, თუ შემტევმა ნებისმიერი გზით მოახერხა მომხმარებლის ანონიმურობის დარღვევა, მაშინ შეტევა შეიძლება ჩავთვალოთ წარმატებულად. Tor ის უპირატესობა VPN და PROXY სერვისებთან მიმართებით მდგომარეობს შემდეგში : ორივე ზემოთ ხსენებულ ანონიმურობის სერვისი არის ცენტრალიზებული სისტემები ანუ გააჩნიათ ე.წ (single point of failure). რაც ნიშნავს რომ მომხარებლის დეანონიმიზაციისთვის საჭიროა წარმატებული შეტევის განხორციელება მხოლოდ ერთ წერტილში.ზოგადად Onion routing -ის ანონიმურობას განაპირობებს დეცენტრალიზებული ბუნება. რაც უფრო მეტი მომხმარებელი და კვენძია ქსელში

მით უფრო რთულია კონკრეტული აქტივობა ინტერნეტში მივაკუთვნოთ ერთერთ მათგანს.

არსებობს მრავალნაირი თავდასხმის მეთოდები კრიპტოგრაფიული სისტემების წინააღმდეგ. ყველაზე ხშირად Tor ის წინააღმდეგ გამოყენებული თავდასხმები ცდილობენ კონტროლ ქვეშ ჰყავდეთ ერთი ან მეტი მარშუტიზატორი წრედში.ჩვეულებრივ საკმარისია შემავალი კვანძის (entry node) და გამავალი კვანძის (exit node) გაკონტროლება წრედში, რათა შემტევმა მოახერხოს ტრაფიკის ანალიზზე დაყრდნობით მომხარებელის გამოაშკარავება. გამოიყოფა აქტიური და პასიური შეტევის ტიპები. პასიური თავდასხმის დროს ხდება უბრალოდ ტრაფიკის ნაკადის მიყურადება მისი შეცვლის გარეშე. აქტიური თავდასხმისას ხდება ტრაფიკის იმგვარად მოდიფიცირება(პაკეტების დანიშვნა), რომ ადვილი გახდეს მისი ანალიზი.

4.1 დეანონიმიზაციის ცნობილი კატეგორიები.

უკვე არსებული დეანონიმიზაციის ტექნიკებზე დაყრდნობით შეგვიძლია ეს ხერხები დავყოთ ორ ჯგუფად.

- პასიური და აქტიური შეტევები : შემტევი მხოლოდ აკვირდება ტრაფიკს და შემტევი ახერხებს ტრაფიკის მანიპულაციას.
- Single end და End to End შეტევები : შემტევი ახერხებს ქსელის ანონიმურობის დარღვევას Tor ის წრედის მიყურადებით და მანიპულაციით ან მხოლოდ შემავალი კვანძის კონტროლით ან შემავალი და საბოლოო კვანძის კონტროლის მეშვეობით.

შეტევები Tor ის ქსელზე შეგვიძლია დავყოთ მისი მეთოდის და მიზნის შესაბამისად :

- კორელაციის შეტევა (Correlation attack) – End to end პასიური შეტევა
- შეგუბების ტიპის შეტევა (Congestion attack) - End to end აქტიური შეტევა
- დროზე დამოკიდებული შეტევა (Timing attack) – End to end ატიური შეტევა
- ამომცნობი შეტევა (Fingerprinting attack)- Single-end პასიური შეტევა

- DOS (Denial of Service attack) Single end აქტიური შეტევა
- დამხმარე შეტევები

4.2 შეტევები Tor-ზე

საფუვლიანი და მრავალრიცხოვანი კვლევების საგანი გახდა Tor ის მექანიზმების დაუცველობა. ამ ქვეთავში განვიხილავთ რამდენიმე გასაჯაროვებულ კვლევას Tor-ზე თავდასხმების შესახებ.ზოგადად ძალიან დიდი ინტერესი არსებობს Tor -ზე შეტევების განხორციელების მიმართ. მაგალითად, გავრცელდა ინფორმაცია, რომ FBI-მ Carnegie Mellon University -ს გადაუხადა თანხა, რათა შეემუშავებინათ Tor ზე თავდასხმის ეფექტური მეთოდი. აღნიშნული შეტევა ცნობილია როგორც „relay early traffic confirmation attack". ეს შეტევის ტექნიკა განხილული იქნება მომდევნო პარაგრაფში.

4.3 კორელაციის ტიპის შეტევები (Correlation Attacks)

ამ კატეგორიის შეტევებში იგულისხმება , რომ შემტევი აკონტროლებს შემავალ და ასევე გამავალ კვანძებს მომხარებლიდან სერვერამდე არსებულ წრედში. შემტევი ცდილობს დაინახოს კორელაცია შემავალი და გამავალი კვანძების ტრაფიკში. რადგან შეძლოს იმაში დარწმუნება, რომ შემავალი და გამავალი კვანძები შედიან Tor წრედში. შემავალმა წრედმა იცის მომხმარებელი ხოლო გამავალმა კი სერვერი, შესაბამისად შემტვს შეუძლია დაადასტუროს რომ კლიენტი და სერვერი ერთმანეთთან კომუნიკაციაში არიან.

Relay Early Traffic Confirmation შეტევების მიზანია იმ მომხმარებელთა დეანონიმიზაცია, რომლებიც დაფარულ სერვისებს იყენებენ. ცნობილია, რომ ამ ტიპის შეტევები არაერთხელ განხორციელდა ტორის ქსელზე, რის საპასუხოდაც ახალი

თავდაცვის გეგმა შემუშავდა. ასეთი შეტევები წარმოადგენს კორელაციისა და *სიბილ* შეტევის კომბინაციას. *სიბილ* შეტევაზე უფრო ვრცლად ქვევით ვისაუბრებთ.

სიბილ შეტევა გულისხმობს, რომ შემტევის ინფრასტრუქტურა ხდება შემავალი მცველი (entry guard) და დაფარული სერვისის დირექტორია. შემდგომ ხორციელდება კორლაციის შეტევა, რომ დადასტურდეს კომუნიკაცია კლიენტსა და დაფარულ სერვისს შორის. იმისთვის, რომ მსგავსი შეტევა იყოს წარმატებული შემტევს ჭირდება დაფარული სერვისის დირექტორიაზე და მომხმარებლის შემავალ კვანძზე კონტროლი. თუ კლიენტს სურს დაფარულ სერვისთან დაკავშირება ის ითხოვს ე.წ გაცნობის წერტილებს (introduction points) დაფარული სერვისის დირექტორიისგან. დაფარული სერვისის დირექტორიისგან იგზავნება დაფარული სერვისის სახელი წრედში, რომელიც კოდირებულია Relay და Relay early უჯრედების თანმიმდევრობით(pattern).

Relay early უჯრედები ხელს უშლიან მომხარებელს, შექმნას გრძელი წრედები, რომლებიც შესაძლოა გახდნენ „დაგუბებული“ტიპის შეტევების სამიზნეები.

შემავალ კვანძს შეუძლია გაშიფროს დაფარული სერვისის სახელი „ტრაფიკის ფათერნის“ (გრაფიკული გამოსახულების) მიხედვით და აღმოაჩინოს დაფარული სერვისისა და კონკრეტული კლიენტის კავშირი.

გამეორებითი შეტევა [Replay attack]

კიდევ ერთი შეტევის ტიპი, რომელიც ერევა მომხმარებელსა და სერვერს შორის კომუნიკაციაში არის ე.წ გამეორებითი ტიპის შეტევა. ამ ტიპის შეტევის აღწერა გამოქვეყნებულია 2008 წელს.

დავუშვათ, რომ წრედის სიგრძე კლიენტიდან სერვერამდე არის 3ის ტოლი. შემტევი ირჩევს შემავალ როუტერში შემოსულ უჯრედს და აკოპირებს მას. კოპირებული უჯრედი ჩვეულებრივ იგზავნება წრედის მომდევნო რგოლში, მაგრამ მისი გადაგზავნა მხოლოდ მას შემდეგ უნდა მოხდეს რაც წრედი უკვე სრულად შექმნილია. შესაბამისად, კოპირებული უჯრედი უნდა იყოს relay უჯრედი. შემტევს

შეუძლია relay უჯრედის დაფიქსირება გამავალ კვანძში, რომელსაც ის ასევე აკონტროლებს.

ყოველი ტორ. შრე იშიფრება თვლის რეჟიმში. დუბლირებული უჯრედი იწვევს დაშიფრა-გაშიფვრის მთვლელების თანადროული მუშაობის შეფერხებას ესე იგი მათი სინქონიზაციიდან ამოვარდნას, რაც იწვევს გაშიფრის შეცდომებს [decryption error] .

შემტევს შეუძლია დაშიფვრის შეცდომების დეტექტირება გამავალ კვანძში. იმის დასადასტურებლად, რომ შეცდომა გამოწვეულია დუბლირებული უჯრედით, შემტევმა უნდა შეამოწმოს, შეცდომის ფიქსირების დრო ნამდვილად ემთხვევა თუ არა დროს, რომელიც დაჭირდებოდა დუბლირებულ უჯრედს ქსელის გასავლელად. როცა გამავალი კვანძი აფიქსირებს შეცდომას და ამავდროულად „თაიმიგი“ ზუსტია კომუნიკაცია კლიენტსა და სერვერს შორის დადასტურებულია. აქედან გამომდინარე, შემტევი წარმატებით ახორციელებს მომხმარებლის დიანონიმიზაციას.

HTTP based application level attack (HTTP ბაზაზე დაფუძნებული აპლიკაციის დონის შეტევა)

HTTP based application level შეტევა არ არის უშუალოდ Tor-ის ვებ ბრაუზინგზე დამოკიდებული, არამედ დაბალი მოლოდინის დროის აპლიკაციებზე, რომლებიც ეყრდნობიან TCP ნაკადებს. ამ შეტევების ფარგლებში შემტევს შესაძლებლობა აქვს აკონტროლოს რამდენიმე როუტერი (წრედის შემავალი და გამავალი როუტერები). ეს შესაძლებელია იმდენად, რამდენადაც Tor დაფუძნებულია მოხალისეობრივ პრინციპზე. ასეთი შეტევები ხორციელდება იმ სისუსტეების გამოყენებით, რომლებიც HTTP ს აქვს Man in the middle შეტევების მიმართ.

თავიანთი როუტერების რესურსების [სიმძლავრე-სიჩქარის] არარეალურად კარგი მონაცემების გავრცელებით, შემეტევები შანსს იზრდიან Tor ის ალგორითმმა სწორედ მათი როუტერები აირჩიოს წრედის შემავალ და გამავალ როუტერად. თუ კლიენტი გაგზავნის HTTP request ს (მოთხოვნას) შემტევს შეუძლია განახორციელოს „გაყალბებული ვებ.გვერდის შეტევა“ (forged webpage attack). იდეა მდგომარეობს იმაში, რომ კლიენტის ვებ.ბრაუზერმა [არასასურველ ვებ.გვერდებთან კომუნიკაციის

დროს] დააგენერიროს ადვილად გარჩევადი ტრაფიკის გამოსახულება-პატერნი, რომელიც ადვილად ამოცნობადი იქნება შემავალი როუტერისთვის.
ისეთი ტიპის შეტევების თავიდან ასაცილებლად, რომლებიც როუტერების ჰაკინგს იყენებენ საჭიროა არასასურველი როუტერის არჩევის შანსის მინიმიზაცია Tor როუტერების რიცხვის ზრდით. ასევე Tor - *ის* წრედის კონსტრუქციის ალგორითმის გაძლიერებით ისე, რომ აირჩეს მხოლოდ ბოლომდე სანდო და ერთგული როუტერები მკაცრი აუთენტიფიკაცია-ავტორიზაციის და“ რეპუტაციის სისტემის“ გამოყენებით. შეტევის პრევენციის დამატებითი მეთოდებია არანორმალური [abnormal], უჩვეულო ტრაფიკის დაფიქსირება ვებ.ბრაუზერ ფლაგინების გამოყენებით მომხმარებლის მხარეს.

4.4„დაგუბების“ ტიპის შეტევა (congestion type attack)

დაგუბების ტიპის შეტევებში შემტევი ცდილობს მიზანშიამოღებული მომხმარებლის მიერ შექმნილ წრედში შემავალი როუტერების იდენტიფიცირება მოახდინოს . ამ მიზნის მისაღწევად შემტევი აყოვნებს / აგუბებს წრედში შემავალ გადამცემ როუტერებს სათითაოდ და აკვირდება სამიზნის ტრაფიკში მოლოდინის დროებში სხვაობას. შემტევს შეუძლია გაზომოს მოლოდინის დროებში სხვაობა როუტერების დაგუბებით. მაგალითად, მაშინ, როცა კლიენტი იწერს დიდი ზომის ფაილს შემტევის მიერ კონტროლირებადი საიტიდან შემტევი აფიქსირებს გადმოწერის სიჩქარის შენელებას.

პრაქტიკული დაგუბების შეტევა

შეტევის მიზანია დანამდვილებით გაიგოს, რომ კვანძი(შემავალი) მონაწილეობს Tor წრედში მომხარებლიდან გამავალ კვანძამდე. შეტევის პირველი ნაბიჯი მდგომარეობს შემდეგში : (ვუშვებთ,რომ შემტევი აკონტროლებს გამავალ კვანძს) შემტევი უმატებს გამავალი კვანძის შეტყობინებას Javascript-ის კოდს HTML

პასუხში. კოდი მომხარებლის ბრაუზერს აგენერებინებს HTTP მოთხოვნას 1 წამიანი ინტერვალებით.

HTTP მოთხოვნა შეიცავს კონკრეტულ დროს თუ როდის გაიგზავნა, იმისთვის რომ გამავალ კვანძზე მოქმედმა შემტევმა მოახერხოს დროის განსხვავებების კორელაცია. ამ მეთოდის დახმარებით შემტევი გებულობს საშუალო მოლოდინის დროს აღნიშნულ წრედზე.

შემტევი ქმნის დაგუბებას შემდეგი ხერხით: იქმნება წრედი შემტევის კლიენტით მისივე სერვერამდე. ამ წრედში რამდენჯერმე მეორდება მომხარებლის სავარაუდო შემავალი კვანძი. ვინაიდან შეუძლებელია რომ კვანძი პირდაპირ უკავშირდებოდეს თავის თავს, შემტევი უმატებს საკუთარ ორ დიდ გამტარიან კვანძს გზაზე და ქმნის ე.წ loop-ს (ციკლს). შედეგად, სავარაუდო შემავალ კვანძს გააჩნია დამატებითი წრედები რომელსაც უნდა მოემსახუროს. ვინაიდან Tor კვანძებში, პაკეტები სხვადასხვა წრედებისთვის იგზავნება რიგის მიხედვით ე.წ (round robin), შემტევმა შეძლო დაგუბების შექმნა კვანძზე.

შედეგად, javascript ის კოდის მიერ გამოწვეული HTTP მოთხოვნები განიცდიან შეყოვნებას (ე.წ delay) რომელის გაზომვაც შესაძლებელია გამავალ კვანძზე. შეტევის საკმარისად ბევრი გამეორების შემდეგ და საშუალო მოლოდინის დროის გაზომვით, შემტევი რწმუნდება რომ შემავალი კვანძი მონაწილეობს Tor წრედში.

კონკრეტულად მსგავსი შეტევებისგან თავის ასარიდებლად Tor ის ბრაუზერს დაემატა ფუნქცია, რომლიც თიშავს javascript-ს მომხარებლის მხარეს და თავს არიდებს მსგავს Javascript injection ტიპის შეტევებს. თუმცა, რანდგან javascript ფართოდ გამოიყენება ინტერნეტში, მისი გათიშვა მნიშვნელოვნად აუარესებს მომხარებლის გამოცდილებას. (user experience).

4.5 DOS ტიპის შეტევა

Denial of Service (DoS) შეტევები არ გამოიყენება მომხარებლების დეანონიმიზაციისთვის არამედ, ქსელის დასატბორად (flood), რაც იწვევს ძალიან შენელებულ ან საერთოდ მიუწვდომელ კავშირებს. ასევე შესაძლოა აიძულონ ჩვეულებრივი მომხარებლები, რომ ისარგებლონ შემტევის კვანძებით, რადგან სანდო

კვანძები მიუღწეველი გახდნენ შეტევის გამო. ჩვეულებრივ Distributed Denial of Service (DDoS) შეტევის მსხვერპლს ეგზავნება ძალიან დიდი რაოდენობის UDP პაკეტები ბევრი წყაროებისგან, მაგრამ რადგან Tor მხოლოდ იყენებს TCP ნაკადებს ამ ტიპის DDoS შეუძლებელია.

The Sniper შეტევა

2014 წელს გამოქვეყნდა ახალი შეტევა რომელიც იყენებს სისუსტეს Tor-ის ნაკადის კონტროლის (flow control) ალგორითმში ,სახელად The Sniper Attack. შეტევა მოითხოვს მინიმალური მოცულობის რესურსებს და გამოიყენება Tor ის ქსელში კვანძების მწყობრიდან გამოსაყვანად.

Tor ნაკადის კონტროლის მექანიზმს იყენებს იმ შემთხვევებში, როცა დიდი რაოდენობის პაკეტები იგზავნება ერთი ლოკაციიდან მეორეში, მაგალითად კლიენტიდან სერვერის მიმართულებით. დიდი მოცულობის მონაცემების პირდაპირ გაგზავნით შესაძლოა მოხდეს გადავსება მიმღების მხარეს, ამიტომ საჭიროა ნაკადის კონტროლი რომელიც განაპირობებს, პაკეტების თანმიმდევრულ გადაცემას. ნაკადის კონტროლი საშუალებას აძლევს კლიენტს აკონტროლოს მონაცემის გადაგზავნის ტემპი,იმ ფაქტის დახმარებით რომ იგი აიძულებს გამავალ კვანძს რომ დააგროვოს პაკეტები ბაფერში (buffer). ყოველ ჯერზე როცა მიმღები მზადაა მომდევნო პაკეტის წასაკითხად ბაფერიდან იგი აგზავნის SENDME უჯრედებს. გამგზავნი ბოლო ინახავს ბაფერში 1000 უჯრედს და აგზავნის 100 უჯრედს ყოველთვის როცა მიიღებს SENDME უჯრედს(ამავდროულად ამატებს მომდევნო 100 უჯრედს ბაფერში). ნაკადის კონტროლის პროტოკოლი გამოიყნება შემდეგნაირად :

1.კლიენტი ქმნის წრედს Tor ქსელში გარე სერვერთან

2.კლიენტი : აგზავნის მოთხოვნას რომ გადმოწეროს მონაცემები სერვერიდან

3.გამავალი კვანძი:აგზავნის მოთხოვნას სერვეთან და ბუფერში ინახავს მონაცემებს 1000 უჯრედამდე.

4.კლიენტი: კითხულობს მონაცემებს და აგზავნის SENDME უჯრედებს როცა 100 უჯრედი უკვე დაამუშავა

5.გამავალი კვანძი: ამატებს შემდეგ 100 უჯრედს ბაფერში
6.თუ მთლიანი მონაცემი არაა წაკითხული დაბრუნდი მეოთხე ნაბიჯზე.

ამ პროტოკოლის ბოროტად გამოყენება შესაძლებელია რამდენიმე გზით. პირველი ხერხის შემთხვევაში, როცა შემტევი ცდილობს მწყობრიდან გამოიყვანოს შემავალი კვანძი, იგი აფორმირებს წრედს იმგვარად რომ მის მიერ კონტროლირებადი კვანძი არის გამავალი. პროტოკოლი გულისხმობს, რომ ნაკადის კონტროლს ახორციელებს გამავალი კვანძი,მაგრამ რადგან იგი კონტროლდება შემტევის მიერ, ეს კვანძი აგზავნის იმდენ ინფორმაციას რამდენიც უნდა, ანუ უგულვებელყოფს ნაკადის კონტროლით დაწესებულ ლიმიტებს. ამ შეტევის ხრიკი შემდგია : მომხმარებელი ანუ ჩვენს შემთხვევაში შემტევი არ კითხულობს გამოგზავნილ ინფორმაციას ბაფერიდან და გამავალი გვანძი კი აგზავნის დიდი მოცულობის ინფორმაციას, რაც იწვევს მონაცემთა პაკეტების შემავალ კვანძზე დაგროვებას და საბოლოოდ მის მწყობრიდან გამოყვანას.

ამ შეტევისგან თავის დასაცავად, Tor ის დეველოპერებმა შეიმუშავეს სტრატეგია, რამაც ამ ტიპის შეტევა გახადა ნაწილობრივ არაეფექტური. კვანძებში ჩამატდა პროტოკოლი, რომელიც უზრუნველყოფს წრედის განადგურებას თუ კვანძში მისთვის გამოყოფილი ბუფერი გადაივსება. თუმცა ეს შეტევა კვლავაც გამოიყენება დიდი რაოდობის bandwidth-ის ასათვისებლად.

4.6 დამხარე შეტევები

Sybil შეტევა

2010 წელს აქტიური Tor როუტერების რაოდენობა რამდენიმე საათში დრამატულად გაიზარდა. აღმოჩნდა რომ შემტევმა ერთდროულად ჩართო რამდენიმე ასეული როუტერი Tor ქსელში. ეს შეიძლება ერთიშეხედვით ჩანდეს უსაფრთხოდ, თუმცა სწორედ ეს ხერხი გამოიყენება შეტევების კატეგორიაში Tor ქსელზე რომელთაც ეწოდებათ Sybil.

ამ ტიპის თავდასხმებში, შემტევი აკონტროლებს მრავალ ვირტუალურ იდენტობებს,რათა მოიპოვოს არაპროპორციულად დიდი გავლენა ქსელზე. მრავალი ტიპის შეტევის ეფექტურობა დამოკიდებულია იმაზე თუ რა რაოდეობის ტრაფიკის მეთვალყურეობა შეუძლია მას. მაგალითი ასეთი შეტევების, რომელთა განხორციელებაც იოლდება Sybil შეტევასთან ერთად კომბინირებით, არის კორელაციის და ამომცნობი ტიპის შეტევები.

გარდა სხვა შეტევების გაიოლებისა, Sybil შეტევა რისკის ქვეშ აყენებს თვითონ Tor ქსელის გამოყენებას, რადგან Tor ქსელის ეფექტურობა განპირობებულია Tor კვანძების სანდოობით. არასანდო კვანძები აზარალებენ მომხმარებლის გამოცდილებას და ასევე საფრთხეს უქმნიან მათ ანონიმურობას,რომელიც წესით გარანტირებულია Tor ის მიერ. გარკვეული მომხარებლები წყვიტავენ Tor ით სარებლობას, როცა აწყდებიან პრობლემებს, რომლებიც შექმნილია არასანდო კვანძების მიერ. ნაკლები Tor ის მომხმარებელი კი თავის მხრივ ნიშნავს ზოგადი ანონიმურობის ხარისხის დაკლებას ქსელში. ის მომხმარებლები კი რომლებიც აგრძელებენ დაბალი ანონიმურობის ხარისხის გარემოში Tor ის გამოყენებას, არიან კარგი სამიზნეები დაკვირვებისთვის.

პრაქტიკული თავდაცვა ამ ტიპის შეტევისგან საკმაოდ რთულია და სავარაუდოდ ყოველთვის იქნება შესაძლებელი მათი განხორციელება, რადგან თავად სისტემა დეცენტრალიზებულია. თუმცა, რადგან Sybil კვანძები გარკვეულწილად გვანან ერთმანეთს და იქცევიან ერთნაირად, არსებობს ხერხები მათ დასადგენად გარკვეულ დონემდე. კვანძები, რომლებიც ეკუთვნიან Sybil შეტევას, ხშირად უერთდებიან და ტოვებენ Tor ქსელს ერთდროულად. მათ გააჩნიათ საერთო კონფიგურაციის პარამეტრები და ხშირად იცვლიან თავიანთ იდენტობას, იმისთვის რომ მოახდინონ მანიპულაცია Tor ის ჰეშ ცხრილში.

5 Tor აპლიკაციის ეფექტურობის შეფასება.

Tor-ის ეფექტურობის შეფასებისთვის გამოვიყენებთ Web აპლიკაცია ip-check.info-ს, რომელიც ეკუთვნის JonDonym (Private and Secure Web Surfing) პროექტს. იგი იყენებს ანონიმურობის ტესტებს და გვიჩვენებს შედეგებს თვალსაჩინოდ. ეს ტესტები დაყოფილია სამ ნაწილად :

- Plug-in ტესტები (Flash,Java)
- HTTP/HTML ტესტები
- CSS/JavaScript ტესტები

მეტი თვალსაჩინოებისთვის, Tor აპლიკაციას რომელიც იყენებს სპეციალურად მასზე მორგებული Mozilla Firefox ის იმპლემენტაციას, შევადარებთ პოპულარულ ვებ ბრაუზერს Google Chrome-ს. ასევე, ტესტის პირობებში გამოვიყენებთ ანონიმიზაციის განსხვავებულ მეთოდს ვებ პროქსის და გავაკეთებთ შედარებას.

უნდა აღვნიშნოთ, რომ ხშირშემთხვევაში დეანონიმიზაციის ძირითადი მიზეზი მომხარებლის ვებ ბრაუზერია განსაკუთრებით კი მომხარებლის მიერ დაყენებული ფლაგინები. ყოველ მომხარებელს, გააჩნია ბრაუზერის/კომპიუტერის უნიკალური პარამეტრები და ყოველთვის, როცა უკავშირდება ვებგვერდს, შემტევს შეუძლია მიიღოს გამოსადეგი ინფორმაცია.

5.1 ტესტი ip-check.info -ს დახმარებით, ბრაუზერის Default პარამეტრებით.

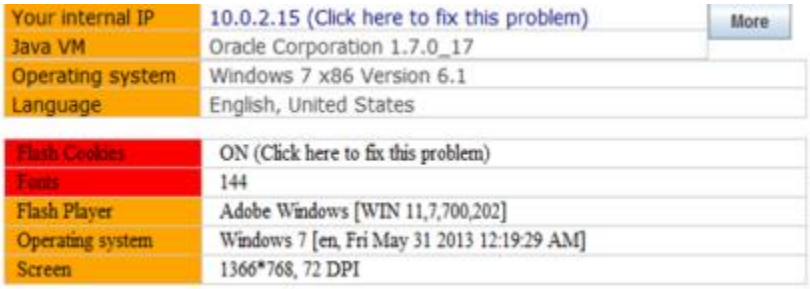
Your IP	149.[redacted]	Traceroute
Your location	Dushet'is Raioni, Tbilisi	Show on map
Your net provider	2`□	Whois IP

პირველ, გრაფაში ჩვენ ვხედავთ მომხარებლის IP მისამართს, ადგილმდებარეობას და ინტერნეტ პროვაიდერს.

შემდეგი გრაფა შეიცავს ინფორმაციას HTTP/HTML ტესტებიდან, ასევე HTTP სესიას ქუქიებს და აუტენტიფიკაციის მონაცემებს.

Attribute	Value	Rating
Cookies	This web site may receive cookies from you	medium
HTTP session	unlimited	bad
Signature	1142a9b979396a415ad2a8176e56b2f9	good
User-Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.99 Safari/537.36	bad
SSL_session_id		neutral
Language	en-US,en;q=0.9	medium
Content types	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8	medium

Adobe Flash player და Java პლაგინები კითხულობენ მომხარებლის ოპერაციული სისტემის მონაცემებს და ასევე შიდა ქსელის ინფორმაციას.



Your internal IP	10.0.2.15 (Click here to fix this problem)	More
Java VM	Oracle Corporation 1.7.0_17	
Operating system	Windows 7 x86 Version 6.1	
Language	English, United States	

Flash Cookies	ON (Click here to fix this problem)
Fonts	144
Flash Player	Adobe Windows [WIN 11,7,700,202]
Operating system	Windows 7 [en, Fri May 31 2013 12:19:29 AM]
Screen	1366*768, 72 DPI

საბოლოო ტესტი აკვირდება CSS და JavaScript ის მონაცემებს. აღსანიშნავია რომ სწორედ JavaScript-ის გამოყენება ქმნის მრავალ პრობლემას ანონიმურობის შენარჩნუნების მიმართულებით. ჩვენს შემთხვევაში შემტევს შეუძლია ამოიკითხოს დაინსტალირებული პლაგინების სია და გამოიყენოს ერთ ერთი მათგანის სისუსტე.

Attribute	Value	Rating
JavaScript	JavaScript is activated! (Version: 1.7)	medium
Plugins	Found 4 plugins. Flash is active!	bad
Mime types	Found 6 mime types that your browser supports.	medium
Tab name	"window.name" is traceable. Your unique ID: 0630826	medium
Tab history	There are 5 pages in your tab history.	medium
Local storage	Local storage is enabled. Your unique ID: 10630826	medium
Screen	1920 x 1080 pixels 24 bit color depth	medium
Screen (usable)	1920 x 1040 pixels (does not match screen)	medium
Browser window	1920 x 898 pixels (inner size)	medium
Browser bars	MenuBar PersonalBar StatusBar ToolBar ScrollBars LocationBar	good
WebGL	WebGL is activated, WebGL 1.0 (OpenGL ES 2.0 Chromium), WebKit WebGL	medium
Browser type	20030107 Netscape (en-US)	medium
System	Win32 (Sun Jul 01 2018 05:48:44 GMT+0400 (Georgia Standard Time))	medium
Fonts	139 installed fonts have been found on your computer.	medium

შედეგებს აქვს სამი მნიშვნელობა : „attribute”, “value”, და „“rating”. „attribute” არის ტესტის დასახელება და ფორმდება სამი შესაძლო ფერით , რომელიც შეესაბამება „rating”-ს. მწვანე ფერი მიანიშნებს რომ ტესტზე დაყრდნობით მომხარებელს აქვს ოპტიმალური პარამეტრები. სტაფილოსფერი- შესაძლოა მოხდეს შემტევის მიერ ამ ინფორმაციის მომხმარებლის საზიანოდ გამოყენება. წითელი ფერი მიანიშნებს, რომ ამ პარამეტრების შენარჩუნებით მომხარებელი საფრთხეში აგდებს ანონიმურობას, რადგან ფართოდ ცნობილია ხერხები რომელსაც შემტევი გამოიყენებს დეანონიმიზაციისთვის.შესაბამისად პარამეტრების ცვლილება არის აუცილებელი.

5.2 ტესტი ვებ პროქსის და Tor-ის გამოყენებით

იმის დასადგენად თუ რამდენად ეფექტურია პროქსი სერვერი შევასრულოთ მსგავსი ტესტები.

Your IP	85.17.24.66 (Proxy) 149[redacted] [JavaScript]	Traceroute
Your location	Dushet'is Raioni, Tbilisi	Show on map
Your net provider	JJ□	Whois IP

როგორც, ტესტის შედეგი გვეუბნება, ვებ აპლიკაცია არამარტო მიხვდა რომ HTTP მოთხოვნა მოდიოდა ვებ პროქსის გავლით, არამედ გამოააშკარავა რეალური მომხმარებლის IP

მისამართი, ლოკაცია და ინტერნეტ პროვაიდერი, პროქსი სერვერის უკან. ეს შესაძლებელი გახადა JavaScript ის ცნობილი სისუსტის ექსპლუატაციამ.

შევამოწმოთ იმავე ხერხით Tor ის ქსელთან დაკავშირებული Tor აპლიკაციის კუთვნილი ბრაუზერი.

Your IP	62.210.129.246 (Tor)	Traceroute
Your location	France	Show on map
Your net provider	Free SAS	Whois IP
Reverse DNS	relay1.tor.openinternet.io	Whois Domain

Attribute	Value	Rating
Cookies	This web site may receive cookies from you	medium
HTTP session	10 minutes (until your Tor identity is changed)	medium
Signature	8ab3a24c55ad99f4e3a6e5c03cad9446 (Firefox)	good
User-Agent	Mozilla/5.0 (Windows NT 6.1; rv:52.0) Gecko/20100101 Firefox/52.0	good
SSL_session_id	E1F69E77C41BC6993D5D243D47A61F9FEDE867D82ADEA1221B8AEBEF9791FE5D	neutral
Language	en-US,en;q=0.5	good
Content types	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8	good
Encoding	gzip, deflate	good
Do-Not-Track		good
Upgrade-Insecure-Requests	1	good

Attribute	Value	Rating
JavaScript	JavaScript is currently turned off.	good
Browser window	1000 x 900 pixels (inner size)	good
Fonts	Do you see strange symbols here? If yes, your fonts are readable!	good

როგორც, ვხედავთ IP მისამართად ფიქსირდება გამავალი კვანძის მისამართი. ტესტების უმრავლესობა მიგვანიშნებს, რომ პარამეტრები ვებ ბრაუზინგისთვის ოპტიმალურია. Tor აპლიკაცია ავტომატურად თიშავს JavaScripts , თუ მოვისურვებთ მის ჩართვას, გაფრთხილების შეტყობინება მოგვდის, რომ შესაძლოა შემტევმა გამოიყენოს იგი ჩვენი დეანონიმიზაციისთვის. აქვე უნდა აღინიშნოს ის ფაქტი, რომ ეს ტესტი, სულაც არ განაპირობებს იმას, რომ მომხარებლის ანონიმურობა

გარანტირებულია ნებისმიერი შეტევის მიმართ. უბრალოდ გვაჩვენებს რომ, უკვე არსებული ცნობილი Application დონის შეტევების მიმართ, აპლიკაცია არის რეზისტენტული თუ გამოვიყენებთ ოპტიმალურ პარამეტრებს ვებ ბრაუზინგისთვის.

6. დასკვნა

Tor ის განვითარება ამ დროისთვისაც აქტიურად მიმდინარეობს. ხდება აღმოჩენა ახალი დეფექტების და მათი გასწორება. დამტკიცებულია, რომ მას ნშეუძლია გაუძლოს თავდასხმების დიდ ნაწილს , ხოლო ყოველი წარმატებული თავდასხმა სარგებლობს დეფექტებით არა Tor პროტოკოლში არამედ, მომხმარებლის დაუდევრობით ან JavaScript ის შეუთავსებლობით Tor აპლიკაციასთან.Tor ის დეველოპერების თქმით, Tor ის ანონიმურობას ყველაზე დიდ საფრთხეს უქმნის ისეთი დამკვირვებელი რომელსაც შეუძლია დააკვირდეს მთლიან ავტონომურ სისტემას (AS), რაც ქსელის დიდ ნაწილს გულისხმობს. ვინაიდან Tor ის მიზანია დარჩეს დაბალი მოლოდინის დროით გამორჩეულ სერვისად, გარკვეულწილად გაკეთებულია დათმობები ინფორმაციის უსაფრთხოების ხარჯზე. AS დონის დამკვირვებლის (შემტევის) განეიტრალებისთვის ისეთი მეთოდების გამოყენება გახდება საჭირო რომელიც მნიშვნელოვნად გაზრდის გამოთვლით ხარჯს და შესაბამისად მოლოდინის დროს, რაც Tor ის ძირითად იდეას ეწინააღმდეგება.

თუმცა, რაც უფრო იზრდება Tor ქსელი, გლობალური დამკვირვებლისთვის (AS დონის) უფრო დიდი რესურსი ხდება საჭირო ტრაფიკის ანალიზის ჩასატარებლად. Tor არის open source პროექტი, ხოლო დეველოპერები ხელს უწყობენ ახალ მომხარებლებს ჩაერთონ მის ქსელში, ასევე გახდნენ დეველოპერები და აღმოაჩინონ დეფექტები.ამ დროისთვის Tor ისევ რჩება საუკეთესო ანონიმურობის სერვისად დაბალი მოლოდინის დროით.

7. გამოყენებული ლიტერატურა

[1] Tariq Elahi, Kevin Bauer, Mashael AlSabah, Roger Dingledine, Ian Goldberg. *Changing of the Guards:* Workshop on Privacy in the Electronic Society (WPES '12), October 2012.

[2] Rob Jansen, Kevin Bauer, Nick Hopper, Roger Dingledine. *Methodically Modeling the Tor Network*. Usenix Workshop on Cyber Security Experimentation and Test (CSET '12), August 2012

[3] Roger Dingledine and Nick Mathewson. "Anonymity Loves Company: Usability and the Network Effect." In *Security and Usability*, an O'Reilly Media book, August 2005

[4] Abbott, T. G., Lai, K. J., Lieberman, M. R., & Price, E. C. (2007, June). Browserbased attacks on Tor. In Privacy Enhancing Technologies .Springer Berlin Heidelberg.

[5] Bauer, K., McCoy, D., Grunwald, D., Kohno, T., & Sicker, D. (2007, October. In Proceedings of the 2007 ACM workshop on Privacy in electronic society

[6] Chakravarty, S., Barbera, M. V., Portokalidis, G., Polychronakis, M., & Keromytis, A. D. (2014, March). Springer International Publishing.

[7] Evans, N. S., Dingledine, R., & Grothoff, C. (2009, August). A Practical Congestion Attack on Tor Using Long Paths. In USENIX Security Symposium

[8] Hopper, N., Vasserman, E. Y., & Chan-Tin, E. (2010. ACM Transactions on Information and System Security (TISSEC)

[9] Jansen, R., Tschorsch, F., Johnson, A., & Scheuermann, B. (2014). The sniper attack: Anonymously deanonymizing and disabling the Tor network. OFFICE OF NAVAL RESEARCH ARLINGTON VA

[10] Kwon, A., AlSabah, M., Lazar, D., Dacier, M., & Devadas, S. (2015). Circuit fingerprinting attacks: passive deanonymization of tor hidden services. In 24th USENIX Security Symposium (USENIX Security 15)

[11] Ling, Z., Luo, J., Yu, W., Fu, X., Xuan, D., & Jia, W. (2009, November). A new cell counter based attack against tor. In Proceedings of the 16th ACM conference on Computer and communications security